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

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2024 р.

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

Виконав здобувач вищої освіти
IV курсу, групи КБ-20
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Іваненко М.О.
«____» _____ 2024 р.

Керівник проекту
докт. техн. наук, професор
_____ Смірнов О.А.
«____» _____ 2024 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

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

Спеціальність *125 “*Кібербезпека*”*

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

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2024 року

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

Іваненку Максиму Олеговичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки антивірусного захисту файлових серверів*

2. Керівник роботи *Смірнов Олексій Анатолійович, докт. техн. наук, професор*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 135-02 від 01.04.2024 року

3. Строк подання студентом роботи до захисту *23.05.2024 р.*

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

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

1. Призначення та область використання.

2.Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи кібербезпеки в промислову експлуатацію.

6. Висновки

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

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

7. Дата видачі завдання « 17 » січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2024 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2024 р.	
3.	Розробка моделі компонента	20.03.2024 р.	
4.	Розробка структур даних	25.03.2024 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2024 р.	
6.	Програмування алгоритмів	10.04.2024 р.	
7.	Оформлення ПЗ	17.04.2024 р.	
8.	Попередній захист роботи	23.05.2024 р.	

Дата видачі завдання
« 17 » січня 2024 р.

Підпис керівника
_____ Смірнов О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2024 р.

Підпис здобувача
_____ Іваненко М.О.
(прізвище та ініціали)

АНОТАЦІЯ

Іваненко М.О. Програмне забезпечення системи кібербезпеки антивірусного захисту файлових серверів. 125 Кібербезпека. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

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

Метою розробки є програмне забезпечення системи кібербезпеки антивірусного захисту файлових серверів.

Результат роботи – програмна реалізація системи кібербезпеки антивірусного захисту файлових серверів.

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 з ОС Windows 10/11.

Програму розроблено в середовищі Delphi 10.

Ключові слова: кібербезпека, антивірус, файловий сервер

ABSTRACT

Ivanenko M.O. Software of the cyber security system of antivirus protection of file servers. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

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

The goal of the development is the software of the cyber security system of antivirus protection of file servers.

The result of the work is the software implementation of the cyber security system of antivirus protection of file servers.

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

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

The program can be used on a PC with Windows 10/11 OS.

The program was developed in the Delphi 10 environment.

Keywords: cyber security, antivirus, file server

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

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

ВСТУП

Актуальність теми. Безпека файлового сервера стосується політик, процедур і технологій, які використовуються для захисту файлів і даних на мережевих серверах від несанкціонованого доступу, зміни або знищення. У сфері кібербезпеки та антивірусного захисту це вважається фундаментальним елементом. У більш широкому сенсі файловий сервер – це надійний комп'ютер, який відповідає за зберігання, керування та обмін файлами та даними через мережу для кількох користувачів або клієнтських комп'ютерів.

Під час обговорення контексту безпеки файлового сервера основна увага приділяється тому, щоб файли, розміщені, надані або використані, не потрапили до чужих рук – із наміром змінити, викрасти чи видалити файли, що призведе до витоку даних . Ці заходи також повинні працювати для пом'якшення потенційно серйозних загроз, таких як віруси, шпигунське програмне забезпечення та троянські програми, які можуть поширюватися мережею та завдавати значної шкоди.

Стратегії кібербезпеки використовують різні тактичні та стратегічні рівні для захисту файлових серверів від потенційних загроз. Автентифікація користувача є основною лінією захисту, яка гарантує, що лише особи з законним доступом можуть взаємодіяти з ресурсами сервера. Це можна вирішити за допомогою паролів, облікових записів користувачів sperate, біометричної перевірки або двофакторної автентифікації.

Прийняття принципів найменших привілеїв і необхідності знати може додатково зменшити ймовірність несанкціонованого доступу або пошкодження. Особи повинні мати доступ лише до ресурсів сервера, необхідних для виконання своїх службових обов'язків, і не більше. Також поширеними є системи запобігання та виявлення вторгнень, які виявляють моделі неналежної чи

підозрілої діяльності та автоматично вживають запобіжних заходів або подають тривогу.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки антивірусного захисту файлових серверів.

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

– Огляд існуючих систем антивірусного захисту файлових серверів.

– Дослідження системи кібербезпеки антивірусного захисту файлових серверів.

– Програмна реалізація системи кібербезпеки антивірусного захисту файлових серверів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі антивірусного захисту файлових серверів.

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

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Один із компонентів безпеки файлового сервера стосується саме вірусів і шкідливих програм. Антивірусні програми використовуються для запобігання, виявлення та видалення цих загроз. Регулярно оновлювані визначення вірусів гарантують, що програмне забезпечення може ідентифікувати та блокувати найновіші віруси, а звичайне сканування допомагає виявити будь-які загрози чи вразливості.

Брандмауери також пропонують фармацевтичний захист файлових серверів. Вони або дозволяють, або забороняють трафік через мережу на основі встановлених правил і протоколів. Це гарантує, що лише дозволені команди або фрагменти даних із перевірених джерел потраплять на сервер.

Крім захисту самого файлового сервера, необхідно також вжити заходів для захисту файлів, що зберігаються на ньому. Це включає такі технології, як шифрування, коли дані у файлі перетворюються на нечитабельний формат, який можна розшифрувати лише за допомогою правильного ключа дешифрування.. Це гарантує, що навіть якщо файловий сервер буде зламано, дані у створених файлах залишаться недоступними для неавторизованих користувачів.

Безпека файлового сервера також значною мірою залежить від узгоджених стратегій резервного копіювання та відновлення . У разі найгіршого сценарію – ефективної кібератаки, яка призведе до значної втрати даних – резервні копії файлів служать точкою відновлення, що дозволяє організаціям мінімізувати збитки та відновити роботу. Використання комбінації локальних і зовнішніх резервних копій додає додатковий рівень безпеки.

Регулярні оновлення серверів і технічне обслуговування обладнання сприяють запобіганню вразливостей, якими можуть скористатися кіберзлочинці.

Операційні системи (ОС) і програми потребують постійного оновлення для безперебійної роботи та захисту від нових загроз.

Безпека файлового сервера необхідна для збереження конфіденційних даних і пом’якшення потенційних загроз, що, як наслідок, допоможе підвищити загальну продуктивність і стабільність організації без страху втрати важливих даних. У багатовимірному розумінні безпека файлового сервера інкапсулює захист апаратного забезпечення, програмного забезпечення, даних у дорозі та даних у стані спокою – справді, це важлива перевага сфери кібербезпеки.

Безпека файлового сервера передбачає складну взаємодію підзвітності користувачів, надійних заходів кібербезпеки, потужних антивірусних рішень , постійного обслуговування сервера, жорстких систем резервного копіювання, оперативного аналізу загроз і механізмів реагування. Організації та фахівці з кібербезпеки приділяють пильну увагу цьому аспекту, значно знижуючи ймовірність витоку даних, забезпечуючи безперервність бізнесу та зберігаючи довіру споживачів.

1.2 Область застосування

Областю застосування розроблювальної програми є файлові серверу або як їх ще називають FTP-серверу.

У мережі для зберігання більших обсягів даних існують FTP-серверу. FTP-сервер представляє із себе своєрідну бібліотеку файлів. Для перекачування файлів між FTP-серверами й комп'ютером користувача використовується протокол FTP (File Transfer Protocol – протокол передачі файлів).

Для чого потрібний FTP-сервер? Можна завантажувати на свій комп'ютер файли, які знаходяться на численних FTP-серверах. У мережі існують тисячі FTP-серверів, що надають безкоштовний анонімний доступ до гігабайтів найрізноманітнішої інформації: текстовим документам, дистрибутивам програм, фотографіям і музичним файлам. За FTP-протоколом можна закачувати свої

домашні сторінки на безкоштовні сервери, що надають під них місце. Це набагато зручніше, ніж застосовувати HTTP, коли на спеціальній сторінці серверу ви вказуєте файли, які треба захитати.

При використанні FTP варто пам'ятати деякі особливості цього сервісу, що прямо випливають із тієї операційної системи, де він виник – UNIX. Будь-який FTP-сервер завжди вимагає авторизації користувача, тобто уведення його ім'я й пароля. Залежно від цього користувачеві буде наданий доступ лише до певних каталогів і файлів разом з можливістю здійснювати тільки дозволені дії над умістом FTP-сховища.

Що ж робити, якщо ви не є зареєстрованим користувачем? Практично кожний FTP-сервер надає так званий анонімний вхід (інша назва цього сервісу – анонімний FTP). Для анонімного (або гостьового) входу на сервер необхідно замість імені користувача вказати ключове слово anonymous і як пароль набрати адресу своєї електронної пошти. Після чого вам буде наданий доступ до загальних каталогів, до даних, якими власник серверу хоче поділитися. Звичайно, у такому режимі доступу до серверу користувач може тільки переглядати каталоги й завантажувати файли до себе на диск. Цей спосіб роботи із загальнодоступними FTP-серверами називається анонімним FTP. Деякі сервери створюють спеціальні каталоги, куди кожний бажаючий також може захитати свої власні файли.

Для роботи з FTP-сервером можна використовувати звичайний WWW-браузер. Після набору в рядку адреси URL бажаного FTP-серверу ваш браузер підключиться до нього й виведе вміст віддаленого каталогу.

Як виглядає URL для FTP-серверу?

Для підключенню до FTP-серверу через WWW-браузер необхідно використовувати наступну форму запису URL (Uniform Resource Locator).

При використанні FTP-серверу, що вимагає авторизації:

ftp://ім'я_користувача:пароль@адреса_ FTP-серверу:порт/шлях_до_файлу

При використанні анонімного FTP-серверу:

ftp://адреса_ FTP-серверу/шлях_до_файлу

На перший погляд все добре й подібне використання браузера в якості FTP-клієнта досить зручно. Однак необхідно відзначити, що при цьому способі роботи з FTP-сервером відсутня можливість докачки файлу. Якщо зв'язок із сервером раптово обірвалася й ви не встигли скачати файл цілком (що трапляється досить часто при викачуванні більших файлів з дуже віддалених серверів), то вам доведеться завантажувати весь файл із самого початку. Це одна із чималої кількості досить вагомих причин, що змушують використовувати при роботі з FTP-сервером окремий FTP-клієнт. FTP-клієнт дозволяє переписувати (вивантажувати, посилати) файли на FTP-сервер і, що зустрічається чаші, переписувати (завантажувати, одержувати) файли з FTP-серверу.

У мережі можна знайти велику кількість FTP-клієнтів для всіляких операційних систем:

WS_FTP – найпростіша безкоштовна програма для роботи з FTP-серверами.

LeechFTP – безкоштовний, але досить потужний FTP-клієнт із широкими можливостями й зручним графічним інтерфейсом. Особливістю цієї програми є можливість роботи з декількома FTP-серверами одночасно.

CuteFTP – потужний FTP-клієнт із великим набором різноманітних функцій, що займає лідируючі позиції серед аналогічних програм. Необхідно відзначити, що CuteFTP – комерційний продукт і більшість його можливостей будуть відключені після місяця безкоштовного використання. Якщо LeechFTP здійснює одночасно кілька з'єднань із різними FTP-серверами, то CuteFTP по черзі взаємодіє з кожним сервером і послідовно виконує завдання, які записані в створену їм "черга". CuteFTP підтримує автоматичне продовження завантажування або викачування файлів із серверу, порівняння віддаленого каталогу й каталогу на локальному диску, пошук файлів у мережі.

Примітка: FTP розділяє файли на шість різних типів, з яких корисні тільки два: ASCII і двійковий. Файл типу ASCII – це текстовий файл. Двійковим файлом

називається все інше. Існує два режими FTP для перекачування цих типів файлів: ASCII і двійковий (його називають ще дзеркалъним). Коли ви переписуєте ASCII-файл між комп'ютерами різних типів з різними способами зберігання інформації в режимі ASCII, це перетворення виконується автоматично й тому на приймаючій машині він записується у вигляді зрозумілого для нас текстового файлу. Двійковий же файл не обробляється й передається в незмінному виді.

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

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Розглянемо декілька сучасних антивірусів.

AVG Anti-Virus Free Edition

У тестах на виявлення реальних шкідливих програм, що дозволяють оцінити, наскільки успішно антивірусний пакет реагує на нові погрози, безкоштовний AVG блокував 85,7% атак, продемонструвавши найкращий результат серед протестованих програм. Він виявив 98,7% з більш ніж 129 тис. зразків шкідливих програм, що з'явилися в нашім «зоопарку» за останні чотири місяці. Це гарний результат, хоча й трохи уступає найкращим досягненням, які нам довелось спостерігати.

AVG виявився лідером у своєму класі при нейтралізації активних шкідливих програм. На нашім тестовому ПК він виявив і блокував всі шкідливі програми, причому від двох третин з них не залишилося ніяких слідів перебування.

Пакет сканує ПК досить швидко. При перевірці 4,5 Гбайт даних він упорався зі своїм завданням за 1 хв 35 с. Він показав третій результат у тестуванні, усього на 4 з поступившись переможцеві. Процедура сканування файлів при обігу, що ініціюється при відкритті або збереженні файлу, зайняла при тому же обсязі даних 4 хв 55 с. За цим критерієм AVG перебуває в середині рейтингу. Основний інтерфейс сподобався, а от режим розширеного налаштування може налякати деяких користувачів. Але якщо потрібна швидкий і надійний захист ПК, має сенс зупинитися саме на AVG.

Avast Free Antivirus

Безкоштовна версія Avast фінішувала в тесту другою. Пакет Free Antivirus, що має універсальний інтерфейс, по більшій частині відмінно впорався із захистом від шкідливих програм. Він успішно пройшов всі наші тести.

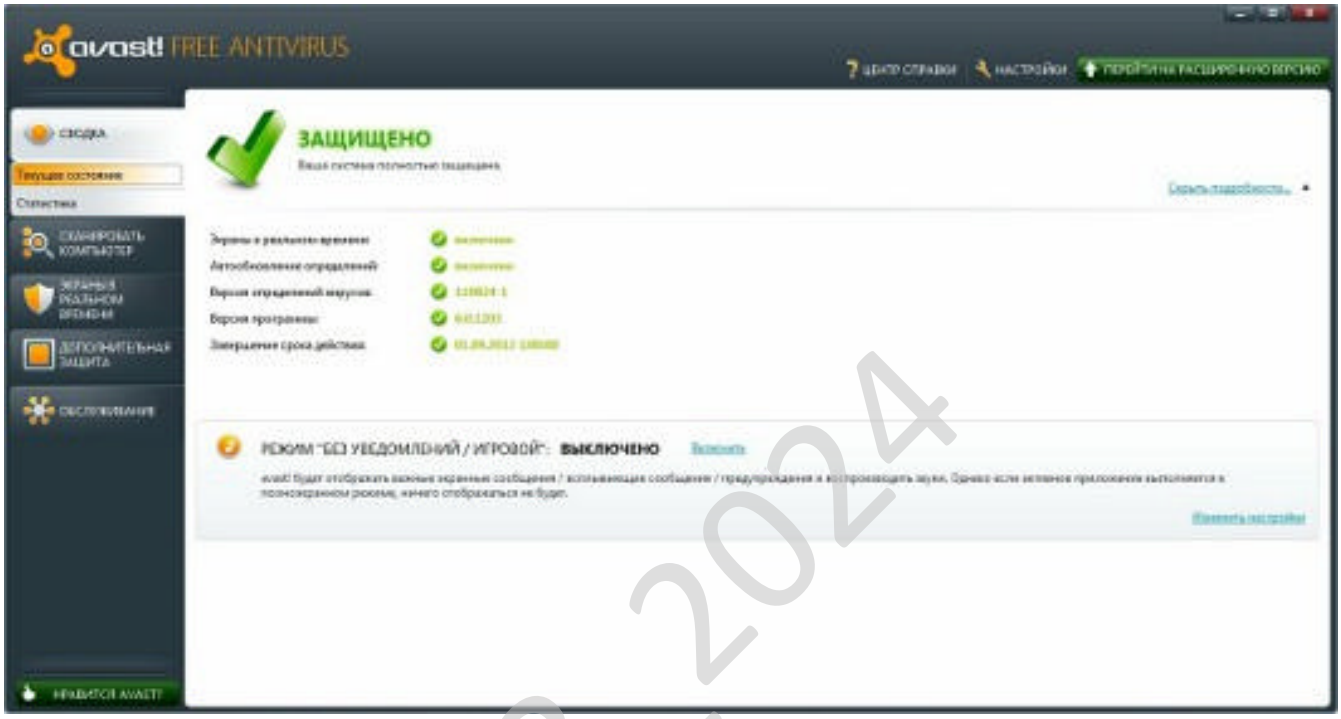

Рисунок 2.1 – Інтерфейс користувача Avast

Самий слабкий захист Avast виявилася при блокуванні нових реальних погроз. Антивірус зумів виявити 78,6% з їхнього числа. Результат – не видатний, але трохи вище середнього. Набагато краще Avast виявив себе при блокуванні відомих шкідливих програм, виявивши 99,1% екземплярів – дуже гарний показник. Пакет ефективно справляється з очищенням комп'ютера. Він знайшов всі шкідливі програми, що були присутні на тестовому ПК, і знешкодив 93% з них, зайнявши по цьому показнику друге місце в ході випробувань. При цьому від двох третин шкідливих програм не залишилося взагалі ніяких слідів.

Присутність Avast не занадто сповільнює роботу ПК. Тест перевірки за запитом був виконаний за 1 хв 34 с. (це другий результат), а на сканування файлів при звертанні до них антивірус затратив 4 хв 16 с. (третє місце в рейтингу).

Інтерфейс Avast має широкі можливості. Підготовлені користувачі знайдуть тут масу параметрів, якими легко маніпулювати.

Незважаючи на складність, він досить дружній, так що не повинен віджахнути менш спокушених користувачів.

Panda Cloud Antivirus

Якщо потрібно зовсім простий антивірус, то Panda Cloud Antivirus саме те, що потрібно. Panda ефективно блокує всі дії шкідливих програм, а його надзвичайно елементарний інтерфейс ідеально підходить для тих, хто не хоче возитися із численними налаштуваннями.

Panda так само, як і AVG, нейтралізував 85,7% нових шкідливих програм. Крім того, безкоштовний варіант Panda перевершив переможця наших тестів AVG у виявленні змішаного набору шкідливих програм, виявивши 99,94% із представлених зразків.

При очищенні інфікованого диска антивірус поводилося ледве гірше. Він виявив всі активні заражені файли й блокував 87% з них, продемонструвавши результати вище за середнє рівня. Йому вдалося повністю видалити 60% шкідливого тексту, що також вище середніх показників.

У деяких тестах продуктивність Panda залишала бажати кращого. При копіюванні файлів по мережі спостерігалася істотна зміна роботи, а по швидкості сканування антивірус перебував у числі аутсайдерів. Тест перевірки за запитом був виконаний за 6 хв 10 с., а на сканування при безпосереднім звертанні до файлів пішло 7 хв 25 с.

Але якщо враховувати не тільки швидкість, але й простоту, а також гарний відсоток виявлення шкідливих програм, можна затверджувати, що Panda Cloud Antivirus стане непоганим вибором.

Avira Free Antivirus

Avira Free Antivirus працює швидко й навряд чи буде лякати попередженнями про зараження цілком легітимних файлів. Однак при виявленні шкідливих програм він показав результати нижче середнього, а інтерфейс його занадто складний.

Пакет Avira блокував 70,4% зразків нових вірусів, що трохи гірше середнього рівня, продемонстрованого розглянутими інструментами. Хоча антивірус і знайшов у нашім тестовому ПК всі активні шкідливі програми, нейтралізувати він зміг лише 80% з них, що також нижче середніх показників.

Головна панель додатка виглядає ледве краще, ніж у попередніх версіях, але інтерфейс як і раніше має надмірну технічну спрямованість. Крім того, час від часу на екрані відображається реклама.

Втім, не всі результати Avira так вже погані. У змішаному тесті додаток виявив 99,7% відомих зразків шкідливих програм. Крім того, він виявилося в числі двох безкоштовних антивірусів, які не віднесли помилково до числа шкідливих жодного незараженого файлу. При скануванні по запиті програма показала кращий результат – 1 хв 31 с. Тривалість сканування при безпосередньому звертанні до файлів, що склала 4 хв 54 с., також заслуговує поваги.

Використання Avira, безумовно, не зможе захистити ПК від всіх шкідливих програм, але в запасі в нас є й отримавше більш високі бали програмне забезпечення AVG, Avast і Panda.

PC Tools Threatfire: відмінне доповнення

Пакет PC Tools Threatfire не замінить уже наявне у вас антивірусне програмне забезпечення, оскільки не занадто ефективно очищає комп'ютер від шкідливих програм, що проникнули в нього. Але зате він відмінно справляється з роллю фільтра, відбиваючи нові погрози.

З урахуванням особливостей функціонування Threatfire так і не вдалося випробувати його в дії на відомих погрозах. Пакет не використовує традиційні

файли сигнатур, а покладається на поведінковий аналіз – шкідливі програми розпізнаються по характеру їхнього впливу на ПК.

З такою технологією Threatfire повинен демонструвати відмінні результати при виявленні нових шкідливих програм, і наші тести підтвердили це. Інструмент показав видатні показники при блокуванні реальних погроз. Повністю були зупинені 92,9% шкідливих програм. Що залишилися 7,1% удалося зупинити частково. Він лідирував у всіх тестах, що проводилися цього року.

Як вже відзначали раніше, з очищенням комп'ютера Threatfire справляється не дуже добре. Йому вдалося виявити й знешкодити лише біля половини заражених файлів на тестовому ПК. Тому краще встановлювати Threatfire на чисту, вільну від вірусів систему.

Утиліта проектувалася з урахуванням підтримки тісної взаємодії з більшістю антивірусних продуктів.

Malwarebytes Antimalware

Malwarebytes Antimalware – ще одне безкоштовне доповнення, що зміцнює вже наявні засоби безпеки. Розроблювачі пакета Malwarebytes Antimalware позиціонують його як інструмент для виявлення й видалення новітніх шкідливих програм. Для рішення цього завдання застосовується цілий ряд різних технологій.

Із блокуванням нових шкідливих програм утиліта справляється не гірше інших безкоштовних антивірусів. У наших реальних тестах продукт Malwarebytes блокував 78,6% зразків повністю (що трохи краще середнього рівня) і 14,3% частково. Додаток працює по-справжньому швидко. Із всіх безкоштовних антивірусів воно вплинуло на загальну продуктивність системи.

Разом з тим, використовувати його в якості основного антивірусного програмного забезпечення навряд чи має сенс. У боротьбі із уже відомими шкідливими програмами утиліта виявила себе не з найкращої сторони. Вона виявила всього 57,1% шкідливих програм, що з'явилися за останні чотири місяці. Вона продемонструвала найгірший результат із всіх показаних безкоштовних антивірусів.

Інтерфейс ПЗ виконаний без надмірностей, але простий і зрозумілий. Корисна функція Chameleon дозволяє функціонувати додатку, навіть коли шкідлива програма намагається блокувати його.

Розроблювачі Malwarebytes не рекомендують використовувати свій продукт на першій лінії антивірусної оборони, і вам, чесно говорячи, теж не треба цього робити. Але він буде гарним доповненням до вже існуючих засобів безпеки.

Microsoft Security Essentials

Пакет Microsoft Security Essentials, що фінішував п'ятим, викликав у нас суперечливі почуття. Сподобався інтерфейс, особливо ж удало додаток виявив себе при очищенні заражених файлів. Але от що стосується нових шкідливих програм, побачити що-небудь за рамками своїх шор він так і не зміг.

Користувальницький інтерфейс Security Essentials виконаний майже на ідеальному рівні – сама досконаліть, якщо немає часу довго возитися з налаштуваннями антивірусного ПЗ.

Додаток Microsoft упорався зі шкідливими програмами зовсім непогано: у тестах він виявив на комп'ютері всі активні заражені файли й нейтралізував більше 93% з них. В 80% випадків від шкідливих програм не залишилося ніяких слідів – кращий результат у тесті. Крім того, це другий випадок, коли не було зафіксовано жодного помилкового спрацьовування.

Security Essentials відмінно впорався з видаленням із ПК шкідливих програм, що вже бути на ньому, але настільки ж блискуче виявити себе при захисті комп'ютера від атак йому не вдалося. Антивірус блокував 71,4% нових шкідливих програм, показавши результат нижче середнього. При змішаному тестуванні додаток виявив 97% відомих шкідливих зразків. У цьому змісті воно відстає від конкурентів, адже деяким з них удалося виявити 99,9% погроз.

У цілому Security Essentials не занадто сильно знижує продуктивність системи, але при копіюванні файлів і установці додатків швидкодія комп'ютера падала нижче середнього рівня. Швидкість сканування також залишає бажати

кращого. Перевірка за запитом була виконана за 3 хв 56 с. Та й сканування при безпосереднім звертанні до файлів не дуже добре, завершившись через 6 хв 43 с.

Хоча Microsoft Security Essentials і володіє цілим рядом позитивних якостей, думаю, що краще все-таки пошукати що-небудь інше.

PC Tools AntiVirus Free

У пакета PC Tools AntiVirus Free є свої сильні сторони, але при блокуванні нових шкідливих програм він показав найгірший результат. При перевірці реальних погроз інструмент PC Tools виявив лише 57,1% від загальної кількості запропонованих йому зразків, значно відставши від конкурентів.

Рисунок 2.2 – Інтерфейс користувача PC Tools AntiVirus

Серед позитивних моментів слід зазначити розпізнавання 99,96% відомих шкідливих програм, що з'явилися за останні чотири місяці.

Крім того, це один із кращих антивірусів, що очищають ПК від «інфекції». У нашій тестовій системі він виявив всі заражені файли й успішно усунув 93% погроз. Ніяких слідів від шкідливих програм не залишилося в 73% випадків.

У цілому PC Tools AntiVirus Free мало сповільнює роботу комп'ютера. Однак конкретні результати виявилися суперечливі. На перевірку за запитом знадобилося цілих 4 хв і 51 с. А от сканування при безпосереднім звертанні до файлів завершилося вже через 2 хв 50 с. Антивірус продемонстрував другий результат серед всіх протестованих програм.

У більшості випадків з використанням PC Tools ніяких утруднень не виникло. Керуючі елементи інтерфейсу розміщені грамотно.

Єдине, що дратувало, – це поява при кожному запуску реклами із пропозицією платного програмного забезпечення компанії.

Comodo Internet Security Premium

Антивірус Comodo Internet Security Premium має зручний інтерфейс і поставляється разом з міжмережним екраном (єдиний випадок серед протестованого безкоштовного ПЗ). Але при виконанні операцій по блокуванню погроз і очищенню комп'ютера цей продукт відстає від своїх конкурентів.

Ефективність Comodo при перевірці реальних погроз виявилася на цілком прийнятному рівні. Додаток блокував 78,6% нових шкідливих програм, продемонструвавши результат ледве вище за середнє рівня. Частково було блоковано 21,4% атак. Разом з тим, антивірус Comodo слабко впорався із уже відомими шкідливими програмами. Він виявив 98,2% зразків. Це може здатися гарним результатом, гірше виявили себе лише два із протестованих продуктів.

Очищення заражених файлів можна вважати проблемою Comodo. Додаток розпізнав всі інфіковані файли, але нейтралізувати змогло лише 80% з них. Це явно уступає показникам лідерів. Та й слідів заражених файлів після очищення

системи залишилося значно більше, ніж в інших. Додатку вдалося повністю видалити тільки третину шкідливих програм.

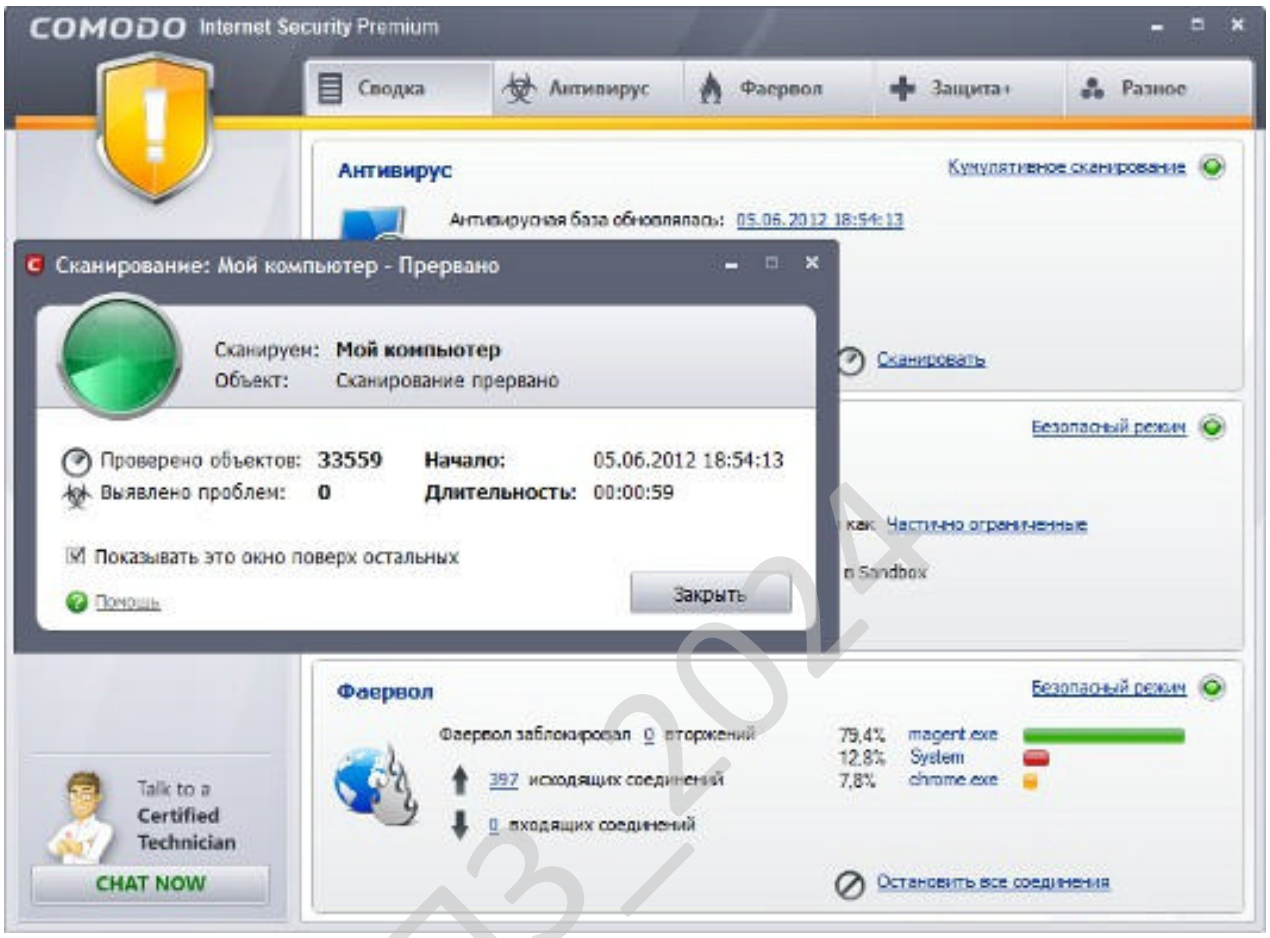

Рисунок 2.3 – Інтерфейс користувача Comodo

У більшості ситуацій вплив Comodo на продуктивність ПК відчувався досить слабко, але от операція копіювання файлів по мережі виконувалася дуже повільно. Багато часу зайняла й процедура сканування файлів. Перевірка за запитом тривала 4 хв 31 с. Це третій результат з кінця. А на сканування файлів при безпосереднім звертанні до них пішло цілих 13 хв 58 с. – майже вдвічі більше, ніж у наступного самого повільного продукту. Звичайно, наявність міжмережного екрана робить Comodo Internet Security Premium досить привабливим варіантом, але ми б все-таки рекомендували спочатку звернути увагу на інші безкоштовні продукти.

2.2 Обґрунтування вибору засобів для побудови системи кібербезпеки та мови програмування

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

Delphi 10.4 Sydney

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

Основні можливості Delphi 10.4.1:

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

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

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

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

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

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

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

− Підтримка Metal Driver GPU для macOS і iOS.

− Вбудований Fmxlinux.

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

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

− Додані оновлені драйвери для FireBird, PostgreSQL і SQLite.

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

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

RAD Studio 10.4 Короткий огляд:

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

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

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

Істотне поліпшення Delphi Code Insight

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

Delphi Custom Managed Records

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

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

Єдине керування пам'яттю

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

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

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

Розширена підтримка бібліотек C++

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

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

Win 64-відладник і збирач для C++

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

Підвищення якості й швидкодії інструментів

– Велика кількість поліпшень STL від Dinkumware.

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

– Поліпшена підтримка Cmake.

– Велика кількість виправлень для підвищення стабільності і якості.

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

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

Змінені стилі VCL для High DPI

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

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

Нові High DPI стилі й стилізація окремих VCL компонент

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

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

Поліпшена кроссплатформеність

– Додана підтримка Metal Driver GPU для macOS і iOS.

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

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

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

– Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API.

– Реалізація компонента Media Player для macOS тепер використовує Avfoundation.

Оновлений менеджер пакетів Getit

Менеджер пакетів Getit в IDE був значно вдосконалений.

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

Універсальний інсталятор для установки Online і Offline

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

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

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи кібербезпеки антивірусного захисту файлових серверів.

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

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

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

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

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

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

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

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

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

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Розглянемо деякі поширені запитання щодо безпеки файлового сервера.

Що таке безпека файлового сервера і чому це важливо?

Безпека файлового сервера означає захист файлів і даних, що зберігаються на сервері, від несанкціонованого доступу, крадіжки або пошкодження. Це важливо, оскільки сервери часто містять чутливу або конфіденційну інформацію, а злам може мати серйозні наслідки, наприклад втрату даних, фінансові наслідки або шкоду репутації.

Які поширені ризики безпеки пов’язані з файловими серверами?

Деякі поширені ризики безпеки включають зараження зловмисним програмним забезпеченням, несанкціонований доступ або використання, слабкі паролі, невиправлені вразливості програмного забезпечення, фізичну крадіжку або пошкодження та внутрішні загрози.

Як антивірусне програмне забезпечення може допомогти підвищити безпеку файлового сервера?

Антивірусне програмне забезпечення може допомогти виявити та видалити зловмисне програмне забезпечення, віруси та інші шкідливі програми, які можуть поставити під загрозу безпеку файлового сервера. Він також може забезпечити захист у реальному часі від нових загроз і може бути налаштований на регулярне сканування файлів і папок. Регулярні оновлення антивірусного програмного забезпечення можуть забезпечити його ефективність проти останніх загроз.

Які найкращі методи захисту файлового сервера?

Деякі найкращі практики включають впровадження контролю доступу та автентифікації користувачів, використання надійних паролів і обмеження

привілеїв користувачів, регулярне резервне копіювання важливих даних, шифрування конфіденційних даних, встановлення та оновлення антивірусного та іншого програмного забезпечення безпеки, моніторинг активності сервера та журналів, а також проведення регулярних аудитів безпеки та оцінки. Також важливо навчати користувачів хорошим звичкам безпеки, наприклад уникати відкриття підозрілих електронних листів або натискання невідомих посилань.

Протокол FTP (File Transport Protocol)

Щоб далі були зрозумілі деякий налаштування, розглянемо, що ж із себе представляє алгоритм роботи FTP.

Протокол FTP (File Transport Protocol), створений спеціально для передачі файлів, працює по двох портах. Клієнт приєднується до серверу (за замовчуванням на 21-й порт) і передає йому команди, а для передачі файлів сервер приєднується до клієнта, і вже по цьому каналу передаються файли. Такий режим (сервер приєднується до клієнта) називається "PORT", і його неможливо використовувати користувачам вихідним в інтернет через NAT, тому був придуманий режим, де для передачі файлів клієнт сам установлює друге з'єднання – це називається "PASV" або "пасивний FTP-протокол".

Microsoft Internet Explorer тільки починаючи з версії 5.5 навчився працювати в PASV-режимі (відповідну опцію можна включити в його параметрах). Однак режим "PORT" здаватися не став – у просунутих FTP-клієнтах (як, наприклад,C uteFTP, FTP-Voyager і навіть Total Commander) можна вказати діапазон портів для PORT-режиму, які переходить на NAT до вашої машини й ці клієнти будуть просити сервер приєднується саме на ці порти.

Є два варіанти передачі даних – Binary і ASCII. У режимі "Binary" дані передаються будь-які, але небагато повільніше; у режимі "ASCII" можуть передаватися тільки текстові файли. Режим передачі вибирається клієнтом, і звичайно автоматично встановлюється ASCII-режим для TXT-, HTML-, INI-файлів і інших явно текстових. Краще завжди й для всіх файлів використовувати режим "Binary".

FTP може мати різні розширення в плані захисту від перехоплення трафіку. Захист від перехоплення може бути реалізована двома методами, залежно від необхідного рівня безпеки.

По-перше, можна використовувати зашифровані паролі (OTP-MD5), у результаті чого по каналу зв'язку буде переданий не пароль, а його хеш, причому з додаванням часу – тому зловмисники не зможуть довідатися пароль, а за рахунок додавання часу повторне відправлення копії хешу від зловмисників не прийметься. Це не дозволить зловмисникам увійти на персональний FTP-аккаунт.

По-друге, можливе шифрування всього FTP-трафіку (FTP SSL/TLS). Це не дозволить зловмисникам одержати всі ті файли, які ви передавали. Однак відзначу, що стандартні клієнти, споконвічно присутні в Windows (Internet Explorer і ftp.exe), ніяких цих методів захисту від перехоплення не мають, тому користувачі вашого FTP-серверу повинні будуть використовувати альтернативні FTP-клієнти – CuteFTP або FTP-Voyager.

Який антивірус краще?

Який антивірус найкращий? Відповідь буде – "будь-який", якщо на вашім файловому сервері віруси не водяться, і ви не користуєтеся вирусонебезпечними джерелами інформації. Якщо ж ви аматор нових програм, іграшок, ведете активну переписку по електронній пошті й використовуєте при цьому Word або обмінюєтеся таблицями Excel, то вам все-таки варто використовувати який-небудь антивірус. Який саме – вирішуйте самі, однак є кілька позицій, по яких різні антивіруси можна зрівняти між собою.

Методика використання антивірусних програм

Стежте за тим, щоб антивірусні програми, використовувані для перевірки, були самих останніх версій. Якщо до програм поставляються апдейти, то перевірте їх на "свіжість". Звичайно вихід нових версій антивірусів анонсується, тому досить відвідати відповідні WWW/ftp/BBS.

"Національність" антивірусів у більшості випадків не має значення, оскільки на сьогоднішній день процес еміграції вірусу в інші країни й імміграції

антивірусних програм обмежується тільки швидкістю Internet, тому як віруси, так і антивіруси не визнають границь.

Якщо на файловому сервері виявлений вірус, то саме головне – не панікувати. Паніка ніколи не доводила до добра: непродумані дії можуть привести до сумних наслідків.

Якщо вірус виявлений у якомусь із нових файлів і ще не проникнув у систему, то немає причин для занепокоєння: убийте цей файл (або видалите вірус улюбленою антивірусною програмою) і спокійно працюйте далі. У випадку виявлення вірусу відразу в декількох файлах на диску або в завантажувальному секторі, то проблема стає більше складної.

Треба ще раз звернути увагу на термін "помилкове спрацьовування". Якщо в якому-небудь одному файлі, що досить довго "живе" на файловому сервері, який-небудь один антивірус виявив вірус, то це швидше за все помилкове спрацьовування. Якщо такий файл запускався кілька разів, а вірус так і не переповз в інші файли, то це вкрай дивно. Спробуйте перевірити файл іншими антивірусами. Якщо вони зберігають мовчання, відправте цей файл у лабораторію фірми-виробника антивірусу, що виявив у ньому вірус.

Якщо ж на файловому сервері дійсно знайдений вірус, то треба зробити наступне:

1. У випадку виявлення файлового вірусу, якщо файловий сервер підключений до мережі, необхідно відключити його від мережі й проінформувати системного адміністратора. Якщо вірус ще не проникнув у мережу, це захистить сервер і інші робочі станції від проникнення вірусу. Якщо ж вірус уже вразив сервер, то відключення від мережі не дозволить йому знову проникнути на файловий сервер після його лікування. Підключення до мережі можливо лише після того, як будуть вилікувані все серверу й робітники станції.

При виявленні завантажувального вірусу відключати файловий сервер від мережі не треба: віруси цього типу по мережі не поширюються (природно, крім файлово-завантажувальних вірусів).

Якщо відбулося зараження макро-вірусом замість, відключення від мережі досить на період лікування переконатися в тому, що відповідний редактор (Word/Excel) неактивний на жодному файловому сервері.

2. Якщо виявлено файловий або завантажувальний вірус, варто переконатися в тому, що вірус або нерезидентний, або резидентна частина вірусу знешкоджена: при запуску деякі (але не всі) антивіруси автоматично знешкоджують резидентні віруси в пам'яті. Видалення вірусу з пам'яті необхідно для того, щоб зупинити його поширення. При скануванні файлів антивіруси відкривають їх, багато хто з резидентних вірусів перехоплюють цю подію й заражають файли, що відкриваються. У результаті більша частина файлів виявиться зараженою, оскільки вірус не вилучений з пам'яті.

Якщо використовуваний антивірус не видаляє віруси з пам'яті, варто перезавантажити файловий сервер. Перезавантаження повинно бути "холодним" (клавіша Reset або вимикання/включення файлового серверу), тому що деякі віруси "виживають" при теплому перезавантаженні.

Крім резидентності/нерезидентності корисно ознайомитися й з іншими характеристиками вірусу: типами файлів, що заражаються вірусом,, проявами та інше.

3. За допомогою антивірусної програми потрібно відновити заражені файли й потім перевірити їхня працездатність. Перед лікуванням або одночасно з ним – створити резервні копії заражених файлів і роздрукувати або зберегти де-небудь список заражених файлів (log-файл антивірусу). Це необхідно для того, щоб відновити файли, якщо лікування виявиться неуспішним через помилку в лікуючому модулі антивірусу або через нездатність антивірусу лікувати даний вірус. У цьому випадку прийдеться вдатися до допомоги якого-небудь іншого антивірусу.

Набагато надійніше, звичайно, відновити заражені файли з backup-копії (якщо вона є), однак однаково будуть потрібні послуги антивірусу – раптом не всі копії вірусу виявляться знищені, або якщо файли в backup-копії також заражені.

Слід зазначити, що якість відновлення файлів багатьма антивірусними програмами залишає бажати кращого. Багато популярних антивірусів буває незворотньо псують файли замість їхнього лікування. Тому якщо втрата файлів небажана, то виконувати перераховані вище пункти треба в повному обсязі.

Крім перерахованих вище пунктів необхідно звертати особливу увагу на чистоту модулів, стислих утилітами типу LZEXE, PKLITE або DIET, файлів в архівах (ZIP, ARC, ICE, ARJ і т.д.) і даних у файлах, що саморозпаковуються, створеними утилітами типу ZIP2EXE. Якщо випадково впакувати файл, заражений вірусом, то виявлення й видалення такого вірусу без розпакування файлу практично неможливо. У цьому випадку типової буде ситуація, при якій всі антивірусні програми, нездатні сканувати усередині впакованих файлів, повідомлять про те, що від вірусів очищені всі диски, але через якийсь час вірус з'явиться знову.

Штами вірусу можуть проникнути й в backup-копії програмного забезпечення при відновленні цих копій. Причому архіви й backup-копії є основними постачальниками давно відомих вірусів. Вірус може роками "сидіти" у дистрибутивній копії якого-небудь програмного продукту й зненацька виявитися при установці програм на новому файловому сервері.

Ніхто не гарантує повного знищення всіх копій комп'ютерного вірусу, тому що файловий вірус може вразити не тільки виконувані файли, але й оверлейні модулі з розширеннями імені, що відрізняються від COM або EXE. Тому доцільно якийсь час після видалення вірусу постійно користуватися резидентним антивірусним сканером (не говорячи вже про те, що бажано користуватися ним постійно).

Виявлення невідомого вірусу

У цьому розділі розглядаються ситуації, з якими може зштовхнутися користувач у тому випадку, якщо він підозрює, що його файловий сервер уражений вірусом, але жодна з відомих йому антивірусних програм не дала позитивного результату. Де і як шукати вірус? Які при цьому необхідні

інструментальні засоби, якими методами варто користуватися і яким правилам випливати?

Найперше правило – не панікувати. Ні до чого гарному це не приведе. Якщо немає впевненості у власних силах – звернетеся до системного програміста, що допоможе локалізувати й видалити вірус (якщо це дійсно вірус) або знайти іншу причину "дивного" поводження файлового серверу.

Виявлення резидентного вірусу

Якщо у файловому сервері виявлені сліди діяльності вірусу, але видимих змін у файлах і системних секторах дисків не спостерігається, то цілком можливо, що файловий сервер уражений одним з "стелс"-вірусів.

Виявлення завантажувального вірусу

У завантажувальних секторах дисків розташовані, як правило, невеликі програми, призначення яких складається у визначенні розмірів і границь логічних дисків (для MBR вінчестера) або завантаженню операційної системи (для boot-сектора).

Для початку варто прочитати вміст сектора, підозрілого на наявність вірусу. Деякі завантажувальні віруси практично відразу можна виявити по наявності різних текстових рядків. Деякі віруси, що вражають boot-сектори дисків, навпаки, визначаються по відсутності рядків, які обов'язково повинні бути присутнім в boot-секторі. До таких рядків відносяться імена системних файлів (наприклад, рядок "IO SYSMSDOS SYS") і рядка повідомлень про помилки.

Однак існують віруси, які впроваджуються в завантажник без зміни його текстових рядків і з мінімальними змінами коду завантажника.

Існують також віруси, що використовують більше складні прийоми зараження, наприклад, що змінюють при інфікуванні MBR усього 3 байти Disk Partition Table, що відповідають адресі активного завантажувального сектора. Для ідентифікації такого вірусу прийдеться провести більше детальне дослідження кодів завантажувального сектора аж до повного аналізу алгоритму роботи його коду.

Виявлення файлового вірусу

Як відзначалося, віруси діляться на резидентні й нерезидентні. Резидентні віруси, що зустрічалися дотепер, відрізнялися набагато більшим підступництвом і витонченістю, чим нерезидентні. Тому для початку розглянемо найпростіший випадок – ураження файлового серверу невідомим нерезидентним вірусом. Такий вірус активізується при запуску якої-небудь зараженої програми, робить усе, що йому треба, передає керування програмі-носію й надалі (на відміну від резидентних вірусів) не буде заважати її роботі. Для виявлення такого вірусу необхідно зрівняти довжини файлів на вінчестері й у дистрибутивних копіях (згадування про важливість зберігання таких копій уже стало банальністю). Якщо це не допоможе, то треба побайтно зрівняти дистрибутивні копії з використовуваними програмами.

Можна також переглянути дамп виконуваних файлів. У деяких випадках можна відразу виявити присутність вірусу по наявності в його коді текстових рядків. Багато вірусів, наприклад, містять рядки: ".COM", "*.COM", ".EXE", "*.EXE", "*.*", "MZ", "COMMAND" і т.д. Ці рядки часто зустрічаються на початку або наприкінці заражених файлів.

Розглянуті вище методи виявлення файлових і завантажувальних вірусів підходять для більшості як резидентних, так і нерезидентних вірусів. Однак ці методи не спрацьовують, якщо вірус виконаний за технологією "стелс", що робить марним використання більшості резидентних блокувальників, утиліт порівняння файлів і читання секторів.

Виявлення макро-вірусу

Характерними проявами макро-вірусів є:

– Word: неможливість конвертування зараженого документа Word в інший формат.

– Word: заражені файли мають формат Template (шаблон), оскільки при зараженні Word-віруси конвертують файли з формату Word Document в Template.

– тільки Word 6: неможливість запису документа в інший каталог/на інший диск по команді "Save As".

– Excel/Word: в STARTUP-каталозі присутні "чужі" файли.

– Excel: наявність у Книзі (Book) зайвих і схованих Аркушів (Sheets).

Для перевірки системи на предмет наявності вірусу можна використовувати пункт меню Tools/Macro. Якщо виявлені "чужі макроси", то вони можуть належати вірусу. Однак цей метод не працює у випадку стелс-вірусів, які забороняють роботу цього пункту меню, що, у свою чергу, є достатньою підставою вважати систему зараженої.

Багато вірусів мають помилки або некоректно працюють у різних версіях Word/Excel, у результаті чого Word/Excel видають повідомлення про помилку, наприклад:

WordBasic Err = номер помилки

Якщо таке повідомлення з'являється при редагуванні нового документа або таблиці й при цьому свідомо не використовуються які-небудь користувальницькі макроси, то це також може служити ознакою зараження системи.

Сигналом про вірус є й зміни у файлах і системній конфігурації Word, Excel і Windows.

Природно, що до проявів вірусу відносяться такі очевидні факти, як поява повідомлень або діалогів з досить дивним змістом або мовою, що не збігається з мовою встановленої версії Word/Excel.

Проблема захисту від макро-вірусів

Існує кілька прийомів і убудованих в Word/Excel функцій, спрямованих на запобігання запуску вірусу. Найбільш діючою з них є захист від вірусів, убудована в Word і Excel (починаючи з версій 7.0a). Цей захист при відкритті файлу, що містить будь-який макрос, повідомляє про його присутність і пропонує заборонити цей макрос. У результаті макрос не тільки не виконується, але й не видний засобами Word/Excel.

Такий захист є досить надійним, однак абсолютно марний, якщо користувач працює з макросами (будь-якими): вона не відрізняє макроси вірусу від не-вірусу й виводить попереджуюче повідомлення при відкритті практично будь-якого файлу. Із цієї причини захист у більшості випадків виявляється відключеної, що дає можливість вірусу проникнути в систему. До того ж включення захисту від вірусів у вже зараженій системі не у всіх випадках допомагає – деякі віруси, один раз одержавши керування, при кожному запуску відключають захист від вірусів і в такий спосіб повністю блокують її.

Існують інші методи протидії вірусам, наприклад функція DisableAutoMacros, однак вона не забороняє виконання інших макросів і блокує тільки ті віруси, які для свого поширення використовують один з авто-макросів.

Запуск Word з опцією /M (або з натиснутою клавішею Shift) відключає тільки один макрос AutoExec і в такий спосіб також не може служити надійним захистом від вірусу.

Дезактивація оперативної пам'яті

Процедура дезактивації пам'яті, як і лікування заражених файлів, вимагає деяких знань про операційну систему й обов'язкове знання мови Асемблер.

При лікуванні оперативної пам'яті необхідно виявити коди вірусу й змінити їх таким чином, щоб вірус надалі не заважав роботі антивірусної програми – "відключити" підпрограми зараження й стелс. Для цього потрібен повний аналіз коду вірусу, тому що процедури зараження й стелс можуть розташовуватися в різних ділянках вірусу, дублювати один одного й одержувати керування при різних умовах.

Аналіз алгоритму вірусу

На мій погляд, найбільш зручним для зберігання й аналізу вірусу об'єктом є файл, що містить його (вірусу) тіло. Як показує практика, для аналізу файлового вірусу зручніше мати кілька заражених файлів різної, але не дуже великої, довжини. При цьому бажано мати заражені файли всіх типів (COM, EXE, SYS, BAT, NewEXE), що вражаються вірусом.

При аналізі алгоритму вірусу має бути з'ясувати:

– спосіб(и) розмноження вірусу;

– характер можливих ушкоджень, які вірус наніс інформації, що зберігається на дисках;

– метод лікування оперативної пам'яті й заражених файлів (секторів).

При аналізі файлового вірусу необхідно з'ясувати, які файли (COM, EXE, SYS) заражені вірусом, у яке місце (місця) у файлі записується код вірусу – у початок, кінець або середину файлу, у якому обсязі можливе відновлення файлу (повністю або частково), у якому місці вірус зберігає відновлювану інформацію.

При аналізі завантажувального вірусу основним завданням є з'ясування адреси (адрес) сектора, у якому вірус зберігає первісний завантажувальний сектор (якщо, звичайно, вірус зберігає його).

Для резидентного вірусу потрібно також виділити ділянка коду, що створює резидентну копію вірусу й обчислити можливі адреси крапок входу в перехоплюються вірусом переривання. Необхідно також визначити, яким образом і де в оперативній пам'яті вірус виділяє місце для своєї резидентної копії.

Існують особливі випадки, коли аналіз вірусу може виявитися дуже складним для користувача завданням, наприклад при аналізі поліморфічного вірусу. У цьому випадку краще звернутися до фахівця з аналізу кодів програм.

3.2 Розробка структурної схеми

Структурна схема системи містить у собі наступні блоки:

1. Антивірусний двигун файлового серверу.

2. Фаєрвол.

3. Поведінковий контроль.

4.Веб-захист браузерів.

5. Батьківський контроль.

6.Анти-спам.

7. Захист від уразливостей.

8. Відновлення.

9. Захист налаштувань.

10. Онлайн резервне копіювання.

Розглянемо ці структурні блоки більш докладно.

Антивірусний двигун файлового серверу

Антивірусний двигун файлового серверу – основний компонент. Основна роль движка – сканування сховища даних, він проникає в комп'ютер з метою виявлення й видалення шкідливого ПЗ. Шкідливий код може зберігатися у файлах на жорстких дисках, переносних USB накопичувачах, в оперативній пам'яті комп'ютера, мережних драйверах, завантажувальному секторі диска або надходити як частина мережного трафіку.

Фаєрвол

Основна роль фаєрвола – контролювати доступ до ПК із боку зовнішньої мережі, тобто вхідний трафік і, навпаки, контролювати доступ із ПК у мережу, тобто вихідний трафік.

Фільтрація мережного трафіку може відбуватися на декількох рівнях. Більшість фаєрволів, включених у комплекти безпеки для ПК мають набір правил принаймні для двох рівнів – нижній інтернет рівень, контрольований IP правилами й верхній рівень додатка.

Говорячи про верхній рівень, фаєрвол містить набір правил для того щоб допустити або заборонити доступ конкретного додатка в мережу. Такі терміни, як мережні правила (Network Rules), експертні правила (Expert Rules) або налаштування правил IP (IP Rule Setting) використовуються на нижньому рівні правил. На верхньому рівні ми зустрічаємося з термінами Контроль програм (Program Control) або правила додатків (Application Rules).

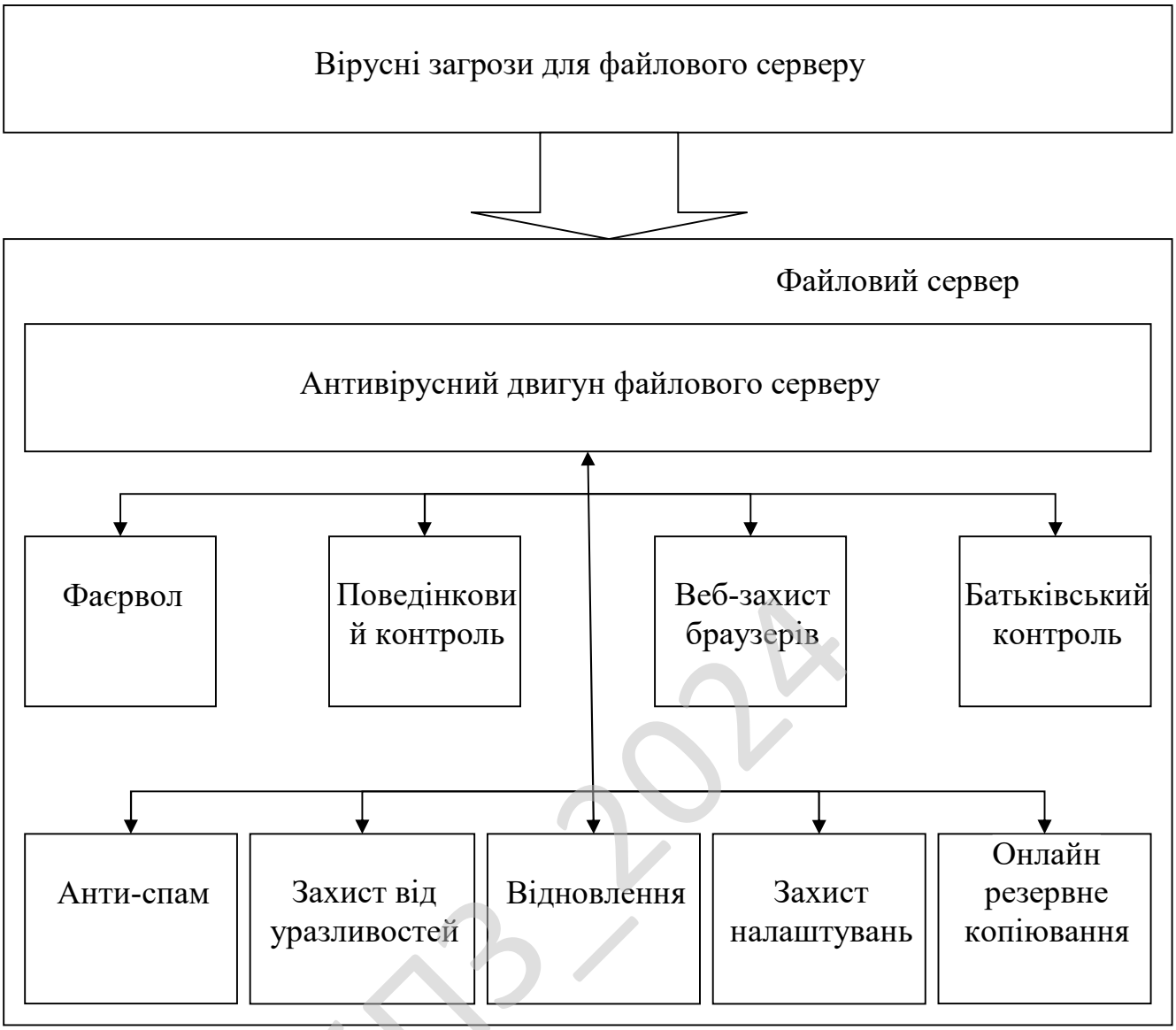

Рисунок 3.1 – Структурна схема системи

Поведінковий контроль

Компоненти поведінкового контролю (Behavior Control) здійснюють моніторинг дій всіх додатків у системі й блокують дії, які загрожують безпеці системи і її користувачів. Поведінковий аналіз містить базу даних наборів правил, які визначають, які дії повинні бути дозволені або заблоковані для кожної програми. Система захисту виконує контроль і припиняє роботу програм, які можуть виконати потенційно небезпечну дію. Якщо існує правило, що визначає конкретну ситуацію, воно використовується для того щоб або щоб дозволити, або

заблокувати дію. Якщо яка-небудь дія вирішена заблокувати, виконання програмного потоку модифікується таким чином, щоб дія не виконувалася й всі параметри додатка міняються для того щоб гарантувати безпека; якщо дія програми дозволена набором правил, його виконання відбувається без змін з боку захисту. Іноді не існує певного правила для дії програми в базі даних. У таких випадках, залежно від налаштувань компонентів поведінкового контролю, користувачеві або пропонується прийняти рішення, або дія виконується або блокується в автоматичному режимі на підставі інформації евристичного аналізу.

Поведінковий контроль не сканує файли додатків перед їхнім виконанням, отже, неможливо визначити шкідливе ПЗ до запуску додатка. Проте, ця опція антивірусних програм дозволяє ефективно блокувати небезпечне поводження й, таким чином, запобігати ушкодження й захищати систему від відомих і невідомих вірусів. Інші компоненти, наприклад Антивірусний двигун файлового серверу (Anti-virus Engine), можуть бути тісно пов'язані з поведінковим контролем і критично необхідні для його правильної роботи. Основний напрямок розвитку розробки поведінкового контролю пов'язане з можливістю визначення ступеня небезпеки конкретного додатка до його запуску, навіть якщо воно є непізнаним.

Веб-захист браузерів

Веб-браузери є самими основними цілями атак з боку шкідливого ПЗ. Вони можуть бути використані для зараження ПК через інтернет, для передачі украденої інформації із зараженого комп'ютера на серверу зловмисників. Браузери часто стають об'єктом атаки через дані платіжні системи, які можуть зберігатися в базі автозавнення форм браузерів.

Батьківський контроль (Parental Control)

Також називають: контроль доступу (Access control).

Батьківський контроль дозволяє пріоритетним користувачам (батькам) контролювати використання комп'ютера менш пріоритетною аудиторією (діти). Функція може використовуватися для обмеження доступу до облікового запису

комп'ютера. Батьківський контроль може ефективно обмежити дітям доступ до комп'ютера протягом певного часу в день, або на певну кількість годин у день. Також може бути обмежене призначення використання ПК, шляхом контролювання запуску сторонніх додатків. Обмеження за часом можуть зіставлятися з обмеженням запуску певних додатків.

Батьки можуть мати намір контролювати веб-сайти, відвідувані дітьми. Обмеження створюються за допомогою спеціальних чорних списків, однак, більшість антивірусів з функцією батьківського контролю дозволяють контролювати доступ у мережу по конкретних категоріях умісту сайтів. Контент більшості самих популярних сайтів оцінюється або безпосередньо розроблювачами антивірусного ПЗ, або аналізується на основі ключових слів видачі пошукових систем. Потім кожний веб-сайт може бути прилічений до однієї або декількох категоріям, таким як контент 18+, насильство, екстремізм, зброя, азартні ігри, наркотичні речовини, ненормативна лексика, чати, тероризм, електронна пошта, соціальні мережі, платіжні системи онлайн, віртуальні знайомства, порнографія, злом, шахрайство й т.д. Батьки можуть обмежити доступ до сайтів, що складають в окремій категорії або утримуючі ключові слова, що відносяться до категорії.

Анти-спам (Anti-spam)

Основне призначення модуля анти-спам – зменшення кількості небажаних повідомлень, що користувач одержує за допомогою електронної пошти. Анти-спам модуль у сучасних антивірусних програмах звичайно має підтримку більшості популярних поштових клієнтів, таких як Outlook, Outlook Express, Windows Mail, The Bat, Thunderbird, але не працює з іншими поштовими програмами. Веб-інтерфейс пошти звичайно також не підтримується. Існує кілька способів визначення спаму. Комплексне анти-спам рішення, як правило, сполучає кілька таких методів. Одним з найбільш загальних методів є аналіз умісту вхідних повідомлень за допомогою комплексного алгоритму анти-спам модуля й наступне присвоєння спеціальних балів кожному листу.

Якщо бали перевищують певний рівень, електронне повідомлення позначається як спам, і або відсилається у відповідну папку, або негайно віддаляється. Налаштувати рівень обмеження й дія, що потрібно почати можна в конфігурації анти-спам фільтра. Спам може бути розпізнаний за допомогою аналізу тексту або характерних частин спам-повідомлення.

Інша методика включає використання хмарних даних або думки співтовариства (іноді називають веб-запит (Web-query). Вендор антивірусного ПЗ містить значну базу широкорозповсюджених спам-повідомлень, що використовують фішинг, шахрайство й обман. Коли користувач одержує електронне повідомлення, анти-спам модуль робить відповідний запит у базу даних на предмет відповідності повідомлення зразкам спаму. Новий спам додається в базу розроблювачами антивірусного, ручним способом самим користувачем, автоматичними скриптами або за допомогою позначок у поштових клієнтах користувачами співтовариства.

У загальному випадку модулі анти-спаму працюють із білими й чорними списками адрес електронної пошти. Повідомлення, що відсилаються з надійних джерел ніколи не позначаються як спам. Електронні листи, що приходять із адрес із чорного списку завжди розцінюються як спам. Білі списки можуть формуватися з обліком користувальницької адресної книги, але кінцевий список редагується користувачем. Чорні списки обновляються із серверів розроблювачів антивірусного ПЗ.

Інший ефективний метод протистояти спаму – блокування поштових серверів, що відсилають спам. Кожному спаммеру потрібен комп'ютер для відправлення спам-повідомлень. Невірно настроєного або зламаного поштового серверу звичайно використовуються для цих цілей. Такі джерела спаму можуть бути ідентифіковані антивірусом, а їхні адреси внесені в чорний список. Існує безліч користувальницьких чорних списків спам-серверів в інтернеті, які також можуть бути використані для запобігання спам-атак. Якщо електронний лист отриманий з адреси в чорному списку, воно розцінюється як спам, незважаючи

навіть на його зміст. Все більша кількість комп'ютерів зловредно використовуються для відправлення спаму, тому щоб чорні списки були ефективними, їх потрібно постійно обновляти.

Захист від уразливостей (Vulnerability Protection)

Також називають: монітор уразливостей (Vulnerability Monitor).

Ця опція дозволяє користувачеві зробити комп'ютер недоступним для відомих видів уразливостей, які можуть бути використані шкідливими ПЗ для зараження ПК. Монітор уразливостей стежить, щоб були встановлені всі останні відновлення ОС і основних установлених додатків.

Відновлення (Updates)

Постійні відновлення життєво необхідні для комп'ютерної безпеки. Існує кілька видів відновлень.

Перший вид відновлень називається відновлення антивірусної бази (Database updates), бази даних або бази наборів правил. Ці відновлення використовуються модулями антивірусу для розпізнавання останніх видів погроз. Коли новий вірус виявлений і проаналізований розроблювачем антивірусного ПЗ, робиться зразок вірусу, що поширюється на всі антивірусні програми вендора за допомогою відновлень. До виконання відновлень антивірусної бази конкретне шкідливе ПЗ може бути не розпізнано антивірусом за результатами евристичного аналізу або поведінкового контролю.

Відновлення програмних файлів (Program updates) є дуже важливим. Будь-яке програмне забезпечення, включаючи антивіруси містить помилки. Помилки усуваються під час користування антивірусом, а програмні патчи поширюються на клієнтські системи за допомогою відновлень. Крім того, відновлення програмних файлів може додати додаткову функціональність антивірусному продукту. Іноді навіть глобальні відновлення програми встановлюються за допомогою відновлень, що залежить від ліцензійної угоди відповідного вендора антивірусного ПЗ.

Існує небагато користувальницьких налаштувань відновлень. У загальному випадку параметри відновлень включають спосіб їхнього одержання – автоматичне завантаження й установка або частково ручний спосіб. У випадку вибору автоматичних відновлень, існує можливість налаштування їхньої періодичності. Відновлення антивірусної бази повинні виконуватися регулярно, один раз у день, наприклад, у той час як відновлення програмних файлів можуть відбуватися один раз у тиждень або рідше. Для комп'ютерів, підключених до інтернету через проксі-сервера надається можливість налаштування параметрів проксі для відновлень. Як би те не було, автоматичне визначення налаштувань проксі-серверів прекрасно працює на більшості систем і, отже, користувач може не торкати ці налаштування. Деякі антивіруси дозволяють включати режим економії мережі (bandwidth saving mode), при якому тільки критичні відновлення завантажуються відразу, а інші відновлення припиняються до відключення цього режиму.

Захист налаштувань (Settings Protection)

Також називають: захист паролем (Password Protection), менеджер доступу (Access Management), користувальницький менеджер (User Management).

Призначення модуля захисту налаштувань очевидно – не допустити зміна конфігурації антивірусу шкідливим ПЗ або невідомими користувачами системи. Існує три види антивірусних програм, пов'язаних із цією опцією. Перша група антивірусів взагалі не містить даної функціональності – це, як правило, невеликі програми. Їхні налаштування незахищені від дії незареєстрованних користувачів, але їхня цілісність може бути захищена від деяких вірусів за допомогою самозахисту.

Друга група антивірусів використовує простий пароль для захисту конфігурації. Цей пароль уводиться в програму для зміни налаштувань або створення нового правила.

Третя група має комплексну систему захисту, засновану на поділі користувачів або груп користувачів і присвоєнні їм спеціальних прав і обмежень

для зміни налаштувань. У таких програмних рішеннях, адміністратори системи, як правило, мають повні права доступу до налаштувань антивірусного ПЗ, права інших користувачі повністю настроюються.

Онлайн резервне копіювання (Online Backup)

Онлайн резервне копіювання є неосновною функцією, багато антивірусів її навіть не пропонують. Основна ідея функції – забезпечити користувача незалежним, добре захищеним резервним сховищем для найбільш критичних даних. Користувач може втратити дані на ПК через проблему з апаратними засобами, зараження вірусом або ненавмисною дією. Резервна копія в мережі може бути використана для одержання щодо недавньої версії найважливіших даних.

Кінцевий вибір файлів і папок, які підлягають резервному копіюванню залишається за користувачем. Однак, система резервного копіювання може запропонувати основні папки, що містять важливу інформацію (наприклад директорія «Мої документи») для копіювання. Крім того, деякі інструменти резервного копіювання дозволяють створювати копії ключів системного реєстру.

Резервні копії зберігаються на серверах розроблювачів антивірусного ПЗ. Стандартний розмір резервні копії звичайно встановлюється порядку декількох гігабайт. Процес резервування повинен бути регулярним для максимальної ефективності. З іншого боку, інтенсивне використання жорсткого диска й мережі може принести незручність користувачеві, тому резервне копіювання повинне виконуватися в режимі очікування комп'ютера.

Моніторинг продуктивності (Performance monitoring)

Моніторинг продуктивності є не дуже розповсюдженою функцією, однак деякі антивіруси містять її для виявлення нестандартного поводження, викликуваного шкідливим ПЗ або іншими проблемами. Існує безліч параметрів у системі, пов'язаних зі швидкодією системи й використанням системних ресурсів, за яких можна встановити спостереження.

Багато процесів у системі мають типову кількість ланцюжків процесів, дескрипторів, певне споживання пам'яті, використання центрального процесора,

жорсткого диска – одним словом параметри які практично не змінюються із часом. Можуть бути звичайно й піки споживання ресурсів, однак у довгостроковій перспективі існує можливість визначення незвичайного поводження процесу. Якщо відбувається підозріла подія монітор продуктивності виводить тривожне повідомлення. Користувач сповіщається про ситуацію, що відбувається, також може відбуватися автоматична дія, наприклад антивірусне сканування підозрілих об'єктів.

Користувач може налаштувати обмеження цих дій, щоб зберегти стабільність операційної системи, або встановити додаткові дії, виконувані антивірусом. Монітор продуктивності, як правило, спочатку працює в режимі навчання для збору статистичної інформації про нормальне поводження системі й лише пізніше здатний відрізняти невідповідності.

Налаштування системи (Tune-up)

Також називають: очищення системи (PC Clean-up).

Налаштування системи або очищення системи є назвою набору додаткових утиліт, які розроблювачі антивірусів пропонують як бонус до основної функціональності. Утиліти налаштування системи застосовуються для поліпшення швидкодії комп'ютера за допомогою різних прийомів. Чим більше використовується система, тим більше в ній накопичується файлів, ключів реєстру й інших об'єктів. У системі й додатках створюється безліч списків, наприклад списки відкритих документів. Деякі списки не обмежуються по розмірі й згодом стають усе більше й більше. У такий же спосіб збільшується кількість усіляких тимчасових файлів, файлів кеша. Трапляється також, що при видаленні програми залишаються деякі файли й ключі реєстру, які вона використовувала. Іншою проблемою, що знижує швидкодія системи, є фрагментація жорсткого диска.

Інструментарій функції налаштування системи справляється зі знаходженням і видаленням непотрібних об'єктів, дублів, дефрагментацією диска, очищенням інтернет історії й кеша відомих додатків. Таким чином,

швидкодія комп'ютера збільшується. Істотні поліпшення в продуктивності проявляються, тільки якщо очищення системи жодного разу не виконувалася. В інших випадках, ефект від застосування очищення може бути непомітний.

Звіти й журнали реєстрації подій (Reports and logs)

Антивірусні програми мають функцію, ведення журналу реєстрації подій, що містить деталізовану інформацію про всі події в системі й діях компонентів антивірусу. Такі журнали корисні, коли відбувається інцидент із безпекою комп'ютера. Користувач має можливість переглянути, що саме трапилася і яка дія пішла. Логи фаєрвола можуть бути використані для виявлення заражених машин у локальній мережі або мережі Інтернет і наступне додавання нових правил для фаєрвола.

Журнали компонента поведінкового контролю можуть привернути увагу користувача до потенційно небезпечних додатків у системі, про які користувач не догадувався. Як би те не було, для правильного трактування балки необхідно чітке розуміння роботи кожного компонента, що веде балку. Звіти є більше наочними для користувача, чім логи, і не жадають від користувача додаткових знань для розуміння. Звіти накопичують всі дії антивірусу за певний період без надання детальної інформації.

Ведення журналу конче потрібно для фіксації помилок ПЗ.

3.3 Розробка функціональної схеми

Функціональна схема системи містить у собі наступні блоки.

1. Антивірусний двигун файлового серверу:

– Методи визначення.

– Типи сканування й налаштування.

– Спеціалізовані сканери.

– Додаткові зв'язані опції.

2. Фаєрвол:

– Розподіл мереж на достовірні й ні.

– Виявлення й запобігання вторгнення.

– Чорний список IP.

– Блокування всього трафіку.

– Контроль програм.

3. Поведінковий контроль:

– Режими роботи.

– Пісочниця.

– Потенційно небезпечні дії й процедури.

– Керування компонентами.

– Системний щит.

– Захист переносних мультимедійних пристроїв.

– Самозахист.

4.Веб-захист браузерів:

– Контроль плагінів.

– Фільтри URL-адрес, блокування реклами.

– Динамічний зміст.

– Куки.

– Віртуалізація браузера.

– Браузерний і пошуковий помічник, анти-фішинг.

– Сканування веб-змісту.

– Захист конфіденційної інформації.

5. Батьківський контроль.

6.Анти-спам.

7. Захист від уразливостей.

8. Відновлення.

9. Захист налаштувань.

10. Онлайн резервне копіювання.

Рисунок 3.2 – Функціональна схема системи

Антивірусний двигун файлового серверу (Anti-virus Engine)

Також називають: антивірусний захист реального часу, захист реального часу, моніторинг файлів, захист від шкідливого ПЗ.

Методи визначення

Антивірусний двигун файлового серверу використовує велику кількість методів для виявлення шкідливого ПЗ. Антивірусні програми містять у собі велику базу зразків вірусів, які необхідно виявити під час сканування. Кожний зразок може або визначати унікальний шкідливий код або, що є більше розповсюдженим, описувати ціле сімейство вірусів. Основна особливість визначення вірусів шляхом порівняння зі зразками в тому, що антивірусна програма може визначити тільки добре відомий вірус, у той час як нові погрози можуть бути не виявлені.

Метод евристичного аналізу (heuristic-based detection) служить для виявлення навіть тих вірусів, для яких не існує зразків у базі антивірусної програми. Існує безліч різних методів евристичного аналізу. Основний принцип – ідентифікувати програмний код, що є вкрай небажаним для безпечних програмних продуктів. Як би те не було, цей метод неточний і може викликати безліч фіктивних тривог. Гарний евристичний аналіз відмінно збалансований і викликає мінімальну кількість фіктивних тривог при великій частці виявлення шкідливого ПЗ. Чутливість евристики може бути настроєна.

Віртуалізація (створення віртуального середовища, Virtualization) або пісочниця (sandboxing) є більше дієвими методами визначення погроз. Певний час зразки коду виконуються у віртуальній машині або іншому безпечному середовищі, звідки сканируємі зразки не можуть вибратися й нашкодити операційній системі. Поводження досліджуваного зразка в пісочниці перебуває під спостереженням і аналізується. Цей метод є зручним у тому випадку, коли шкідливе ПЗ запаковано невідомим алгоритмом (це звичайний спосіб виявитися невразливим для системи виявлення для вірусу), і воно не може бути розпаковано антивірусною системою. Усередині віртуального середовища вірус розпаковує

сам себе, начебто він виконується на реальній системі й Антивірусний двигун файлового серверу може просканувати розпакований код і дані.

Одне з новітніх досягнень антивірусного інструментарію – **хмарне сканування** (scanning in the cloud). Цей метод заснований на тому, що ПК обмежені у своїх обчислювальних здатностях, у те час як антивірусні вендори мають можливість створення великих систем з величезною продуктивністю. Комп'ютерна потужність необхідна для виконання складного евристичного аналізу, а також аналізу з використанням віртуальних машин. Серверу вендорів можуть працювати з набагато більше об'ємними базами даних зразків вірусів у порівнянні із ПК у режимі реального часу. При виконанні хмарного сканування єдиною вимогою є наявність швидкого й стабільного інтернет підключення. Коли клієнтській машині необхідно відсканувати файл, цей файл відправляється на сервер вендора за допомогою мережного з'єднання й очікується відповідь. Тим часом, клієнтська машина може виконувати своє власне сканування.

Типи сканування й налаштування

З погляду користувача, існує кілька типів антивірусного сканування, Які залежать від подій, які викликали запуск процесу сканування:

– **Сканування при доступу** (On access scan) – сканування, що відбувається, коли ресурс стає доступний. Наприклад, коли файл копіюється на жорсткий диск або коли запускається виконується файл, що (запуск процесу сканування в цьому випадку іноді називається сканування при запуску). Тільки ресурс, до якого з'являється доступ, піддається скануванню в цьому випадку.

– **Сканування на вимогу** (On demand scan) провокується кінцевим користувачем – наприклад, коли користувач викликає сканування відповідною командою меню в провіднику Windows. Таке сканування також називається ручним. Тільки обрані папки й файли скануються при цьому способі.

– **Сканування за розкладом** (Scheduled scan) є звичайно повторюваною дією, що забезпечує постійну перевірку системи на предмет шкідливого ПЗ.

Користувач може налаштовувати час і частоту сканування. Це сканування звичайно застосовується для повного сканування системи.

– **Сканування при завантаженні** (Startup scan) – сканування, ініціалізуємо антивірусною програмою при запуску ОС. Це сканування відбувається швидко й торкається папки автозавантаження, що запускаються процеси, системну пам'ять, системні служби й завантажувальний сектор.

Більшість продуктів дозволяє користувачам налаштовувати кожний вид сканування роздільно. Нижче зібрані одні із самих основних параметрів антивірусного сканування:

– Розширення файлів для сканування – сканувати всі файли або тільки виконувані (.exe, .dll, .vbs, .cmd і інші.);

– Максимальний розмір файлів – файли понад цей параметр не піддаються скануванню;

– Сканування файлів в архівах – чи сканувати файли в архівах, таких як .zip, .rar, .7z і інші;

– Використання евристичного аналізу – налаштування використання евристики й, опціонально, налаштування чутливості;

– Типи програм, про які повідомити тривогою – існує безліч програм які можуть бути неточно визначені як шкідливі. Звичайно вендори використовуються такі терміни, як Потенційно небажане ПЗ або програма з деяким ризиком погрози;

– Типи носіїв для сканування – чи сканувати файли на мережних сховищах або переносних пристроях зберігання даних;

– Дія, яку потрібно почати, коли погроза виявлена – спроба вилікувати зразок якщо можливо, видалення зразка, приміщення в карантин (спеціальна папка, з якої шкідливий код не може виконуватися, а може бути відправлений для подальшого дослідження безпосередньо вендору), заблокувати доступ або запитати про дію в користувача.

Безліч цих параметрів можуть впливати на швидкість сканування. Набір автоматичних правил сканування для швидкого, але в теж час ефективного сканування називається Інтелектуальне сканування (Smart Scan) або Швидке сканування (Quick Scan). У противному випадку, сканування називається повним (Full Scan) або глибоким (Deep Scan). Ми також можемо зустріти сканування переносних пристроїв, що застосовується для перевірки оптичних дисків, USB накопичувачів, флеш карт і схожих пристроїв. Користувальницьке сканування (Custom Scan) також доступно і є повністю кінцевим користувачем, що налаштовується.

Спеціалізовані сканери

Руткіт-сканування (або антируткіт-сканування) це опція, що пропонують деякі антивірусні вендори в своїх продуктах, тому що руткіти стали надзвичайно розповсюдженими за останнє десятиліття. Руткіт – особливий тип шкідливого ПЗ, що використовує хитрі прийоми для того, щоб залишатися невидимим для користувача й основних методів детектування вірусу. Він застосовує внутрішні механізми ОС для того щоб зробити себе недосяжним. Боротьба з руткітами жадає від розроблювачів антивірусного ПЗ створення особливих способів виявлення. Руткіт-сканування намагається знайти розбіжності в роботі ОС, які можуть бути доказом наявності руткіта в системі. Деякі реалізації перевірок на наявність руткітов ґрунтуються на постійному моніторингу системі, у той час як інші реалізації антируткітного інструментарію можуть викликатися на вимогу.

Сканування файлів Microsoft Office (або сканування на предмет макровірусів) – опція, що захищає користувача від шкідливого коду усередині офісних документів. Внутрішні принципи сканування схожі із загальними методами сканування, вони просто спеціалізуються на пошуках вірусу усередині макросів. Опція сканування може бути представлена як плагін для Microsoft Office.

Додаткові зв'язані опції

Антивірусний двигун файлового серверу звичайно тісно пов'язаний з іншими компонентами пакета безпеки. Деякі продукти представляють додаткові функції, як частина антивірусного движка, інші відображають їх роздільно. Веб контроль – опція, що є типовим представником другої групи. Ми обговоримо цю опцію окремо.

Фаєрвол (Firewall)

Також називають: персональний фаєрвол, міжмережний екран, розширений брандмауер, двосторонній міжмережний екран.

Мережі

Безліч сучасних продуктів дозволяють користувачеві налаштовувати рівень довіри до всіх мереж, підключених до комп'ютера. Навіть якщо існує тільки одне фізичне підключення, ПК може бути підключений до декількох мереж – наприклад, коли ПК підключений до локальної мережі, що має шлюзи для виходу в інтернет. Антивірусний комплекс буде роздільно управляти локальним і інтернет трафіком. Кожна зі знайдених мереж може бути або довіреною, або недовіреною і різні системні сервіси, такі як загальний доступ для файлів або принтерів можуть бути дозволені або заборонені. За замовчуванням, тільки комп'ютери з довірених мереж (trusted networks) можуть мати доступ до захищаємого комп'ютера. Підключення, реєструємі з недовірених мереж (untrusted networks) звичайно блокуються, якщо відповідна опція не дозволяє доступ. От чому інтернет з'єднання звичайно позначається як недовірені. Як би те не було, деякі продукти не розрізняють мережі в рамках одного користувальницького інтерфейсу й налаштування довірених/недовірених мереж можуть бути роздільно зазначені для кожного інтерфейсу. Термін Мережна зона (Network Zone) або просто зона (Zone) звичайно використовується замість логічної мережі.

Для недовірених мереж існує можливість налаштувати стелс-режим. Цей режим дозволяє змінити поводження системи, начебто її адреса не доступна для

мережі. Ця міра здатна ввести в оману хакерів, які спочатку знаходять об'єкти для атаки. Поводження системи за замовчуванням передбачає відповідь на всі повідомлення, навіть відправлені із закритих портів. Стелс-режим (також відомий як стелс-режим порту (stealth ports) запобігає виявлення ПК під час сканування портів.

Виявлення й запобігання вторгнення(Intrusion Detection/Prevention)

Також називають: Виявлення атаки, Система виявлення вторгнення, IP блокування, шкідливі порти.

Хоча всі перераховані вище терміни не є еквівалентними, вони відносяться до набору властивостей, які здатні запобігти або виявити спеціальні види атак з віддалених комп'ютерів. Вони включають такі опції, як виявлення порту сканування, виявлення підмінного IP, блокування доступу до добре відомих портів шкідливого ПЗ, використовуваних програмами вилученого адміністрування, троянським коням, клієнтами ботнета. Деякі терміни включають механізми для захисту від ARP атак (атак з підмінною адресою протоколу розширення) – ця опція може називатися захист від APR, захист від кеша ARP і т.д. Основна здатність цього виду захисту – автоматичне блокування атакуючої машини. Це опція може бути прямо пов'язана з нижченаведеною функцією.

Чорний список IP (IP Blacklist)

Використання цієї простої опції полягає в змісті в антивірусному продукті бази мережних адрес, з якими комп’ютер, що захищається, не повинен зв'язуватися. Ця база може поповнюватися як самим користувачем, при виявленні вірусів, так і автоматично обновлятися з великого списку небезпечних систем і мереж антивірусного вендора.

Блокування всього трафіку (Block All Traffic)

У випадку раптового зараження системи, деякі антивірусні рішення пропонують «нажати на кнопку екстреного гальмування», тобто заблокувати весь вхідний і вихідний трафік. Ця опція може представлятися як більша червона кнопка, або як частина налаштувань політики безпеки фаєрвола або за допомогою іконки в системному меню. Передбачається, що ця функція використовується,

коли користувач довідається про те що ПК заражений й хоче запобігти небажаному використанню комп'ютера шкідливим ПЗ: крадіжку персональних даних і завантаження додаткових вірусів через інтернет. Блокування мережного трафіку може бути сполучене із завершенням всіх невідомих системних процесів. Ця опція повинна бути використана з обережністю.

Контроль програм (Program Control)

Також називають: контроль додатків, інспектор додатків.

Фільтрація мережного трафіку на програмному рівні дозволяє програмам безпеки роздільно контролювати доступ у мережу кожного додатка на ПК. Антивірусний продукт містить базу даних властивостей додатків, що чи визначає доступна мережа для чи програми ні. Ці властивості розрізняються між клієнтськими програмами, які ініціалізують підключення з локальної машини на віддалені сервери (вихідний напрямок) і серверними програмами, які сканують мережний порт і приймають з'єднання з віддалених комп'ютерів (вхідний напрямок). Сучасні антивірусні рішення дозволяють користувачеві визначати детальні правила для кожного конкретного додатка.

Загалом, поводження контролю програм залежить від політики безпеки (Firewall Policy), обраної у фаєрволі й може включати наступні режими поводження:

– **Тихий режим** (автоматичний режим) працює без втручання користувача. Всі рішення приймаються автоматично з використанням бази даних антивірусного продукту. У випадку якщо не існує явного правила для програми, що хоче одержати доступ у мережу, цей доступ може бути або завжди дозволений (режим повного дозволу – Allow All mode), або завжди заблокований (режим повного блокування – Block All mode), або спеціальний евристичний аналіз використовується для визначення подальшої дії. Алгоритм вироблення рішення може бути дуже складним і може залежати від додаткових умов, таких як рекомендації мережного співтовариства. Як би те не було, деякі продукти використовують терміни: режим повного дозволу/блокування в обхід існуючої в

базі даних наборам правил і просто дозволяють, або блокують доступ будь-якому додатку в системі.

– **Користувальницький режим** (Режим. що налаштовується, – Advanced mode, Custom mode) призначається для просунутих користувачів, які хочуть контролювати кожну дію. У цьому режимі продукт автоматично справляється тільки з тими ситуаціями, для яких існують виняткові правила в базі даних. У випадку будь-яких інших дій користувачеві пропонується прийняти рішення. Деякі антивірусні рішення пропонують визначити політикові поводження, коли неможливо запитати користувача – наприклад при завантаженні комп'ютера, завершенні роботи, коли графічний інтерфейс програми недоступний або під час особливих умов – запуску гри у весь екран, коли користувач не хоче відволікатися (іноді називається Ігровий режим – Gaming mode). Звичайно всього дві опції доступні в цих випадках: режим повного дозволу й режим повного блокування.

– **Нормальний режим** (безпечний режим – Normal mode, Safe mode) дозволяє антивірусному продукту самому справлятися з більшістю ситуацій. Навіть коли не існує явних правил у базі даних, дія програми дозволена, якщо програма вважається безпечною. Аналогічно автоматичному режиму рішення може прийматися на підставі евристичного аналізу. У випадку, коли програма безпеки не може визначити чи безпечно чи додаток ні, вона виводить оповіщення, як у користувальницькому режимі.

– **Режим навчання** (режим тренування, режим установки – Learning mode, Training mode, Installation mode) в основному використовується відразу після установки антивірусного продукту або у випадках, коли користувач установлює нове ПЗ на комп'ютер. У цьому режимі антивірусний продукт дозволяє всі дії, для яких немає записів у базі даних наборів правив і додає нові правила, які будуть дозволяти відповідні дії в майбутньому після зміна режиму безпеки. Використання режиму навчання дозволяє знизити кількість тривожних оповіщень після установки нового ПЗ.

Контроль програм звичайно містить налаштування, які можуть допомогти продукту вирішити спірні ситуації, незалежно від включеного режиму роботи. Ця особливість відома як автоматичний набір правил (Automatic rule creation). Типова опція в цьому випадку дозволяє будь-які дії додаткам із цифровим підписом від довірених вендорів, навіть якщо немає відповідного запису в базі даних. Ця опція може бути розширена іншою функцією, що дозволяє робити будь-які дії додаткам без цифрового підпису, але знайомих антивірусному продукту. Контроль програм звичайно тісно пов'язаний з іншими функціями, які ми освітимо пізніше, особливо опція поведінкового контролю.

Поведінковий контроль (Behavior Control)

Також називають: Проактивная захист (Proactive Protection, Proactive Defense), активний вірусний контроль (Active Virus Control), захисний екран від вторгнень (Intrusion Guard), основна система запобігання вторгнень (HIPS, Host-based Intrusion Prevention System), поведінковий екран (Behavioral Shield), превентивний захист.

Режими роботи (Operation modes)

Аналогічно політикам безпеки фаєрвола (Firewall Policy), які визначають прийняття рішень за допомогою компонента контролю програм (Program Control), функція поведінкового контролю може мати кілька режимів роботи. Більшість антивірусних програм не надають окремих налаштувань для поведінкового контролю, у таких рішеннях ці налаштування єдині для фаєрвола (Firewall) і модуля поведінкового контролю. Деякі антивірусні рішення дозволяють конфігурувати ці дві функції роздільно, однак доступні режими роботи будуються на тих же принципах, що й політика безпеки фаєрвола:

– **режим навчання** (Learning mode) – поведінковий контроль в автоматичному режимі створює нові набори правил для дій додатків;

– **інтерактивний режим** (Interactive mode) – при спірних ситуаціях з'являється оповіщення й користувачеві пропонується прийняти рішення;

– **тихий режим** (автоматичний режим, Silent mode) – всі дії опції відбуваються в автоматичному режимі.

Деякі антивіруси дозволяють установити ступінь захисту для поведінкового контролю. Це налаштування визначає, які дії додатків вважаються потенційно небезпечними. На нижчих рівнях захисту додаткам дозволяється вільно виконувати практично всі можливі дії, за винятком самих небезпечних. На вищих рівнях, захист є більш твердим, програми перебувають під ретельним спостереженням. Іноді, навіть абсолютно безпечні дії додатків можуть блокуватися в таких випадках.

У деяких випадках є можливість налаштування конкретних реакцій на кожне потенційно небезпечну дію. Дозволити (Allow), заборонити (Block) або запитати (Prompt) – основні доступні опції, які означають, що конкретна дія може бути дозволено, заблоковане або потрібне рішення користувача.

Налаштування поведінкового контролю за замовчуванням мають на увазі використання переважно автоматичних дій і меншого втручання користувача. У таких випадках режим безпеки часто називається «оптимальний». Це означає, що більшість розширених функцій поведінкового контролю відключені й тільки основні способи виявлення шкідливого ПЗ активні. Якщо користувач хоче убезпечити себе від самих складних видів атак потрібно активувати відповідну опцію. Її назва може відрізнятися в різних вендорів, вона може називатися розширений моніторинг подій (Advanced events monitoring), «анти-витік» (Anti-leak). Звичайно для цієї технології використовуються унікальні назви під зареєстрованими торговельними марками.

Пісочниця, ізольована програмне середовище (Sandbox)

Під час роботи автоматичних режимів антивірусна програма може дозволити потенційно небезпечну дію (опція «Дозволити все» (Alow All/Alow Most) активована) або заблокувати абсолютно безпечні дії програми (опція «Заблокувати все» (Block All/Block Most) включена). Робота автоматичних режимів неідеальна, а використання інтерактивного режиму припускає наявності

в користувача певної кваліфікації для прийняття рішень. Крім того, велика кількість питань в інтерактивному режимі може дратувати користувача. У цьому випадку альтернативою може послужити використання ізольованого програмного середовища (sandbox).

Антивірусні продукти, які містять у своєму інструментарії пісочницю, обробляють всі невідомі або підозрілі програми спеціальним методом, які гарантують, що вони не принесуть шкоди системі. Створюється спеціальне середовище, називана пісочницею, що виглядає для як справжня система, що запускаються усередині додатків. Програми можуть вільно управляти об'єктами тільки в пісочниці, їхні дії недоступні для реальної системи (наприклад, зміна записів системного реєстру). Ізольоване програмне середовище гарантує, що небезпечні дії не нашкодять ОС, що запускаємий в ній додаток не може визначити, де саме він виконується. Якщо привести приклад із внесенням змін до реєстру, то додаток намагається прочитати значення запису після зроблених змін, у цей час пісочниця повертає змінені значення, незважаючи на те, що в дійсності системний реєстр виявляється не торкнуть. Існує кілька причин, чому не можна створити ідеальну пісочницю й чому деякі критичні дії постійно блокуються в безпечному середовищі. Ступінь ефективності ізольованого програмного середовища визначається можливістю її розпізнання з боку шкідливого ПЗ. Чим пісочниця менш помітна додатку, що запускається в ній, тим краще.

Надійні програми завжди запускаються поза ізольованим програмним середовищем, що дозволяє їм виконувати будь-які необхідні для нормальної роботи операції. Коли на комп'ютер установлюється нове невідоме ПЗ і ізольоване програмне середовище забороняє які-небудь дії додатку, користувач може додати цей додаток у список виключень. Деякі антивірусні програми мають у своєму арсеналі пісочницю як окрему функцію поведінкового аналізу. Ці продукти дозволяють, відключивши ізольоване середовище, як і раніше контролювати дія програм. Інші антивірусні рішення вбудовують пісочницю в компоненти поведінкового контролю. Також існують пакети безпеки, які

дозволяють налаштовувати, у яких випадках дії додатків повинні бути автоматично заблоковані, а в яких повинне прийматися рішення на підставі поточних налаштувань політики безпеки.

Потенційно небезпечні дії й процедури (Potentially Dangerous Actions and Techniques)

Потенційно небезпечні дії, що розрізняються сучасними антивірусними рішеннями можуть бути розділені на кілька груп.

Сесія динамічного обміну даними (DDE communication) – DDE є міжпроцесорним методом зв'язку, що дозволяє одночасно запускати дві або кілька програм. Серверний додаток, що використовує DDE може одержувати дані від клієнтського додатка й відповідати йому. Деякі додатки, наприклад Internet Explorer, дозволяють іншим додаткам здійснювати контроль, використовуючи команди динамічного обміну. Ця особливість може використовуватися шкідливим ПЗ для маскування небезпечних дій під достовірні джерела.

Контроль доступу об'єктної моделі програмних компонентів (COM Access Control), контроль автоматизації протоколу OLE (OLE Automation Control) – технологія автоматизації OLE заміняє DDE. Це більше розширений механізм міжпроцесорної взаємодії, заснована на об'єктній моделі програмних компонентів. Безліч важливих системних служб забезпечують інтерфейси для додатків за допомогою технологій COM/OLE. Коли інтерфейс використовується вірусом, складається враження, що ми маємо справу з довіреною службою, а не потенційно небезпечною.

Клієнтські служби виклику віддалених процедур і системи динамічних доменних імен, запит прикладного програмного інтерфейсу системи динамічних доменних імен (DNS/RPC Client Services, DNS API Request) – деякі системні служби, такі як клієнт DNS доступні за допомогою технологій, називаних виклик віддалених процедур, виклик локальних процедур або розширений виклик локальних процедур. Ці процедури використовуються для міжпроцесорної взаємодії. Також як і вищезгадані технології, ці служби

можуть бути атаковані шкідливими ПЗ. Моніторинг зв'язаних взаємодій може запобігти зловмисному користуванню цими службами.

Контроль програмних вікон, контроль повідомлень Windows (Application Window Control, Windows Messages) – віконні повідомлення є іншим механізмом міжпроцесорної взаємодії, а також одним з найбільш використовуваних користувальницьких графічних інтерфейсів додатків. Вони можуть часто піддаватися зловмисному використанню шкідливим ПЗ. Використовуючи віконні повідомлення, можливо, імітувати основні дії користувача, наприклад клич кнопкою миші. Поки додаток має графічний інтерфейс, заснований на технології віконних повідомлень, воно може бути атаковано шкідливим ПЗ за допомогою цього методу.

Впровадження коду, впровадження процесу в системну пам'ять, міжпроцесорний доступ до пам'яті (Code Injection, Process Memory Injection, Interprocess Memory Accesses) – впровадження коду в інший процес, запущений у системі є простим методом виконання шкідливого коду під маскою довіреного процесу. Вірус може бути ознайомлений з обмеженнями поведінкового контролю й для обходу захисту може впроваджувати код у надійний процес, щоб мати можливість зробити шкідливі дії. Захист довірених процесів від впровадження коду є самою головною в поведінковому аналізі сучасних антивірусних продуктів.

Впровадження бібліотек DLL (DLL Injection, Binary Planting) – впровадження бібліотеки DLL схоже із впровадженням шкідливого коду. Результат успішної атаки ідентичний – виконання шкідливого коду за допомогою довіреного додатка. Розходження в тому, що у випадку впровадження DLL, цілий модуль завантажується в процес, що піддається атаці,, у той час як впровадження коду має на увазі як правило, включення невеликої частини коду. Впровадження бібліотек є простим прийомом для розроблювачів вірусів, однак ця методика легко визначається антивірусними програмами.

Запуск додатків з підтримкою мережного обміну даними, запуск процесу, батьківське керування процесом (Network-enabled Application Launch, Process Launching, Parent Process Control) – в ОС Windows батьківський процес може контролювати дочірні процеси або за допомогою завдання певних команд, або використовуючи методи пов'язані із внутрішньою функціональністю процесу. Ця особливість представляє ще один метод атаки довіреного процесу шкідливим ПЗ. Антивірусні програми здійснюють моніторинг ланцюжка батьківських процесів: або всіх запущених у системі, або тільки довірених.

Завершення процесу (Process Termination) – завершення процесу й схожі види атак (завершення потоку, спроби критичного завершення процесу або потоку) припускають часткове ушкодження або повне відключення антивірусного захисту. Мети атаки в цьому випадку – процеси антивірусу. Фактичний результат успішної атаки залежить від реалізації конкретного антивірусного продукту. Атака може привести до нестабільності, зависанням, критичним помилкам або відключенню деяких функцій безпеки. Деякі антивіруси можуть розпізнавати ушкодження своїх компонентів і блокують ПК для запобігання подальших шкідливих дій.

Низькорівневий доступ до мережі, прямий доступ до мережі (Low-level Network Access, Direct Network Access) – більшість антивірусів здатні відмінно справлятися з контролем основного мережного трафіку, такого як веб-серфінг, повідомлення e-mail, але з'являються проблеми, коли справа стосується протоколів спеціального призначення. Нерідкі випадок, коли антивіруси дозволяють взаємодія з веб-сайтами (при використанні гіпертекстового протоколу передачі – HTTP) тільки довіреним джерелам, у той час як передача даних за допомогою протоколу керування мережними повідомленнями (ICMP) відбувається безконтрольно в автоматичному режимі. Таким чином, шкідливі програми, що використовують альтернативні методи передачі даних менш уразливі для сучасних антивірусних рішень.

Прямий доступ до диска (Direct Disk Access) – основний спосіб доступу до даних на жорсткому диску включає системні функції, які працюють із файлами й директоріями. Ранні версії Windows дозволяють додаткам прямо звертатися до диска й даних на ньому. Такий метод доступу до даних на диску дозволяє обходити основні способи захисту директорій. На ОС Windows Vista і більше пізніх ОС Windows, ця процедура обмежена й менш уразлива для шкідливих атак.

Доступ до оперативної пам'яті, прямий доступ до пам'яті (Physical Memory Access, Direct Memory Access) – кожний працюючий процес у системі має свою власну пам'ять, недоступну іншим додаткам за замовчуванням. У випадках, коли потрібен віддалений доступ до пам'яті, система робить це можливим за допомогою спеціальних функцій. У той же час антивірусна система здійснює контроль даного вила доступу. Ядро ОС також має власну пам'ять, недоступну іншим додаткам. Як би те не було, у старих ОС Windows була можливість доступу до об'єкта, що торкатися всієї пам'яті, включаючи область системного ядра. Це дозволяло шкідливому ПЗ обходити основні механізми доступу до пам'яті. В Windows Vista і більше пізніх системах, дана опція заборонена.

Установка драйверів пристроїв, ініціалізація драйвера (Device Driver Installation, Driver Load) – Додатка, що працюють в ОС Windows мають деякі обмеження, що особливо стосуються використання ресурсів апаратних засобів, таких як оперативна пам'ять, жорсткий диск, пристрої уведення й виводу й т.д. Коли додаток прагне використовувати апаратний засіб, воно звертається до системного ядра, що може або дозволити, або заборонити конкретну дію. Цей механізм відмінно працює із програмним кодом, що працює в так званому користувальницькому режимі. Код системного ядра у свою чергу працює в так званому режимі ядра, що дозволяє будь-який доступ до апаратних засобів без обмежень. Код системного ядра може обходити всі види захисту, включені в ОС або надавані сторонніми програмами. Додаток, що працює в

користувальницькому режимі може завантажити драйвер пристрою, код якого працює в режимі системного ядра. От чому шкідливі драйвера не повинні завантажуватися, і необхідний постійний контроль за цим. На 64-бітних системах Windows цей метод практично непридатний для використання шкідливими програмами через запит цифрового підпису кожного драйвера працюючого в режимі ядра.

Установка служб (Service Installation) – Системні служби в ОС Windows – спеціальні програми, які можуть працювати, навіть коли завершений сеанс користувача. Вони є більше пріоритетними в порівнянні зі звичайними додатками, не вимагають прямої взаємодії з користувачами й можуть запускатися автоматично під час завантаження системи. Деякі служби не мають своїх власних процесів і розміщаються в інших схожих службах усередині спеціальних процесів. Служби є дуже простим способом для шкідливого ПЗ щоб закріпитися в системі. Антивірусні програми також звичайно мають одну або кілька служб. Шкідливі програми можуть відключити найважливіші компоненти антивірусу, якщо не контролювати постійно установку системних служб. Більше того, Для установки драйверів, що працюють у режимі ядра використовується той же інтерфейс, що й для установки системних служб.

Доступ до файлу HOSTS – файлу бази даних доменних імен (HOSTS File Access) – HOSTS файл – спеціальний файл, що містить відповідності мережних імен і IP адрес. Говорячи загальними словами, мережні імена це домени, а зв'язки між доменом і IP адресою визначаються за допомогою проколу системних доменних імен. Як би те не був, саме файл HOSTS використовується для перекладу мережних імен, включаючи домени, в IP адреси. Таким чином, за допомогою файлу HOSTS можливо перенаправляти домен до довільного IP адресі. Основний прийом вірусів полягає в перенапрямку серверів відновлень антивірусної програми до неіснуючих адрес, що паралізує можливість відновлення антивірусу. Інший прийом використовується для фішингу – перенапрямку домена різних електронних платіжних системам до шкідливих

серверів, які виглядають ідентично оригінальному сайту й здійснення крадіжки конфіденційних платіжних даних.

Активні зміни робочого стола (Active Desktop Changes) – ранні версії ОС Windows мали можливість внесення активного вмісту користувачем на робочий стіл. Ця опція дозволяла створювати повністю робочі столи, що набудовуються. Активний робочий стіл може зловмисно використовуватися шкідливим ПЗ під маскою довіреного додатка провідника Windows. Windows Vista і більше пізні системи не мають підтримку активного робочого стола.

Папки автозавантаження й автозапуску (Autoruns, Autostart Locations) – Додаток має безліч способів для установки в ОС із наступним автозапуском при перезавантаженні системи. Деякі із цих способів дозволяють заражати різні системні процеси шкідливою бібліотекою DLL, тобто виконувати впровадження DLL. У загальному випадку, шкідливі програми використовують кілька папок автозавантаження для того, щоб улаштуватися в системі.

Реєстрація уведень із клавіатури, кейлоггінг (Keylogging, Keyboard Logging) – спостереження за діями користувача є ще однією популярною діяльністю шкідливим програм. Методи реєстрації клавішного уведення дозволяють одержати інформацію, що користувач уводив в інший додаток за допомогою клавіатури. Використання цих методик дозволяє красти паролі, уведені в браузері, поштовому клієнті або клієнті обміну текстовими повідомленнями. Деякі прийоми кейлоггінга ґрунтуються на впровадженні DLL або захвату віконного інтерфейсу.

Захват зображень із екрана й логгінг буфера обміну (Screen and Clipboard Logging) – скринлоггінг і реєстрація буфера обміну також використовується для крадіжки точної конфіденційної інформації. Виконання знімків з екрана може бути використане для крадіжки даних кредитної картки, уведених на безпечної веб-сторінці в браузері. Логгінг буфера обміну дозволяє украсти дані, які користувач використовує для копіювання в ОС Windows. Багато користувачів переносять конфіденційну інформацію, таку як паролі через буфер

обміну. Звичайно це трапляється, коли веб-додаток запитує складні паролі. З одного боку, використання складних паролів є необхідністю, тому що вони менш уразливі для злому, але з іншої сторони користувач, що використовує подібні паролі в різних додатках, фізично не в змозі їх запам'ятати й використовує програму для зберігання паролів або просто текстовий файл для копіювання й вставки пароля у відповідну форму.

Загарбник вікон, захват системних подій (Window Hooking, Windows and WinEvent Hooks) – захват віконних повідомлень Windows і так званих системних подій дозволяє ОС запропонувати ряд спеціалізованих прикладних програмних функцій для програм з метою моніторингу віконних повідомлень і сформованих повідомлень про системні події. Ці функції також можуть бути використані шкідливими ПЗ для впровадження злобливих дій, таких як впровадження DLL бібліотек або кейлоггінг.

Керування компонентами (Component control)

Також називають: **відомі компоненти (Known Components)**

Кожний додаток використовує один або кілька виконуємих модулів, які іноді називають компонентами. Основний модуль – як правило, файл із розширенням .exe, який припускає завантаження деяких динамічно зв'язаних бібліотек (файлів з розширенням .dll), що перебувають у тій же директорії. Основні додаток використовують бібліотеки ядра системи: Kernel32.dll, KernelBase.dll, ntdll.dll, Advapi32.dll, user32.dll і інші. Безліч програм використовують сторонні бібліотеки, які встановлюються в систему разом з основним програмним модулем. Файли .dll можуть завантажуватися до пам’яті або під час ініціалізації додатка, або під час запиту певної функціональності в додатку. Таким чином, кожний додаток має певний набір файлів бібліотек, які завантажуються до пам’яті й від яких залежить його робота. Контроль компонентів (Component Control) визначає ці залежності й контролює завантаження модулів у процес додатка. Коли шкідливе ПЗ намагається

впровадити свою бібліотеку DLL в інший процес, компонентний контроль розпізнає й забороняє ця небезпечна дія.

Компонентний контроль також гарантує недоторканність достовірних безпечних модулів. Будь-які спроби змінити файли надійних відомих модулів можуть бути розпізнані й заблоковані. Це відноситься як до головних файлів, що виконуються, так і до файлів динамічно зв'язаних бібліотек.

Системний щит (System Guard)

Деякі антивірусні програми мають різні рівні захисту для сторонніх програм, а також для ОС і її компонентів. Системний щит є частиною поведінкового аналізу, що відповідає за захист ОС від шкідливого ПЗ і схоронність оригінальних модулів системи. Важливі системні файли, критичні записи реєстру (у тому числі автозавантаження) і інші системні ресурси, які можуть бути інфіковані вірусом перебувають під спостереженням системного щита. Будь-яка спроба зміни ОС блокується.

Захист переносних мультимедійних пристроїв (Removable Media Protection)

Основна функціональність сучасних антивірусних програм у галузі захисту переносних мультимедійних пристроїв (USB-флешки, зовнішні HDD-диски) припускає відключення функції автозавантаження або автозапуска. Коли переносний пристрій включається в комп'ютер, а його корінна директорія містить файл Autorun.inf, стороння програма може запуститися системою. Це може привезти до непомітного зараження комп'ютера.

Більшість антивірусних рішень також визначають спеціальний набір правил для всіх програм, розташованих на переносних пристроях. Передбачаються, що файли на переносних накопичувачах можуть з'явитися з інших ПК, інфікованих і не оснащених достатнім рівнем безпеки. От чому програми на переносних пристроях уважаються за замовчуванням потенційно небезпечними і їхньою дією строго обмежені. Деякі пакети безпеки можуть

розпізнавати програми з електронним підписом від надійних джерел і не обмежувати дії таких додатків.

Самозахист (Self-protection)

Поведінковий аналіз також відповідає за одну із самих критичних функцій – самозахист антивірусної програми. Будь-який антивірусний захист може виявитися марної, якщо шкідливе ПЗ може відключити її. Сучасні антивірусні програми захищають всі свої компоненти від вірусної погрози так щоб вони не могли бути відключені або ушкоджені. Самозахист припускає захист програмних процесів і потоків, файлів і директорій, записів реєстру і їхніх значень, установлених системних драйверів і служб, інтерфейсів COM і інших ресурсів, створених антивірусом і доступних для інших процесів у системі.

Запобігання інфікування найважливіших процесів є життєво необхідним для будь-якої антивірусної програми. Безліч пакетів безпеки покладаються на постійні відновлення їхньої антивірусної бази. Процес відновлення розробляється максимально невразливим для шкідливого ПЗ, щоб вірус не міг зупинити завантаження або установку відновлень або завантажити підмінні файли відновлень. Самозахист, як правило, включений в основний набір правил поведінкового аналізу, які забороняють керування ресурсами антивірусного продукту. Самозахист може йти окремо від модуля безпеки, що управляє сторонніми програмами. У другому випадку компонента антивіруси краще захищені, ніж будь-який інший додаток у системі. Обидва підходи мають місце в сучасних антивірусних рішеннях.

Веб-захист браузерів (Web and Browser protection)

Також називають: Веб-Контроль (Web Control), веб-безпека (Web Security), веб-захист (Web Protection), захист браузера (Browser Protection), захист перегляду веб-сторінок (Web Browsing Protection)

Контроль плагінів (Plugin Control)

Також називають: **захист плагінів (Plugin Prevention)**

Ланцюжки браузерних процесів, файли й інші ресурси можуть бути захищені за допомогою функціональності стандартного поведінкового контролю, але в загальному випадку ці міри недостатні. Більшість основних браузерів мають плагіни, доповнення, панелі керування, додаткові помічники, які можуть також загрожувати їхньої безпеки. Контроль компонентів у цьому випадку може прийти на допомогу. Антивірус повинен переконатися, що все компоненти браузерного додатка є достовірними й безпечними для користувача.

Фільтри URL-адрес, блокування реклами (Domain and URL Filters, Blocking Ads

Внесення в чорний список шкідливих доменів і посилань є основною функцією. Користувачам, у загальному випадку, дозволяється додавати власні адреси в список фільтрів. Коли браузер намагається з'єднатися із заблокованою адресою, фільтр виявляє цю спробу й забороняє дію до того, як почалася передача даних. Дана функціональність може вступати в дію або на мережному рівні з низькорівневим драйвером ядра, або на рівні розширення браузера.

Розширення браузера є більше розповсюдженим, тому що поводження браузера дуже легко контролюється в цьому випадку й реальне повідомлення про помилку виводиться перед мнимою помилкою з'єднання, ініціюємого шкідливим ПЗ. За замовчуванням фільтри містять шляхи призначення, відомі як злобливі або незаконні: сайти, що використовують фішинг, облудні дії, ресурси, що мають шкідливий код, погану репутацію й порнографічний зміст.

Блокування реклами може бути частиною веб-контролю. Деякі антивіруси пропонують користувачеві заблокувати рекламу, незважаючи на те, що рекламний зміст є важливим складовий бізнес моделей інтернету. Якщо антивірусне ПЗ містить фільтри ресурсів, воно може легко визначити рекламних провайдерів і з високою ефективністю заблокувати більшість рекламних оголошень.

Інші форми блокування реклами включають блокування зображень по їхньому розмірі, блокування спеціальних ключових слів і застосовуються проти

настирливого й неналежного контенту. Більшість легітимних рекламних оголошень, відображаються, як правило, коректно.

Динамічний зміст (Dynamic Content)

Динамічний веб-зміст, таке як Flash ролики, Java додатки, об'єкти Active представляють нові способи обходу сучасних захистів. Антивіруси можуть контролювати й блокувати динамічні об'єкти, які завантажуються з недостовірних сайтів. Деякі продукти містять блокування схованих фреймів і спливаючих вікон, які можуть дратувати користувачів.

Найбільш твердий метод – блокування всього динамічного контенту, включаючи скрипти Java. Як би те не було, побічним ефектом цих мір може послужити некоректна робота багатьох сайтів.

Куки (Cookies)

Файли куки – невеликі файли, які можуть зберігатися на користувальницькому комп'ютері при відповідному запиті браузера. Веб-сайт може потім звертатися до файлів cookie щоб розширити свою функціональність і рівень взаємодії з користувачем. Як би те не було, механізм використання куки може бути застосований для відстеження користувальницької активності в мережі.

Це може бути розцінене як неприпустиме порушення користувальницької конфіденційності. Сучасні антивірусні програми дозволяють користувачеві управляти cookie-файлами: видаляти їх або повністю забороняти їхнє створення. Незважаючи на те, що ця функціональність є невід'ємної складової сучасних веб-браузерів, користувачеві може бути більш зручно управляти безпекою комп'ютера з одного додатка – антивірусної програми.

Також як і блокування динамічного вмісту, блокування файлів cookie може викликати порушення в роботі багатьох веб-сайтів.

Віртуалізація браузера (Browser Virtualization)

Віртуалізація браузера є реакцією розроблювачів антивірусного ПЗ на граничну уразливість інтернет браузерів у настільних системах і підвищеній

привабливості для кибер-атак. Браузер використовуються для серфінгу по мережі й у випадку взаємодії із сайтом зі шкідливим змістом, може з'явитися уразливість у браузері, що буде сприяти зараженню всього ПК.

Віртуалізація браузера полягає в створені віртуального середовища для роботи інтернет браузера. Всі дії браузера контролюються, якщо вони вважаються потенційно небезпечними, вони перенаправляються в пісочницю. Наприклад, браузер заражений шкідливим кодом, він намагається зберегти злобливі файли на комп'ютері й змінити ключі реєстру для наступного автозапуску вірусу при перезавантаженні системи. Якщо файлові операції й будь-які дії з реєстром будуть відбуватися у віртуальному середовищі, запуск вірусу відбудеться в пісочниці й реальному зараженні системи не відбудеться. При закритті браузера віртуальне середовище знищується, і зараження ліквідується назавжди. У той же час користувачеві дозволено завантажувати необхідні файли в реальну систему, поза пісочницею. Достовірні дії браузера не перенаправляються у віртуальне середовище, користувальницька функціональність браузера, таким чином, зберігається.

Віртуалізація браузера дуже нагадує функцію пісочниці, що ми обговорювали раніше. Відмінність полягає в тому, що цього разу в пісочниці виконуються не підозрілі додатки, а добре відомий і достовірний веб-браузер.

Браузерний і пошуковий помічник, анти-фішинг (Browser and Search Advisor, Anti-phishing)

Також називають: **безпечний Інтернет (Safe Web)**

Призначення браузерного й пошукового помічника – надання інформації про репутацію відвідуваного веб-сайта. Якщо браузерний помічник визначає потенційно небезпечний веб-сайт, що користувач має намір відвідати, виводиться відповідне попередження. Пошуковий порадник зосереджений саме на результатах видачі пошукових машин. При запиті сторінка пошукової системи модифікуються таким чином, щоб потенційно небезпечні посилання або зовсім

були невидимими, або принаймні позначені додатковою інформацією про їхню небезпеку для користувача.

Вендори антивірусного ПЗ створюють власні рейтинги інтернет сайтів, використовуючи різні критерії. Веб-сайти скануються на наявність посилань, що ведуть на шкідливі ресурси. Контент сайтів аналізується й класифікується відповідно до фільтрів ключових слів. Застосовується також додавання в білі й чорні списки. Одним з найбільш важливих параметрів ранжирування сайту антивірусом є рейтинг співтовариств. Співтовариство користувачів певного антивірусного продукту є потужною захисною одиницею, що дозволяє користувачам уникати злобливі веб-сайти. Якщо значна кількість користувачів антивірусу відвідують шкідливий сайт, існує більша ймовірність, що багато користувачів позначать цей сайт як небезпечний за допомогою антивірусного ПЗ. Ця негативна репутація згодом буде використана для попередження інших користувачів співтовариства.

Опція анти-фішингу використовуються для запобігання крадіжки платіжних даних. Існує безліч способів розпізнати сайт, що використовує фішинг. Ці методи включають аналіз змісту, виявлення недійсних або несправжніх сертифікатів, детектування підозрілих посилань і т.д. Якщо виявлено спробу фішингу з боку шкідливого сайту, він блокується антивірусом, або критичне попередження виводиться користувачеві.

Сканування веб-змісту (Web Content Scanning)

Деякі функції, які ми обговорили вище, ґрунтуються на скануванні веб-контенту. Це завдання є основною для всіх антивірусних движків і дуже схожі зі скануванням файлів. Єдина проблема полягає в методах шифруванні інформації – наприклад, протокол HTTPS або будь-який інший, працюючий за технологією SSL/TLS, що використовує передачу конфіденційних даних веб-додаткам. Цей вид шифрування розроблений з тією метою, що навіть у випадку, якщо зловмисники побажають переглянути всі дані сесії, виникнути серйозні утруднення при їхньому дешифруванні. Єдиними об'єктами, здатними

дешифрувати дані є відправник і одержувач, тобто серверний додаток і користувальницький браузер, наприклад.

Це означає, що антивірусам дуже важко виявити які саме дані передаються й чи не представляють вони небезпеки для користувача. Як би те не було, сучасні антивірусні продукти можуть установлювати свої власні модулі в браузери й, таким чином, стає частиною об'єкта, що має доступу до сирої, незашифрованої інформації. Ця схема діє навіть для сканування веб-змісту, переданого в зашифрованому виді. От чому дуже важливо вести контроль за плагінами й доповненнями, що працюють усередині браузера. Якщо браузер інфікований шкідливим модулем, вірус може спокійно обійти шифрування безпечної сесії.

Захист конфіденційної інформації (Privacy Protection)

Також називають: захист персональної інформації (ID Protection), безпека особистих даних (Identity Safe), захист особистих даних (Identity Protection).

Номера кредитних карт, банківських аккаунтів, електронних платіжних систем і інших паролів, персональна інформація, адреси електронної пошти, номера документів, телефонів є строго конфіденційною інформацією, що повинна бути захищена від дій шкідливого ПЗ.

Функція захисту персональних даних дозволяє користувачеві вирішити, які дані є найбільш конфіденційними й не повинні передаватися без його згоди. Вихідний трафік сканується на предмет даних, що захищаються, і у випадку збігу, передача даних блокується.

Менеджер паролів (Password Management) іноді входить до складу захисту персональних даних. Користувачі можуть зберігати паролі в безпеці, тому що вони перебувають у зашифрованій формі й доступні тільки власникові. Деякі антивірусні програми розширюють область шифрування, включаючи всі користувальницькі файли й, таким чином, захищають налаштування сторонніх програм і їхні журнали реєстрації подій (логи). Ця міра відноситься до різних чатів і програм обміну повідомленнями, які можуть зберігати історію

повідомлень із конфіденційною інформацією на жорсткому диску. Історія браузера, файли cookie і тимчасові файли також перебувають під захистом функції забезпечення безпеки особистих даних. Користувач може налаштувати періодичність очищення цих даних: щораз при закритті браузера (цей агресивний підхід може привести до довгого завантаження деяких веб-сайтів), раз у день, раз у тиждень. Використання розкладу означає, що моніторинг користувальницької активності в мережі обмежений певним періодом часу.

Інші блоки схеми були більш докладно розглянуті в структурній схемі.

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

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання бакалаврського проектування, наведена на рисунку 3.3. Першим процесом, який завантажується у системі, є процес виведення на екран головного модуля антивірусу. Він взаємодіє з наступними процесами:

– Процес сканування трафіку.

– Процес сканування ресурсів.

– Процес он лайн резервного копіювання.

– Процес веб-захисту браузерів.

Процес сканування трафіку взаємодіє з наступними процесами:

– Процес роботи фаєрволу.

– Процес роботи поведінкового контролю.

– Процес роботи анти-спам фільтру.

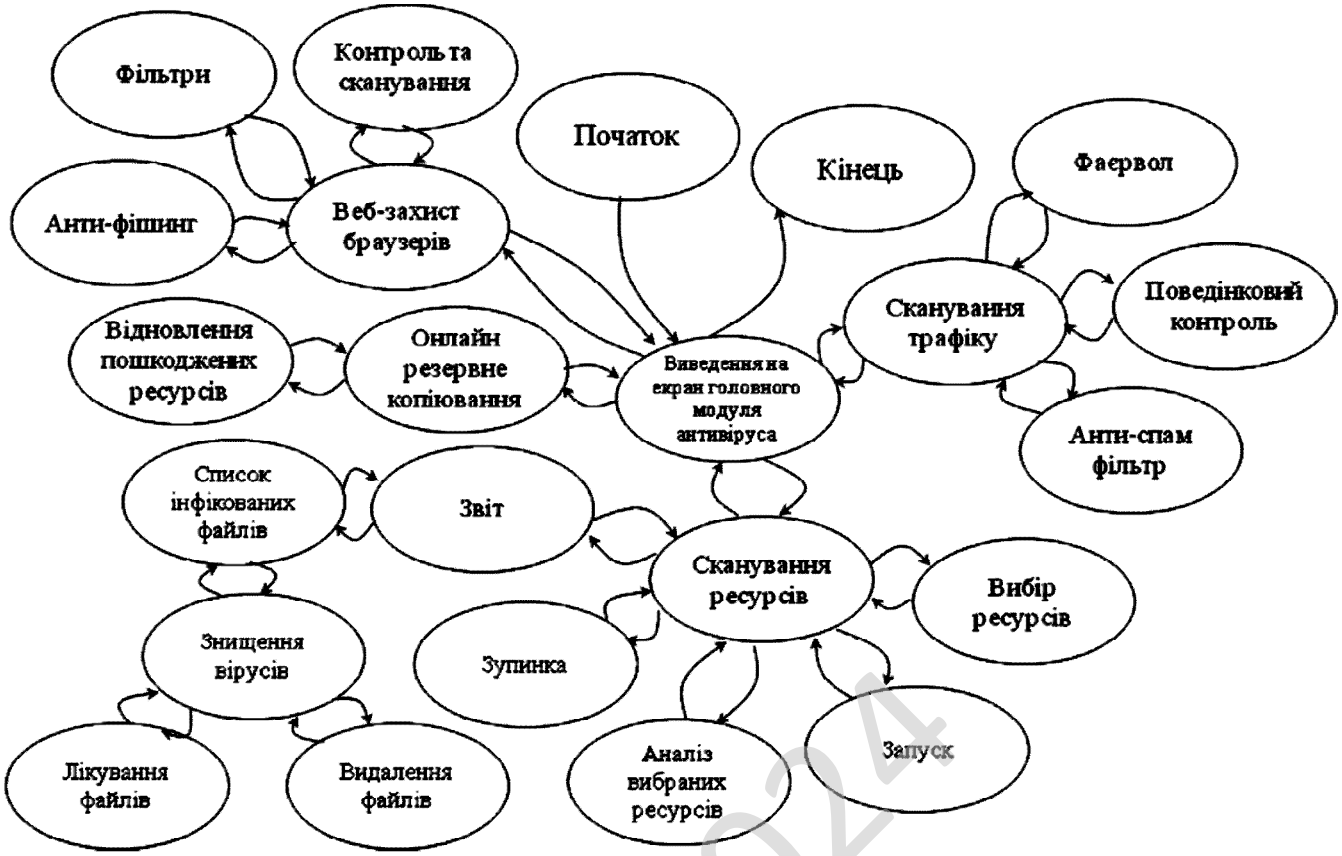

Рисунок 3.3 – Діаграма взаємодії процесів

Процес сканування ресурсів взаємодіє з наступними процесами:

– Процес вибору ресурсів.

– Процес запуску ресурсів.

– Процес аналізу вибраних ресурсів.

– Процес зупинки ресурсів.

– Процес створення звіту.

Останній процес взаємодіє з процесом формування списку інфікованих файлів, який, у свою чергу, взаємодіє з процесом знищення вірусів.

Процес знищення вірусів взаємодіє з наступними процесами:

– Процес лікування файлів.

– Процес видалення файлів.

Процес он лайн резервного копіювання взаємодіє з процесом відновлення пошкоджених ресурсів.

Процес веб-захисту браузерів взаємодіє з наступними процесами:

– Процес роботи анти-фішингу.

– Процес роботи фільтрів.

– Процес роботи контролю та сканування.

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

4 РЕАЛІЗАЦІЯ 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 списку ресурсів.

Наступним кроком є обирання, чи потрібно встановлювати фільтри.

Якщо це потрібно, то виконуються наступні дії:

– Виводиться список фільтрів.

– Встановлюються параметри фільтрації.

– Запускається фільтрація трафіку та URL-адрес.

У протилежному випадку відбуваються наступні дії.

Відбувається запит на потребу сканування.

Якщо це потрібно, то виконуються наступні дії:

– Обираються ресурси для сканування.

– Обирається об’єкт для перевірки.

– Запускається підпрограма сканування ресурсів та знищення вірусів.

Якщо вірус знайдено то виводиться інформація про вірус.

Після цього виводиться запит, чи потрібно вилікувати уражений файл.

Якщо так, то відбувається його лікування, у іншому випадку відбувається запит на знищення цього файлу. Відповідно до результатів запиту відбувається або видалення файлу з вірусом, або переміщення файлу у карантин.

Після цього відбувається запит на завершення сканування, й при позитивній відповіді програма закінчує роботу.

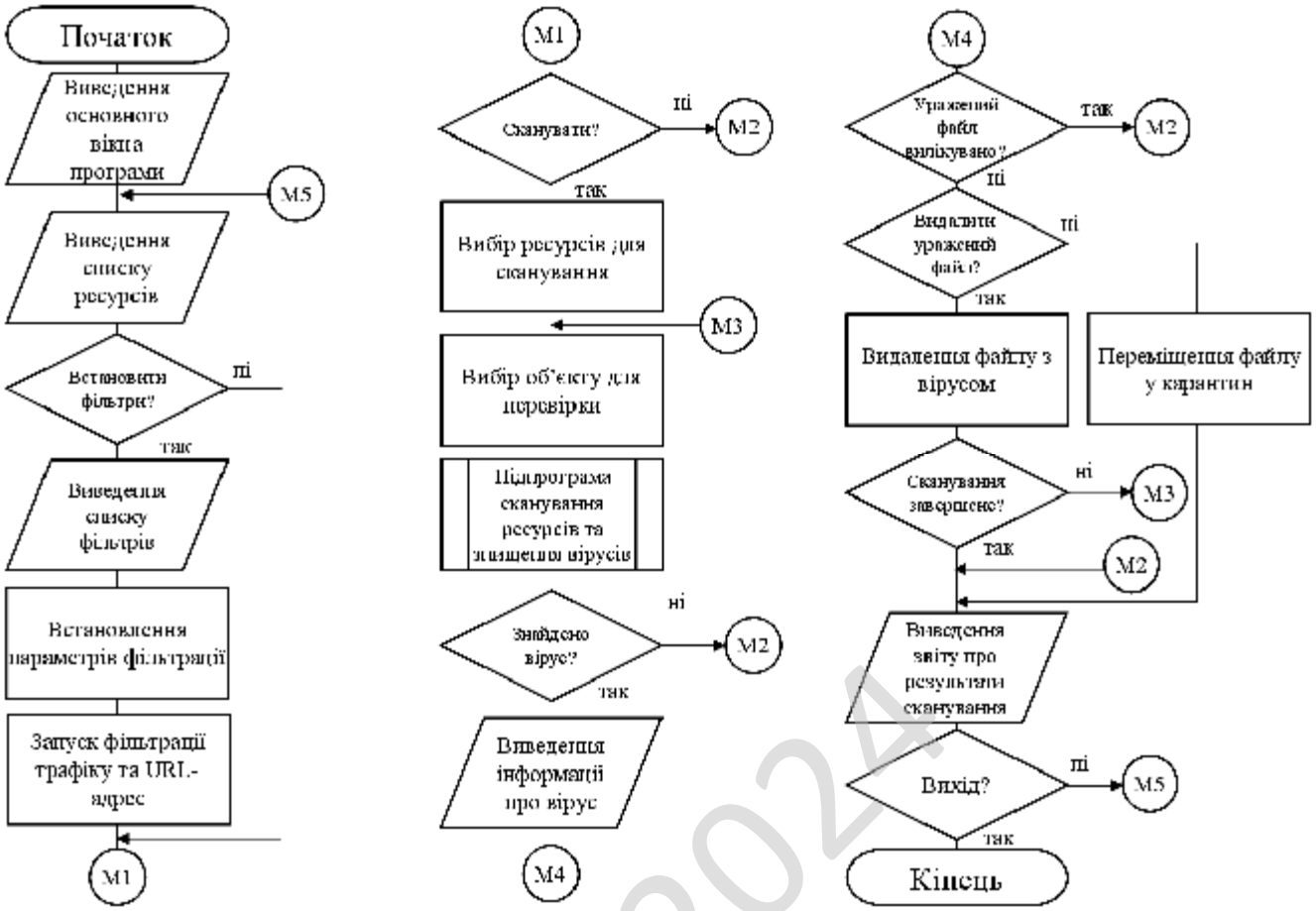

Рисунок 4.1 – Блок-схема основної програми

На рисунку 4.2 зображена блок-схема підпрограми сканування ресурсів та знищення вірусів. Наведена підпрограма працює наступним чином.

Спершу відбувається завантаження об’єкту для перевірки.

Після цього відбувається пошук вірусів за контрольними сумами.

Наступним етапом є сигнатурний пошук.

Крім того відбувається криптоаналіз та евристичний аналіз.

Якщо детектовано шкідливий код, то відбувається визначення типу вірусу.

Якщо вірус ідентифіковано, то відбувається спроба вилікувати файл.

Якщо спроба вилікувати була невдала, то відбувається запит на видалення файлу у користувача.

Якщо вірус не ідентифіковано, то відбувається аналіз та дослідження нового вірусу й оновлення бази даних антивірусної програми.

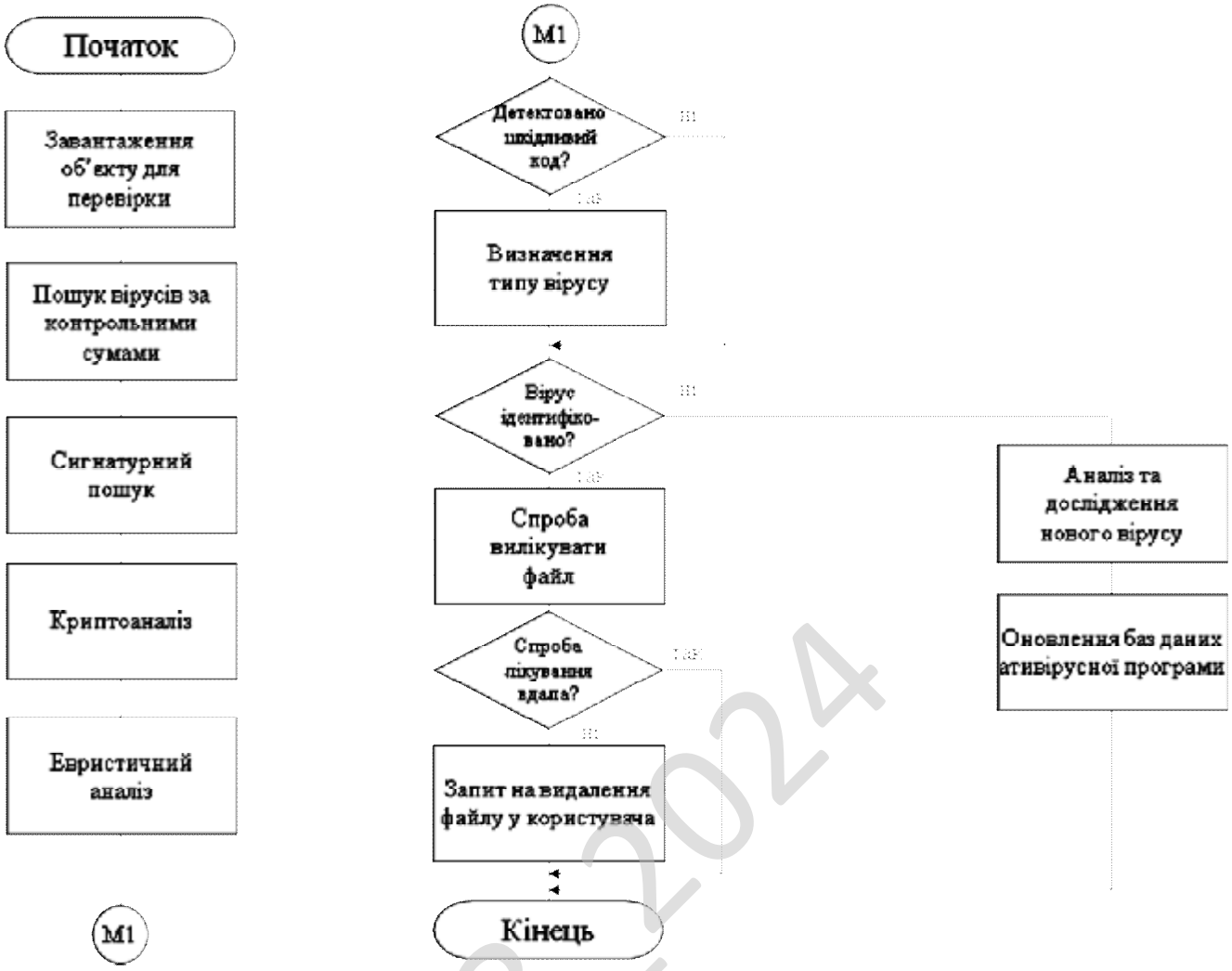

Рисунок 4.2 – Блок-схема підпрограми сканування ресурсів та знищення вірусів

Надалі наведемо програмний код ряду функцій, які реалізовані у програмному забезпечені антивірусного захисту файлових серверів.

```
//***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_ANTI_VIRUS_FOR_FILE_SERVER_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_ANTI_VIRUS_FOR_FILE_SERVER_SCAN_FILE;
          OptionsForm.PathList.Clear;
          OptionsForm.ExtList.Clear;
          for i := 0 to Anti_Virus_For_File_ServerConfig.Count-1 do begin
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'EXT' then
              with OptionsForm.ExtList.Items.Add do begin
                Caption := GetParam(Anti_Virus_For_File_ServerConfig[i]);
                ImageIndex := 3;
              end;
              if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'SHOWBALOONHINT' then
              if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.SHOWBALOONHINT.Checked := False else
              OptionsForm.SHOWBALOONHINT.Checked := True;
              if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'PROCCONTROLAUTOMODE' then
              if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.PCAutoLoad.Checked := False else
              OptionsForm.PCAutoLoad.Checked := True;

              if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'PROCCONTROLAUTOKILL' then
              if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.PCAutoKill.Checked := False else
              OptionsForm.PCAutoKill.Checked := True;
              if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'PROCCONTROLAUTOACTION' then
              if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.PCAutoAction.Checked := False else
              OptionsForm.PCAutoAction.Checked := True;
              if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'PROCCONTROLDELINFECT' then
              if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.PCDelInfect.Checked := False else
```

```
            OptionsForm.PCDelInfect.Checked := True;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'PROCCONTROLSKIPINFECT' then
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
OptionsForm.PCSkipInfect.Checked := False else
            OptionsForm.PCSkipInfect.Checked := True;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'HIDETIP' then
begin
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'OFF' then
HideForm.ShowHideTip.Checked := False else
            HideForm.ShowHideTip.Checked := True;
            end;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'PATH' then
begin
            with OptionsForm.PathList.Items.Add do begin
              Caption := GetParam(Anti_Virus_For_File_ServerConfig[i]);
              if DirectoryExists(Caption) then ImageIndex := 4 else ImageIndex :=
5;
            end;
            end;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'AUTOSAVEREPORT' then
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'ON' then
OptionsForm.AutoSaveReport.Checked := true else
            OptionsForm.AutoSaveReport.Checked := False;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'REGISTERSYSMENU' then
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'ON' then
OptionsForm.RegisterSysMenu.Checked := true else
            OptionsForm.RegisterSysMenu.Checked := False;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'AUTORUN' then
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'ON' then
OptionsForm.AUTORUN.Checked := true else
            OptionsForm.AUTORUN.Checked := False;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'AUTOHIDE'
then
            if GetParam(Anti_Virus_For_File_ServerConfig[i]) = 'ON' then
OptionsForm.AUTOHIDE.Checked := true else
            OptionsForm.AUTOHIDE.Checked := False;
            if GETParamName(Anti_Virus_For_File_ServerConfig[i]) =
'AUTOSAVEREPORTTO' then OptionsForm.ReportSavePath.Text :=
GETParam(Anti_Virus_For_File_ServerConfig[i]);
```

```
        end;
      end;
      //***Функція сканування***//
      Procedure TMainForm.StartAnti_Virus_For_File_Server_Scan(Parametr: String);
        var
        T : String;
      begin
        if GetDBRecCount = 0 then
        begin
          MessageFrm.Caption := DBerrorI1;
          MessageFrm.InformationLabel.Caption := DBerrorI1;
          MessageFrm.InfoLabel.Caption := DBerrorI2;
          MessageFrm.Memo1.Text := DBerrorI3;
          MessageFrm.ShowModal;
          Exit;
        end;
        if Parametr = 'DRV' then
        begin
          Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
TRUE;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction   :=
TAnti_Virus_For_File_Server_ScanDir;
          Path.Add(ExtractFileDrive(Paramstr(0))+'\');
          Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Dirs
:= Path;
          OnAnti_Virus_For_File_Server_ScanStart;
          exit;
        end;
        if DirectoryExists(Parametr+'\') then
        begin
          Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
TRUE;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction   :=
TAnti_Virus_For_File_Server_ScanDir;
```

```
            Path.Add(Parametr+'\');
            Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Dirs
:= Path;
            OnAnti_Virus_For_File_Server_ScanStart;
            exit;
          end;
          if FileExists(Parametr) then
          begin
            Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
false;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction :=
TAnti_Virus_For_File_Server_ScanFile;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.FileName :=
Parametr;
            OnAnti_Virus_For_File_Server_ScanStart;
            exit;
          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 := _Anti_Virus_For_File_Server_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;
```

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою MARS, який є блочно-симетричним шифром з відкритим ключем. Розмір блоку при шифруванні 128 біта, розмір ключа може варіюватися від 128 до 448 біт включно (кратні 32бітам). Творці прагнули поєднати в своєму алгоритмі швидкість кодування і стійкість шифру. В результаті вийшов один з самих криптостійкий алгоритм з алгоритмів, які брали участь в конкурсі AES.

Алгоритм унікальний тим, що використовував практично всі існуючі технології, застосовувані в криптоалгоритмах, а саме:

– Найпростіші операції (додавання, віднімання, виключаюче або).

– Підстановки з використанням таблиці замін.

– Фіксований циклічний зсув.

– Залежний від даних циклічний зсув.

– Множення за модулем 2^{32}.

– Ключове забілювання.

Використання подвійного перемішування представляє складність для криптоаналіз а, що деякі відносять до недоліків алгоритму. У той же час на даний

момент не існує будь-яких ефективних атак на алгоритм, хоча деякі ключі можуть генерувати слабкі підключі.

Структура алгоритму

Автори шифру виходили з наступних припущень:

1. Вибір операцій. MARS був спроектований для використання на найсучасніших комп'ютерах того часу. Для досягнення найкращих захисних характеристик в нього були включені самі «сильні операції» підтримувані в них. Це дозволило добитися більшого відносини securityper-instruction для різних реалізації шифру.

2. Структура шифру. Двадцятирічний досвід роботи в області криптографії підштовхнув творців алгоритму до думки, що кожен раунд шифрування грає свою роль в забезпеченні безпеки шифру. Зокрема, ми можемо бачити, що перший і останній раунди зазвичай сильно відрізняються від проміжних («центральних») раундів алгоритму в плані захисту від криптоаналітичних атак. Таким чином, при створенні MARSa використовувалася змішана структура, де перший і останній раунди шифрування істотно відрізняються від проміжних.

3. Аналіз. Швидше за все, алгоритм з гетерогенною структурою буде краще протистояти криптоаналітичних методам майбутнього, ніж алгоритм, всі раунди якого ідентичні. Розробники алгоритму MARS надали йому сильно гетерогенну структуру – раунди алгоритму дуже різняться між собою.

У шифрі MARS використовувалися такі методи шифрування:

1. Робота з 32-х бітними словами. Всі операції застосовуються до 32-бітовим словами. тобто вся початкова інформація розбивається на блоки по 32біта. (Якщо ж блок опинявся меншої довжини, то він доповнювався до 32біт)

2. Мережа Фейстеля. Творці шифру вважали, що це найкращий варіант поєднання швидкості шифрування і криптостійкості. В MARS використана мережа Фейстеля 3-го типу.

3. Симетричність алгоритму. Для стійкості шифру до різних атакам всі його раунди були зроблені повністю симетричними, тобто друга частина раунду є дзеркальне повторення першої його частини.

Структуру алгоритму MARS можна описати таким чином:

1. Попереднє накладення ключа: на 32-бітові субблоки A, B, C, D накладаються 4 фрагмента розширеного ключа k_0 … k_3 операцією складання за модулем 2^{32}.

2. Виконуються 8 раундів прямого перемішування (без участі ключа шифрування).

3. Виконуються 8 раундів прямого криптоперетворення.

4. Виконуються 8 раундів зворотного криптоперетворення. [2]

5. Виконуються 8 раундів зворотного перемішування, також без участі ключа шифрування.

6. Фінальне накладення фрагментів розширеного ключа k_{36} … k_{39} операцією віднімання за модулем 2^{32}.

Пряме перемішування

У першій фазі на кожне слово даних накладається слово ключа, а потім відбувається вісім раундів змішування згідно з мережею Фейстеля третього типу спільно з деякими додатковими змішування. У кожному раунді ми використовуємо одне слово даних (зване, вихідним словом) для модифікації трьох інших слів (звані, цільовими словами). Ми розглядаємо чотири байта вихідного слова як індексів на двох S-блоків, S_0 і S_1 , кожен, що складається з 256 32-розрядних слів, а далі проводимо операції XOR або додавання даних відповідного S-блоку в три інших слова.

Якщо чотири байти вихідного слова b_0 , b_1 , b_2, b_3 (де b_0 є першим байтом, а b_3 є старшим байтом), то ми використовуємо b_0 , b_2, як індекси в блоку S_0 і байти b_1 , b_3 , як індекси в S-блоці S_1 . Спочатку зробимо XOR S_0 до першого цільовим речі, а потім додамо S_1 до того ж слова. Ми також додаємо S_0 до другого цільовим

слову і XOR блоку-S_1 до третього цільовим слову. У висновку, ми обертаємо вихідне слово на 24 біта вправо.

У наступному раунді ми обертаємо наявні у нас чотири слова: таким чином, нинішні перші цільове слово стає наступним вихідним словом, поточним другий цільове слово стає новим першим цільовим словом, третє цільове слово стає наступний другий цільовим словом, і поточне вихідне слово стає третім цільовим словом.

Більш того, після кожного з чотирьох раундів ми додаємо одне з цільових слів назад у вихідне слово. Зокрема, після першого і п'ятого раундів ми додав третю цільове слово назад у вихідне слово, а після другого і шостого раунду ми додаємо першої цільової слово назад у вихідне слово. Причиною цих додаткових операцій змішування, є ліквідація декількох простих диференціальних криптоатаки в фазі перемішування, щоб порушити симетрію у фазі змішування та отримати швидкий потік.

Криптографічне ядро

Криптографічне ядро MARS – мережа Фейстеля 3-го типу, що містить в собі 16 раундів. У кожному раунді ми використовуємо ключову E-функцію, яка є комбінацією множень, обертань, а також звернень до S-блоків. Функція приймає на вхід слово даних, а повертає три слова, з якими згодом буде здійснена операція додавання або XOR до інших трьох слів даними. У доповненні вихідне слово обертається на 13 біт вліво. Для забезпечення, серйозного опору до криптоатаки, три вихідних значення E-функції (O_1 , O_2, O_3) використовуються в перших восьми раундах і в останніх восьми раундах в різних порядках. У перші вісім раундів ми додаємо O_1 і O_2до першого і другого цільовим речі, відповідно, і XOR O_3 у третьому цільовим слову. За останні вісім раундів, ми додаємо O_1 і O_2 до третього і другого цільовим речі, відповідно, і XOR O_3 до першого цільовим слову.

E-функція

E-функція приймає як вхідні дані одне слово даних і використовує ще два ключових слова, виробляючи на виході три слова. У цій функції ми

використовуємо три тимчасові змінні, що позначаються L, M і R (для лівої, середньої та правої).

Спочатку ми встановлюємо в R значення вихідного слова зміщеного на 13 біт вліво, а в M – сума вихідних слів і першого ключового слова. Потім ми використовуємо перші дев'ять бітів M як індекс до однієї з 512S-блоків (яке виходить суміщенням S_0 і S_1 змішуванням фази), і зберігаємо в L значення відповідного S-блоку.

Потім помножимо друге ключове слово на R, зберігши значення в R. Потім обертаємо R на 5 позицій вліво (так, 5 старших бітів стають 5 нижніми бітами R після обертання). Тоді ми робимо XOR R в L, а також переглядаємо п'ять нижніх біт R для визначення величини зсуву (від 0 до 31), і обертаємо M вліво на цю величину. Далі ми обертаємо R ще на 5 позицій вліво і робимо XOR в L. У висновку, ми знову дивимося на 5 молодших бітів R, як на величину обертання і обертаємо L на цю величину вліво. Таким чином результат роботи E-функції – 3 слова (по порядку): L, M, R.

Зворотне перемішування

Зворотне перемішування практично збігається з прямим перемішуванням, за винятком того факту, що дані обробляються в зворотному порядку. Тобто, якби ми поєднали пряме і зворотне перемішування так, щоб їх виходи і входи були б з'єднані в зворотному порядку (D [0] прямого і D [3] зворотного, D [1] прямого і D [2] зворотного), то не побачили б результату перемішування. Як і в прямому змішування, тут ми теж використовуємо одне вихідне слово і три цільових. Розглянемо чотири перших байта вихідного слова: b_0 , b_1 , b_2, b_3 . Будемо використовувати b_0 , b_2як індекс до S-блоку – S_1 , а b_1 b_3 для S_0 . Зробимо XOR S_1 [b_0] в перше цільове слово, віднімемо S_0 [b_3] з другого слова, віднімемо S_1 [b_2] з третього цільового слів і потім проробимо XOR S_0 [b_1] також до третього цільового слова. Нарешті, ми повертаємо початкове слово на 24 позицій вліво. Для наступного раунду ми обертаємо наявні слова так, щоб нинішнє перше цільове слово стало наступним вихідним словом, поточне друге цільове слово

стало першим цільовим словом, поточне третє цільове слово стало другим цільовим словом, і поточне вихідне слово стало третім цільовим словом. Крім того, перед одним з чотирьох «особливих» раундів ми віднімаємо одне з цільових слів з вихідного слова: перед четвертим і восьмим раундами ми віднімаємо першої цільової слово, перед третьому і сьомим раундами ми віднімемо третя цільова слово з вихідного.

Дешифрування

Процес декодування обернений процесу кодування. Код дешифрування схожий (але не ідентичний) на код шифрування.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

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

– Блок меню.

– Блок кнопок швидкого доступу.

– Блок вибору дій.

– Блок виведення інформації про хід роботи антивірусного захисту файлових серверів.

Блок меню включає в себе наступні елементи:

– Файл.

– Вид.

– Інструменти.

– Параметри.

– Довідка.

Блок вибору дій включає в себе наступні елементи:

– Сканування ресурсів.

– Сканування трафіку.

– Веб-захист браузерів.

– Он-лайн резервне копіювання.

– База даних вірусів.

– Журнал подій.

На рисунку 5.2 наведено вікно довідки, у якому вказано інформацію про найменування проекту, про розробника проекту, його керівника, місце розробки проекту й рік розробки.

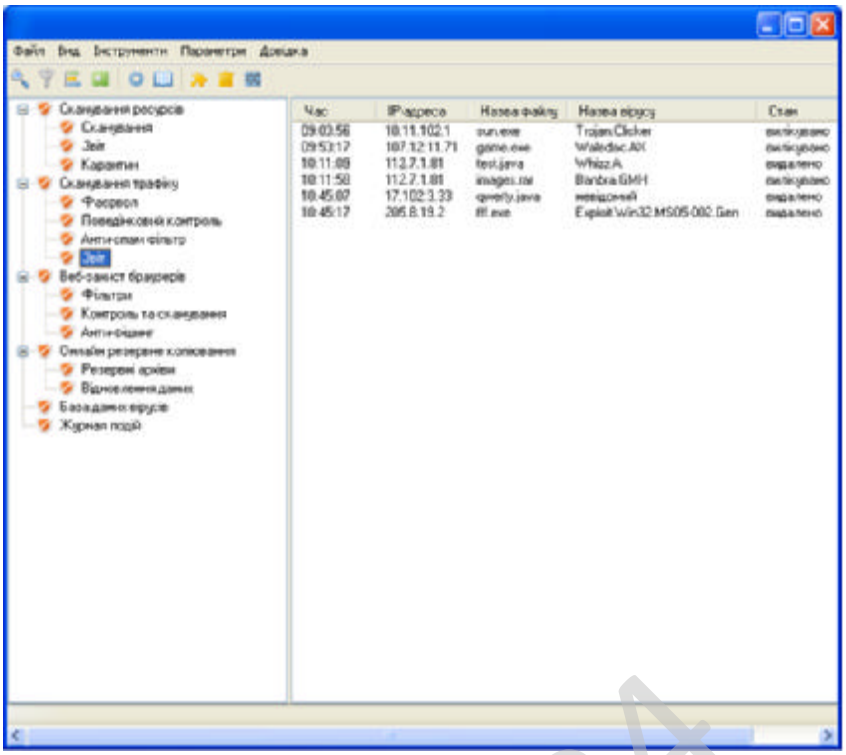

Рисунок 5.1 – Головне вікно програми

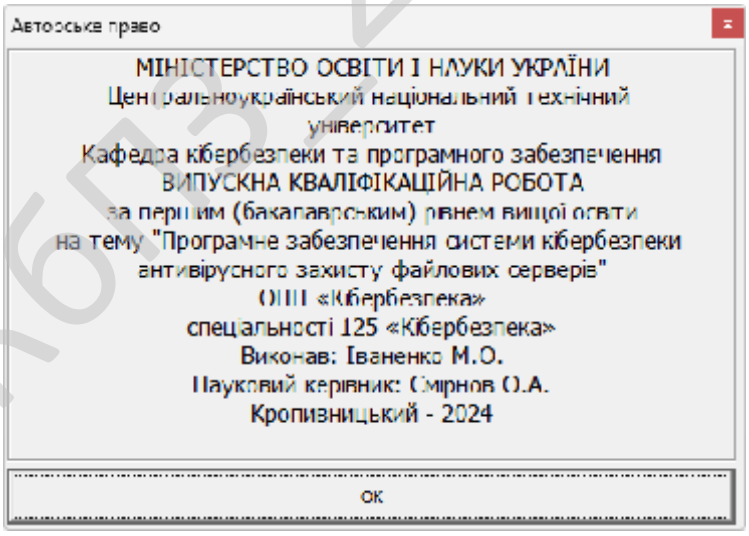

Рисунок 5.2 – Вікно довідки

6 ОСНОВНІ ВИСНОВКИ

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

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем антивірусного захисту файлових серверів.

– Досліджена система антивірусного захисту файлових серверів.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки антивірусного захисту файлових серверів.

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

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 Delphi 10. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи кібербезпеки антивірусного захисту файлових серверів. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної

техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи кібербезпеки й включає програми, що реалізують її функції.

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

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм MARS.

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

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Derek Fisher. Application Security Program Handbook. Manning Publications. 2021. 155 p.

2. Cameron Wyatt PH.D. Kali Linux Tutorial. Independently published. 2021. 60 p.

3. Alex Matrosov, Eugene Rodionov, Sergey Bratus. Rootkits and Bootkits. No Starch Press. 2019. 450 p.

4. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

5. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

6. Kuznetsov, O., Kandiy, S., Frontoni, E., Smirnov, O. «Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece». *CEUR Workshop Proceedings,* Volume 3504, 2023, pp. 1-11.

7. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» *CEUR Workshop Proceedings,* Volume 3187, 2022,

8. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection Means Using a Genetic Algorithm». *In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies*, vol 131. 2023. Springer, Singapore. pp. 21-34.

9. Smirnov O.A., Al-Oraiqat A.M., Ulichev O.S., Meleshko Ye.V., Al-Rawashdeh H.S., Polishchuk L.I. «Modeling strategies for information influence

dissemination in social networks». *Journal of Ambient Intelligence and Humanized Computing* Volume 13, Issue 5. Springer, Cham. 2022, pp. 2463-2477.

10. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». *11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021,* Cracow, Poland, 22-25 September 2021. P. 414-418

11. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». *4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) - 2021*, Lviv, Ukraine, September 21-25, 2021. P. 255-260.

12. Smirnov O., Kuznetsov A., Girzheva O., Kiian A., Nakisko O., Kuznetsova T. «Advanced Code-Based Electronic Digital Signature Scheme». *2020 IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020,* Kharkiv, 6 October 2020-9 October 2020, P. 358-362.

13. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». *CEUR Workshop Proceedings* Volume 2805, 2020, Pages *44-58.*

14. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». *International Journal of Computing*; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

15. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

16. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

17. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 122-131.

18. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 1-14.

19. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». *Lecture Notes in Networks and Systems*, vol 152. Springer, Cham. 2021, pp 66-84.

20. Smirnov O., Kuznetsov A., Onikiychuk A., Makushenko T., Anisimova O., Arischenko A. «Adaptive pseudo-random sequence generation for spread spectrum image steganography». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Ukraine, Kyiv, May 14-18. 2020. P. 161-165.

21. Smirnov O., Kuznetsov A., Kiian A., Babenko V., Perevozova I.,Chepurko I. «New Approach to the Implementation of Post-Quantum Digital Signature Scheme». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 166-171.

22. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

23. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

24. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

25. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». *International Journal of Computer Network and Information Security (IJCNIS)*. Vol. 12, No. 3, 2020. PP.33-43.

26. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 646-660.

27. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

28. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

29. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, Vol 2588, P. 215-227, 2019.

30. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

31. Smirnov, O., Kuznetsov, A., Kiian, A., Gorbenko, Y., Cherep, O., Bexhter L. «Code-based Pseudorandom Generator for the Post-Quantum Period», *2019 IEEE International Conference on Advanced Trends in Information Theory (IEEE ATIT 2019).* 18.12.19-20.12.19 Kyiv Ukraine. P. 204 – 209.

32. Smirnov, O., Kuznetsov, A., Nariezhnii, O., Stelnyk, S., Kokhanovska, T., Kuznetsova T., «Side Channel Attack on a Quantum Random Number Generator», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18 - 21 September 2019. P.713-718.

33. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019;* Metz; France; 18-21 September 2019. P. 707-712.

34. Smirnov, O., Kuznetsov, A., Stefanovych, O., Gorbenko, Y., Krasnobaev, V., Kuznetsova K. «Information Hiding Using 3D-Printing Technology», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.701-706.

35. Smirnov, O., Hu, Z., Vasiliu, Y., Sydorenko, V., Polishchuk, Y., «Abstract Model of Eavesdropper and Overview on Attacks in Quantum Cryptography Systems», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.399-405.

36. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

37. Smirnov, O., Kuznetsov, A., Kiian, A., Babenko, B., Zhosan, H., Prokopovych-Tkachenko, D., «Soft Decoding Method for Turbo-Productive Codes», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT 2019*, Lviv, Ukraine, 2-6 July, 2019, P. 129-134.

38. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in

Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

39. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

40. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

41. Smirnov, O., Kuznetsov, A., Kiian, A., Kuznetsova, K., Ivko, T., Prokopovych-Tkachenko, D., «Soft Decoding Based on Ordered Subsets of Verification Equations of Turbo-Productive Codes», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 873-884.

42. Smirnov A.A., Kuznetsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

43. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології,* 2024, № 13, с. 28-35.

44. Смірнов О.А. Козлов Я.О., Смірнова Т.В. «Дослідження застосування SIEM-систем для забезпечення кібербезпеки та захисту інформації». *II Міжнародна науково-практична Інтернет-конференція «Інновації та перспективні шляхи hозвитку інформаційних технологій (ІПШРІТ-2023)»* м.Черкаси 6 грудня 2023 року – Черкаси: ЧДТУ.– 2023. – С.251-252.

45. Козлов Я.О., Смірнова Т.В., Смірнов О.А. «Дослідження SIEM-систем для забезпечення кібербезпеки». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя*

кафедри кібербезпеки та програмного забезпечення, м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 26.

46. Козлов Я.О., Козірова Н.Л., Смірнов О.А. «Дослідження структури та принципу роботи SIEM-системи». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя кафедри кібербезпеки та програмного забезпечення,* м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 59.

47. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». *Системи управління, навігації та зв'язку*, 2023, вип. 2(72), С. 170-178.

48. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

49. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

50. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки антивірусного захисту файлових серверів.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 135-02 від 01.04.2024 року).

3 Мета та призначення розробки

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

4 Джерела розробки

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

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

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

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи кібербезпеки антивірусного захисту файлових серверів;

– 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, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

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

5.8.1 Обладнання

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

5.8.2 Мова програмування

Середовище Delphi 10.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

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

7 Перелік документів, що розробляються

- Структурна схема системи – 1 аркуш.
- Функціональна схема системи – 1 аркуш.
- Діаграма процесів – 1 аркуш.
- Блок-схема алгоритму роботи програми – 2 аркуша.
- Пояснювальна записка – 100 аркушів.

8 Етапи розробки

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

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

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

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2024 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 4.06.2024 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

_____ Смірнов О.А.

Програмне забезпечення системи кібербезпеки антивірусного захисту файлових серверів

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 44

Літера: РП

Кропивницький – 2024 року

Файл Anti_Virus_For_File_Server_Options.pas – параметри антивірусу

```
unit Anti_Virus_For_File_Server_Options;

interface

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

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

```
    Bevel5: TBevel;
    Bevel3: TBevel;
    Bevel4: TBevel;
    optTabPC: TTabSheet;
    optPCLabel: TLabel;
    Bevel8: TBevel;
    PCAutoLoad: TCheckBox;
    PCAutoKill: TCheckBox;
    PCAutoAction: TCheckBox;
    PCDelInfect: TRadioButton;
    PCSkipInfect: TRadioButton;
    optPCInfoLabel: TLabel;
    SHOWBALOONHINT: TCheckBox;
    procedure ApplyBTNClick(Sender: TObject);
    procedure optTabOtherShow(Sender: TObject);
    procedure optTabFilterShow(Sender: TObject);
    procedure optTabPathesShow(Sender: TObject);
    procedure optTabModulesShow(Sender: TObject);
    Procedure SaveOptions;
    procedure CanselBTNClick(Sender: TObject);
    procedure APIListDblClick(Sender: TObject);
    procedure AddBTNClick(Sender: TObject);
    procedure DelBTNClick(Sender: TObject);
    procedure EditBTNClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure EditSaveReportBTNClick(Sender: TObject);
    procedure FileTAddAction(key, name, display, action: String);
    procedure FileTDelAction(key, name: String);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure optTabMainShow(Sender: TObject);
    procedure ChangeReg(StrName: ShortString; delete: boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  OptionsForm: TOptionsForm;

implementation

uses Anti_Virus_For_File_Server_Main, uPluginInfo,
Anti_Virus_For_File_Server_AddPath, uSelDir, uHideForm;

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

procedure TOptionsForm.FileTDelAction(key, name: String);
  var
```

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

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

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

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

Procedure TOptionsForm.SaveOptions;
  var
  i:integer;
begin
  if AUTORUN.Checked then
  begin
    ChangeReg('Virus_Anti_Virus_For_File_Server_Scanner',False);
  end else
  begin
    ChangeReg('Virus_Anti_Virus_For_File_Server_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_ANTI_VIRUS_FOR_FILE_SERVER_SCAN_SUBDIR         := SCNSUBDIR.Checked;
        OPT_USE_HEX_MODE        := SCNHEX.Checked;
        OPT_USE_CRC_MODE        := SCNCRC.Checked;
        OPT_USE_HEX_INPOS       := SCNHEXINPOS.Checked;
        OPT_SEND_ANTI_VIRUS_FOR_FILE_SERVER_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('*','Anti_Virus_For_File_Server.Anti_Virus_For_File_Server_Scan',
MainForm.SysMenu,ParamStr(0)+' %1');

FileTAddAction('Directory','Anti_Virus_For_File_Server.Anti_Virus_For_File_Serve
r_Scan',MainForm.SysMenu,ParamStr(0)+' %1');
```

```
FileTAddAction('Drive','Anti_Virus_For_File_Server.Anti_Virus_For_File_Server_Sc
an',MainForm.SysMenu,ParamStr(0)+' %1');
  end else
  begin

FileTDelAction('Drive','Anti_Virus_For_File_Server.Anti_Virus_For_File_Server_Sc
an');

FileTDelAction('Directory','Anti_Virus_For_File_Server.Anti_Virus_For_File_Serve
r_Scan');

FileTDelAction('*','Anti_Virus_For_File_Server.Anti_Virus_For_File_Server_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 Anti_Virus_For_File_Server_Monitor.pas – монітор (контроль процесів)

```
unit Anti_Virus_For_File_Server_Monitor;

interface

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

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

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

uses Anti_Virus_For_File_Server_Main, Anti_Virus_For_File_Server_InfectedAction,
Anti_Virus_For_File_Server_Options;
//***Функція створення параметрів сканування***//

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

```
begin
  inherited CreateParams(Params);
  with params do
    ExStyle := ExStyle or WS_EX_APPWINDOW;
end;

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

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

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

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

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

```
      PIDArray[I]);
    if (hProcess <> 0) then
    begin
      EnumProcessModules(hProcess, @hMod, SizeOf(hMod), cb);
      GetModuleFilenameEx(hProcess, hMod, ModuleName, SizeOf(ModuleName));
      if FileExists(ModuleName) then
        List.Add(ModuleName);
      CloseHandle(hProcess);
    end;
  end;
end;

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

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

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

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

  hProcess:=OpenProcess(PROCESS_TERMINATE,false,KillProc);
  if hProcess = 0 then
  begin
    cbPriv:=SizeOf(PrivOld);
    OpenThreadToken(GetCurrentThread,TOKEN_QUERY or
TOKEN_ADJUST_PRIVILEGES,false,hToken);
    OpenProcessToken(GetCurrentProcess,TOKEN_QUERY or
TOKEN_ADJUST_PRIVILEGES,hToken);
    Priv.PrivilegeCount:=1;
```

```
    Priv.Privileges[0].Attributes:=SE_PRIVILEGE_ENABLED;

LookupPrivilegeValue(nil,'SeAnti_Virus_For_File_Server_ProjectPrivilege',Priv.Pr
ivileges[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 := _Anti_Virus_For_File_Server_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 := _Anti_Virus_For_File_Server_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 Anti_Virus_For_File_Server_Project.dpr – головний файл проекту

```
program Anti_Virus_For_File_Server_Project;
// Список підключаємих модулів
uses
  Forms,
  SysUtils,
  Anti_Virus_For_File_Server_Kernel in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner
Modues\Anti_Virus_For_File_Server_Kernel.pas',
  Anti_Virus_For_File_Server_Types in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner
Modues\Anti_Virus_For_File_Server_Types.pas',
  avMonitor in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avMonitor.pas',
  avVirus_Anti_Virus_For_File_Server_Scanner in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner
Modues\avVirus_Anti_Virus_For_File_Server_Scanner.pas',
  avHex in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avHex.pas',
  avDataBase in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avDataBase.pas',
  avHash in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avHash.pas',
  avExt in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avExt.pas',
  avAPI in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avAPI.pas',
  avConfig in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avConfig.pas',
  avShield in '..\Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner Modues\avShield.pas',
  langs in 'langs.pas',
  Anti_Virus_For_File_Server_Main in 'Anti_Virus_For_File_Server_Main.pas'
{MainForm},
  uSelInfo in 'uSelInfo.pas' {InformationForm},
  Anti_Virus_For_File_Server_Options in 'Anti_Virus_For_File_Server_Options.pas'
{OptionsForm},
  uPluginInfo in 'uPluginInfo.pas' {PluginAPIForm},
  Anti_Virus_For_File_Server_AddPath in 'Anti_Virus_For_File_Server_AddPath.pas'
{AddUserPathForm},
  Anti_Virus_For_File_Server_About in 'Anti_Virus_For_File_Server_About.pas'
{AboutForm},
  uSelDir in 'uSelDir.pas' {SelDirFrm},
  uMessage in 'uMessage.pas' {MessageFrm},
  uHideForm in 'uHideForm.pas' {HideForm},
  Anti_Virus_For_File_Server_Monitor in 'Anti_Virus_For_File_Server_Monitor.pas'
{MonitorForm},
  Anti_Virus_For_File_Server_InfectedAction in
'Anti_Virus_For_File_Server_InfectedAction.pas' {ActionForm},
  uSplash in 'uSplash.pas' {SplashForm};

{$R *.res}

begin
  Application.Initialize;
  Application.Title := 'Virus_Anti_Virus_For_File_Server_Scanner';
  Application.CreateForm(TMainForm, MainForm);
  Application.CreateForm(TInformationForm, InformationForm);
  Application.CreateForm(TOptionsForm, OptionsForm);
  Application.CreateForm(TPluginAPIForm, PluginAPIForm);
  Application.CreateForm(TAddUserPathForm, AddUserPathForm);
  Application.CreateForm(TAboutForm, AboutForm);
  Application.CreateForm(TSelDirFrm, SelDirFrm);
  Application.CreateForm(TMessageFrm, MessageFrm);
  Application.CreateForm(THideForm, HideForm);
  Application.CreateForm(TMonitorForm, MonitorForm);
  Application.CreateForm(TActionForm, ActionForm);
```

```
  Application.CreateForm(TSplashForm, SplashForm);
  {Show Splash form}
  SplashForm.CRLabel.Caption   := 'Kernel '+GetKernelVersion;
  SplashForm.CRLabel00.Caption := 'Build ' +GetKernelBuild;
  SplashForm.Show;
  {}
  Init;
  langs.SwitchAllFormsToLng(01,01,ExtractFilePath(Paramstr(0))+'default.lng');
  {init kernel}
  MainForm.InitVirus_Anti_Virus_For_File_Server_ScannerKernel;
  {Hide Splash Form}
  SplashForm.Hide;
  Sleep(200);
  {Create Tray Icon}
  MainForm.CreateTray;
  {}
  if OptionsForm.AUTORUN.Checked then begin
    OptionsForm.ChangeReg('Virus_Anti_Virus_For_File_Server_Scanner',False);
  end else begin
    OptionsForm.ChangeReg('Virus_Anti_Virus_For_File_Server_Scanner',True);
  end;
  {}
  if ParamStr(1) <> '' then
  MainForm.StartAnti_Virus_For_File_Server_Scan(ParamStr(1));
  {}
  if OptionsForm.PCAutoLoad.Checked then begin
    MonitorForm.StartMonitor;
  end;
  {}
  Application.Run;
end.
```

Файл Anti_Virus_For_File_Server_Main.pas – основна програма

```
unit Anti_Virus_For_File_Server_Main;

interface

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

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

  type
  TIconType = (itSmall, itLarge);

  type
  NotifyIconData_50 = record
   cbSize: DWORD;
   Wnd: HWND;
   uID: UINT;
   uFlags: UINT;
   uCallbackMessage: UINT;
   hIcon: HICON;
   szTip: array[0..MAXCHAR] of AnsiChar;
   dwState: DWORD;
   dwStateMask: DWORD;
   szInfo: array[0..MAXBYTE] of AnsiChar;
   uTimeout: UINT; // union with uVersion: UINT;
   szInfoTitle: array[0..63] of AnsiChar;
   dwInfoFlags: DWORD;
  end;

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

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

type
  TMainForm = class(TForm)
    MainPages: TPageControl;
    Anti_Virus_For_File_Server_ScanPathesTab: TTabSheet;
    Anti_Virus_For_File_Server_ScanningTab: TTabSheet;
    ReportTab: TTabSheet;
    BottomPanel: TPanel;
    Anti_Virus_For_File_Server_ScanBTN: TButton;
    SaveBTN: TButton;
    PathList: TListView;
    Bevel1: TBevel;
    Anti_Virus_For_File_Server_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;
    mnuShowAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner:
TMenuItem;
    mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner:
TMenuItem;
    N2: TMenuItem;
    mnAnti_Virus_For_File_Server_Options: TMenuItem;
    N4: TMenuItem;
    mnuHelp: TMenuItem;
    mnuAbout: TMenuItem;
    N7: TMenuItem;
    mnuExit: TMenuItem;
    Image1: TImage;
    TopPn: TPanel;
    Bevel3: TBevel;
    Image2: TImage;
    RightPanel: TPanel;
    ExitBTN: TButton;
    TopRightPanel: TPanel;
    Image3: TImage;
    VersionLabel: TLabel;
    AboutBTN: TLabel;
    DelAll: TMenuItem;
    ApplicationEvents: TApplicationEvents;
    ProgressBar: TProgressBar;
    Anti_Virus_For_File_Server_ScanTopBtn: TLabel;
    Anti_Virus_For_File_Server_ScanMenu: TPopupMenu;
    mnuSelAnti_Virus_For_File_Server_ScanPath: TMenuItem;
    mnuShowReport: TMenuItem;
    N12: TMenuItem;
    OptionTopBtn: TLabel;
    PCTopBtn: TLabel;
    mnuAnti_Virus_For_File_ServerProcessControl: TMenuItem;
    N19: TMenuItem;
    mnuPCShow: TMenuItem;
    N21: TMenuItem;
    mnuPCRun: TMenuItem;
    mnuPCPause: TMenuItem;
    mnuPCStop: TMenuItem;
    mnuAnti_Virus_For_File_Server_ScanStart: TMenuItem;
    mnuStopAnti_Virus_For_File_Server_Scan: TMenuItem;
    N13: TMenuItem;
    mnuSaveReport: TMenuItem;
    N26: TMenuItem;
    mnuGoToTray: TMenuItem;
    SOURCESTRING: TListBox;
    LabelPanel: TPanel;
    Anti_Virus_For_File_Server_ScanFile: TLabel;
    procedure DelAllClick(Sender: TObject);
    procedure FormResize(Sender: TObject);
    procedure ExitBTNClick(Sender: TObject);
    procedure Anti_Virus_For_File_Server_ScanListDblClick(Sender: TObject);
    procedure Anti_Virus_For_File_Server_ScanBTNClick(Sender: TObject);
    procedure InitVirus_Anti_Virus_For_File_Server_ScannerKernel;
    Procedure StartAnti_Virus_For_File_Server_Scan(Parametr: String);
    procedure SaveBTNClick(Sender: TObject);
    procedure DeletePathClick(Sender: TObject);
    procedure RefteshClick(Sender: TObject);
    procedure AddFolderClick(Sender: TObject);
    function CreateDrivesList(ListView: TListView): boolean;
    procedure AboutBTNClick(Sender: TObject);
```

```
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure HelpBTNClick(Sender: TObject);
    procedure DelMenuPopup(Sender: TObject);
    procedure DelClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormDestroy(Sender: TObject);
    procedure FormHide(Sender: TObject);
    procedure
mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerClick(S
ender: TObject);
    procedure
mnuShowAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerClick(S
ender: TObject);
    procedure mnuExitClick(Sender: TObject);
    procedure mnAnti_Virus_For_File_Server_OptionsClick(Sender: TObject);
    procedure mnuHelpClick(Sender: TObject);
    procedure mnuAboutClick(Sender: TObject);
    procedure ApplicationEventsMinimize(Sender: TObject);
    procedure AppMinimize(Sender: TObject);
    procedure FormPaint(Sender: TObject);
    procedure Anti_Virus_For_File_Server_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 Anti_Virus_For_File_Server_ScanTopBtnClick(Sender: TObject);
    procedure mnuShowReportClick(Sender: TObject);
    procedure mnuSelAnti_Virus_For_File_Server_ScanPathClick(Sender: TObject);
    procedure PCTopBtnClick(Sender: TObject);
    procedure OptionTopBtnClick(Sender: TObject);
    procedure mnuGoToTrayClick(Sender: TObject);
    procedure mnuPCShowClick(Sender: TObject);
    procedure mnuPCRunClick(Sender: TObject);
    procedure mnuPCPauseClick(Sender: TObject);
    procedure mnuPCStopClick(Sender: TObject);
    procedure TrayMenuPopup(Sender: TObject);
    procedure Anti_Virus_For_File_Server_ScanMenuPopup(Sender: TObject);
    procedure mnuAnti_Virus_For_File_Server_ScanStartClick(Sender: TObject);
    procedure mnuStopAnti_Virus_For_File_Server_ScanClick(Sender: TObject);
    procedure mnuSaveReportClick(Sender: TObject);
    procedure CopyRightLabelClick(Sender: TObject);
    Procedure CreateTray;
  protected
    procedure MineRestore(var Msg: TMessage); message WM_MINERESTORE;
    procedure SendAnti_Virus_For_File_Server_Scanning(var Msg: TMessage);
message WM_COPYDATA;
  private
    Procedure WMSysCommand(var message: TWMSysCommand); message WM_SysCommand;
    procedure WMTRAYICONNOTIFY(var Msg: TMessage); message WM_NOTIFYTRAYICON;
    { Private declarations }
  public
    FileCN        : Integer;
    FileInfected  : Integer;
    FileIgnored   : Integer;
    FileDVC       : integer;

    MonFileCN        : Integer;
    MonFileInfected  : Integer;

    Path          : TStringList;
    DeActiveTray  : Boolean;

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

        Anti_Virus_For_File_ServerMonitor      : String;
        Anti_Virus_For_File_ServerInit         : String;
        LoadAPI         : String;
```

```
        LoadDB          : String;
        CreateDrvList   : String;
        OptFileNotFnd   : String;
        LoadOptFile     : String;
        InitProcedures  : String;
        initShield      : String;
        ErrorInit       : String;
        LogBevel        : String;
        DBKnowledge     : String;
        SCNOBJ          : String;
        Anti_Virus_For_File_Server_ScanExecute     : String;
        Anti_Virus_For_File_Server_ScanEnd         : String;
        PrepareToAnti_Virus_For_File_Server_Scan   : String;
        FileIgnor       : String;
        FileIfect       : String;
        FileAnti_Virus_For_File_Server_Scanned     : String;
        DataAnti_Virus_For_File_Server_Scanned     : String;
        IGNORED         : String;
        SKIPBYSIZE      : String;
        INFECTED        : String;
        STOPB           : String;
        RETURNB         : String;
        ANTI_VIRUS_FOR_FILE_SERVER_SCANB           : String;
        SCNFILE         : String;
        FileDel         : String;
        FileNotDel      : String;
        PATHNOSEL       : String;
        SysMenu         : String;
        NfoAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner
: String;
        NfoAnti_Virus_For_File_ServerKernel    : String;
        NfoAnti_Virus_For_File_ServerBuild     : String;
        DelDialog       : String;
        DelAllDialog    : String;
        DelError        : String;
        HelpNOFound     : String;
        avShieldMes     : String;
        avError         : String;
        DelResult       : String;
        AllInfected     : String;
        DeleteInfected  : String;
        SkippedInfected : String;
        Anti_Virus_For_File_ServerCloseDlg     : String;
        AllreadyInAnti_Virus_For_File_Server_Scan  : String;
        ProcControlSt   : String;
        ErrorKillProc   : String;
        PCActive        : String;
        PCPaused        : String;
        PCStoped        : String;
        PCInit          : String;
        PCPause         : String;
        PCStop          : String;
        PCRestore       : String;
        LASTDBDATA      : String;
        DATABASEdate    : String;
        BASELOADED      : String;
        DBerrorI1       : String;
        DBerrorI2       : String;
        DBerrorI3       : String;

        MLoad           : String;
        MunLoad         : String;

  end;

//*****************************************************************************//
// Створення головної форми
resourcestring
  Return          = #13#10;
```

```
  Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerCapt  =
'Антивірусний захист операційної системи від шкідливих програм';
  Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerVS    = '';
var
  MainForm      : TMainForm;
  inAnti_Virus_For_File_Server_Scan        : Boolean = False;
  NeedToReturn  : Boolean = False;
  FirstRun      : Boolean = True;
  P             : TPoint;
  MayClose      : boolean=false;
implementation

uses uSelInfo, Anti_Virus_For_File_Server_Options,
Anti_Virus_For_File_Server_AddPath, Anti_Virus_For_File_Server_About, Math,
uMessage, uHideForm,
  Anti_Virus_For_File_Server_Monitor, Anti_Virus_For_File_Server_InfectedAction,
uPluginInfo;
{$R *.dfm}

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

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

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

procedure TMainForm.SendAnti_Virus_For_File_Server_Scanning;
var
  pcd: PCopyDataStruct;
begin
  pcd := PCopyDataStruct(Msg.LParam);
  if not inAnti_Virus_For_File_Server_Scan then
  begin
    StartAnti_Virus_For_File_Server_Scan(PChar(pcd.lpData));
  end
  else begin
    MessageDlg(AllreadyInAnti_Virus_For_File_Server_Scan,mtError,[mbOK],0);
  end;
end;

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

mnuShowAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Click;
  end;
end;

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

function TMainForm.BalloonTrayIcon(const Window: HWND; const IconID: Byte; const
Timeout: TBalloonTimeout; const BalloonText, BalloonTitle: String; const
BalloonIconType: TBalloonIconType): Boolean;
  const
  aBalloonIconTypes : array[TBalloonIconType] of
                      Byte = (NIIF_NONE, NIIF_INFO, NIIF_WARNING, NIIF_ERROR);
  var
  NID_50 : NotifyIconData_50;
begin
  if Not OptionsForm.SHOWBALOONHINT.Checked then Exit;
  FillChar(NID_50, SizeOf(NotifyIconData_50), 0);
  with NID_50 do begin
    cbSize := SizeOf(NotifyIconData_50);
    Wnd := Window;
```

```
    uID := IconID;
    uFlags := NIF_INFO;
    StrPCopy(szInfo, BalloonText);
    uTimeout := Timeout * 1000;
    StrPCopy(szInfoTitle, BalloonTitle);
    dwInfoFlags := aBalloonIconTypes[BalloonIconType];
  end;
  Result := Shell_NotifyIcon(NIM_MODIFY, @NID_50);
end;

procedure TMainForm.WMTRAYICONNOTIFY(var Msg: TMessage);
begin
  case Msg.LParam of
    WM_LBUTTONUP:
      begin
      if Not DeActiveTray then
        begin
          MayClose := False;
          GetCursorPos(p);
          MayClose:= false;
          DeActiveTray := False;
          showwindow(Application.handle, SW_SHOW);
          showwindow(MainForm.handle, SW_SHOW);
          Application.Restore;
        end
        else
        begin
          SetForegroundWindow(HideForm.Handle);
        end;
      end;
    WM_RBUTTONUP:
      begin
        if Not DeActiveTray then
        begin
          GetCursorPos(p);
          TrayMenu.Popup(P.X, P.Y);
        end;
      end;
  end;
end;

Procedure TMainForm.CreateTray;
  var
  tray: TNotifyIconData;
begin
  with tray do
  begin
    cbSize := SizeOf(TNotifyIconData);
    Wnd := MainForm.Handle;
    uID := 1;
    uFlags := NIF_ICON or NIF_MESSAGE or NIF_TIP;
    uCallBackMessage := WM_NOTIFYTRAYICON;
    hIcon :=  Application.Icon.Handle;
    szTip := 'Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_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_ANTI_VIRUS_FOR_FILE_SERVER_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_ANTI_VIRUS_FOR_FILE_SERVER_SCAN_FILE;

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

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

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

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

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

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

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

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

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

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

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

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

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

      if GETParamName(Anti_Virus_For_File_ServerConfig[i]) = 'AUTOSAVEREPORTTO'
then OptionsForm.ReportSavePath.Text :=
GETParam(Anti_Virus_For_File_ServerConfig[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.Anti_Virus_For_File_Server_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.Anti_Virus_For_File_Server_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 OnAnti_Virus_For_File_Server_ScanComplete;
  var
  Anti_Virus_For_File_Server_ScanEndBalloonText: String;
  i: integer;
begin
  MainForm.ProgressBar.Max := 1;
  MainForm.ProgressBar.Position := MainForm.ProgressBar.Max;
  MainForm.Anti_Virus_For_File_Server_ScanBTN.Caption := MainForm.RETURNB;
  NeedToReturn := True;
  inAnti_Virus_For_File_Server_Scan := False;
  MainForm.Path.Clear;

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

  MessageBeep(MB_ICONASTERISK);
  MainForm.SaveBTN.Enabled := true;
  MainForm.Anti_Virus_For_File_Server_ScanFile.caption  :=
MainForm.Anti_Virus_For_File_Server_ScanEnd;
  OnAddToLogStr('',-1);
  OnAddToLogStr(MainForm.Anti_Virus_For_File_Server_ScanEnd,0);
  OnAddToLogStr('',-1);

OnAddToLogStr(MainForm.FileAnti_Virus_For_File_Server_Scanned+inttostr(MainForm.
FileCN),0);
  OnAddToLogStr(MainForm.FileIgnor+inttostr(MainForm.FileIgnored),0);
  OnAddToLogStr(MainForm.FileIfect+inttostr(MainForm.FileInfected),0);

OnAddToLogStr(MainForm.DataAnti_Virus_For_File_Server_Scanned+Format('%.2f',[Ant
i_Virus_For_File_Server_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;
```

```
  Anti_Virus_For_File_Server_ScanEndBalloonText :=
MainForm.Anti_Virus_For_File_Server_ScanEnd + ':' + Return + Return
                        +' >>
'+MainForm.FileAnti_Virus_For_File_Server_Scanned+inttostr(MainForm.FileCN) +
Return
                        +' >> '+MainForm.FileIgnor+inttostr(MainForm.FileIgnored)
+ Return
                        +' >>
'+MainForm.FileIfect+inttostr(MainForm.FileInfected) + Return
                        +' >>
'+MainForm.DataAnti_Virus_For_File_Server_Scanned+Format('%.2f',[Anti_Virus_For_
File_Server_ScannedDataSize / 1024 / 1024])+' Mb';

  MainForm.BalloonTrayIcon(MainForm.Handle
,1,10,Anti_Virus_For_File_Server_ScanEndBalloonText,'Anti_Virus_For_File_Server
Virus_Anti_Virus_For_File_Server_Scanner',bitInfo);
end;

//***Функція початку сканування***//

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

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

  MainForm.Anti_Virus_For_File_Server_ScanBTN.Caption := MainForm.STOPB;
  MainForm.SaveBTN.Enabled := False;
  MainForm.Anti_Virus_For_File_Server_ScanList.Clear;
  MainForm.Anti_Virus_For_File_Server_ScanningTab.Show;
  MainForm.FileCN       := 0;
  MainForm.FileInfected := 0;
  MainForm.FileIgnored  := 0;
  inAnti_Virus_For_File_Server_Scan := True;
  NeedToReturn := False;
  OnAddToLogStr(MainForm.Anti_Virus_For_File_Server_ScanExecute,0);
  if Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction
= TAnti_Virus_For_File_Server_ScanDir then
  else

OnAddToLogStr(MainForm.SCNOBJ+Anti_Virus_For_File_ServerVirus_Anti_Virus_For_Fil
e_Server_Scanner.FileName,0);
  OnAddToLogStr('',-1);
  MainForm.BalloonTrayIcon(MainForm.Handle
,1,10,MainForm.Anti_Virus_For_File_Server_ScanExecute,'Anti_Virus_For_File_Serve
r Virus_Anti_Virus_For_File_Server_Scanner',bitInfo);
  Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Resume;
end;

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

Procedure Anti_Virus_For_File_ServerKernelMessageAPI(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.Anti_Virus_For_File_Server_ScanBTN.Enabled := False;
  end;
```

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

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

  if MES = MES_PREPARINGTOANTI_VIRUS_FOR_FILE_SERVER_SCAN then
MainForm.Anti_Virus_For_File_Server_ScanFile.Caption :=
MainForm.PrepareToAnti_Virus_For_File_Server_Scan;

  if mes = MES_INITKERNEL then
OnAddToLogStr(MainForm.Anti_Virus_For_File_ServerInit,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_ONANTI_VIRUS_FOR_FILE_SERVER_SCANEXECUTE then
    OnAnti_Virus_For_File_Server_ScanStart;

  if MES = MES_ONANTI_VIRUS_FOR_FILE_SERVER_SCANCOMPLETE then
    OnAnti_Virus_For_File_Server_ScanComplete;

  if MES = MES_ONPROGRESS then begin
    if MainForm.ProgressBar.Enabled then begin
      MainForm.FileCN := MainForm.FileCN + 1;
      if MainForm.ProgressBar.Max > 0 then
        MainForm.ProgressBar.Position := MainForm.FileCN-MainForm.FileDVC;
      MainForm.Anti_Virus_For_File_Server_ScanFile.caption  :=
'['+inttostr(MainForm.FileCN)+'] '+ExtractFileName(Pr_1);
    end
    else
      MainForm.Anti_Virus_For_File_Server_ScanFile.caption  :=
ExtractFileName(Pr_1);
    if OPT_SEND_ANTI_VIRUS_FOR_FILE_SERVER_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 антивіруса***//

procedure TMainForm.InitVirus_Anti_Virus_For_File_Server_ScannerKernel;
  var
  i:integer;
begin

//******//
        Anti_Virus_For_File_ServerMonitor      := SOURCESTRING.Items[0];
        Anti_Virus_For_File_ServerInit         := 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];
        Anti_Virus_For_File_Server_ScanExecute     := SOURCESTRING.Items[13];
        Anti_Virus_For_File_Server_ScanEnd         := SOURCESTRING.Items[14];
        PrepareToAnti_Virus_For_File_Server_Scan   := SOURCESTRING.Items[15];
        FileIgnor        := SOURCESTRING.Items[16];
        FileIfect        := SOURCESTRING.Items[17];
        FileAnti_Virus_For_File_Server_Scanned     := SOURCESTRING.Items[18];
        DataAnti_Virus_For_File_Server_Scanned     := SOURCESTRING.Items[19];
        IGNORED          := SOURCESTRING.Items[20];
        SKIPBYSIZE       := SOURCESTRING.Items[21];
        INFECTED         := SOURCESTRING.Items[22];
        STOPB            := SOURCESTRING.Items[23];
        RETURNB          := SOURCESTRING.Items[24];
        ANTI_VIRUS_FOR_FILE_SERVER_SCANB           := SOURCESTRING.Items[25];
        SCNFILE          := SOURCESTRING.Items[26];
        FileDel          := SOURCESTRING.Items[27];
        FileNotDel       := SOURCESTRING.Items[28];
        PATHNOSEL        := SOURCESTRING.Items[29];
        SysMenu          := SOURCESTRING.Items[30];
        NfoAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner
:= SOURCESTRING.Items[31];
        NfoAnti_Virus_For_File_ServerKernel    := SOURCESTRING.Items[32];
        NfoAnti_Virus_For_File_ServerBuild     := 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];
        Anti_Virus_For_File_ServerCloseDlg     := SOURCESTRING.Items[44];
        AllreadyInAnti_Virus_For_File_Server_Scan  := SOURCESTRING.Items[45];
        ProcControlSt   := SOURCESTRING.Items[46];
        ErrorKillProc   := SOURCESTRING.Items[47];
        PCActive        := SOURCESTRING.Items[48];
        PCPaused        := SOURCESTRING.Items[49];
        PCStoped        := SOURCESTRING.Items[50];
        PCInit          := SOURCESTRING.Items[51];
        PCPause         := SOURCESTRING.Items[52];
        PCStop          := SOURCESTRING.Items[53];
        PCRestore       := SOURCESTRING.Items[54];
        LASTDBDATA      := SOURCESTRING.Items[55];
        DATABASEdate    := SOURCESTRING.Items[56];
        BASELOADED      := SOURCESTRING.Items[57];
        DBerrorI1       := SOURCESTRING.Items[58];
        DBerrorI2       := SOURCESTRING.Items[59];
        DBerrorI3       := SOURCESTRING.Items[60];

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

        InitKernel(Anti_Virus_For_File_ServerKernelMessageAPI);
        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)+'
'+NfoAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner
+Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerVS);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+NfoAnti_Virus_For_File_ServerKernel +GetKernelVersion);
  ReportMemo.Lines.Add(FormatDateTime('[hh:mm:ss]',now)+'
'+NfoAnti_Virus_For_File_ServerBuild +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('*','Anti_Virus_For_File_Server.Anti_Virus_For_File_S
erver_Scan',SysMenu,ParamStr(0)+' %1');

OptionsForm.FileTAddAction('Directory','Anti_Virus_For_File_Server.Anti_Virus_Fo
r_File_Server_Scan',SysMenu,ParamStr(0)+' %1');

OptionsForm.FileTAddAction('Drive','Anti_Virus_For_File_Server.Anti_Virus_For_Fi
le_Server_Scan',SysMenu,ParamStr(0)+' %1');
  end else
  begin

OptionsForm.FileTDelAction('Drive','Anti_Virus_For_File_Server.Anti_Virus_For_Fi
le_Server_Scan');

OptionsForm.FileTDelAction('Directory','Anti_Virus_For_File_Server.Anti_Virus_Fo
r_File_Server_Scan');

OptionsForm.FileTDelAction('*','Anti_Virus_For_File_Server.Anti_Virus_For_File_S
erver_Scan');
  end;
end;

//***Функція початку сканування***//

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

  if Parametr = 'DRV' then
  begin
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
TRUE;
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction
:= TAnti_Virus_For_File_Server_ScanDir;
    Path.Add(ExtractFileDrive(Paramstr(0))+'\');
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Dirs
:= Path;
    OnAnti_Virus_For_File_Server_ScanStart;
    exit;
  end;

  if DirectoryExists(Parametr+'\') then
  begin
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
TRUE;
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction
:= TAnti_Virus_For_File_Server_ScanDir;
    Path.Add(Parametr+'\');
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Dirs
:= Path;
    OnAnti_Virus_For_File_Server_ScanStart;
    exit;
```

```
  end;

  if FileExists(Parametr) then
  begin
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI :=
false;
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction
:= TAnti_Virus_For_File_Server_ScanFile;
    Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.FileName
:= Parametr;
    OnAnti_Virus_For_File_Server_ScanStart;
    exit;
  end;

end;

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

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

procedure TMainForm.Anti_Virus_For_File_Server_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 inAnti_Virus_For_File_Server_Scan = False then
    begin
```

```
      if PATH.Count-1 <> -1 then
      begin
        Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner :=
TAvVirus_Anti_Virus_For_File_Server_Scanner.Create(true);

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.FreeOnTermina
te := True;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.NeedForAPI
:= true;

Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.AvAction
:= TAnti_Virus_For_File_Server_ScanDir;
        Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Dirs
:= MainForm.Path;
        OnAnti_Virus_For_File_Server_ScanStart;
      end
      else begin
        MessageDlg(PATHNOSEL,mtError,[mbOk],0);
      end;
    end
    else begin
      CloseAnti_Virus_For_File_Server_ScanThread;
    end;
  end else
  begin
    Anti_Virus_For_File_Server_ScanBTN.Caption :=
Anti_Virus_For_File_Server_ScanB;
    MainForm.SaveBTN.Enabled := False;
    NeedToReturn := False;
    Anti_Virus_For_File_Server_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 Anti_Virus_For_File_Server_ScanList.Items.Count-1 do
      Report.Add(Anti_Virus_For_File_Server_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 :=
Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerVS;
end;

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

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

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

```
                        if
DeleteFileBC(Anti_Virus_For_File_Server_ScanList.Items.Item[i].SubItems[0]) then
          begin
            d:=d+1;

      Anti_Virus_For_File_Server_ScanList.Items.Item[i].ImageIndex := 4;

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

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

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

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

procedure TMainForm.FormCreate(Sender: TObject);
begin
  Path := TStringList.Create;
  TopPn.ControlStyle            := ControlStyle + [csOpaque];
  TopRightPanel.ControlStyle    := ControlStyle + [csOpaque];
  Caption                       :=
Anti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_ScannerCapt;
  TopPn.DoubleBuffered                  := true;
  TopRightPanel.DoubleBuffered     := true;
  PathList.DoubleBuffered            := true;
  Anti_Virus_For_File_Server_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;
  Anti_Virus_For_File_Server_ScanList.Columns.Items[0].Width :=
Anti_Virus_For_File_Server_ScanList.Width - 25;
end;

procedure
TMainForm.mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scan
nerClick(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.mnuShowAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scan
nerClick(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.mnAnti_Virus_For_File_Server_OptionsClick(Sender: TObject);
begin
  if not inAnti_Virus_For_File_Server_Scan then begin
    LoadOptions;
    OptionsForm.Show;
  end;
end;

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

```
end;

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

  try
    AboutForm.ShowModal;
  except
  end;
end;

procedure TMainForm.ApplicationEventsMinimize(Sender: TObject);
begin

mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Click;
end;

procedure TMainForm.FormPaint(Sender: TObject);
begin
  if FirstRun then
    if OptionsForm.AUTOHIDE.Checked then
    begin

mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Click;
    end;
  FirstRun := false;
end;

procedure TMainForm.Anti_Virus_For_File_Server_ScanListCustomDrawItem(Sender:
TCustomListView;
  Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
begin
  with Anti_Virus_For_File_Server_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.Anti_Virus_For_File_Server_ScanTopBtnClick(Sender: TObject);
begin

Anti_Virus_For_File_Server_ScanMenu.Popup(MainForm.Left+Anti_Virus_For_File_Serv
er_ScanTopBtn.Left+3,MainForm.Top+Anti_Virus_For_File_Server_ScanTopBtn.Top+38);
end;

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

procedure TMainForm.mnuSelAnti_Virus_For_File_Server_ScanPathClick(Sender:
TObject);
begin
  if not inAnti_Virus_For_File_Server_Scan then
    Anti_Virus_For_File_Server_ScanPathesTab.Show;
end;

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

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

procedure TMainForm.mnuGoToTrayClick(Sender: TObject);
begin

mnuHideAnti_Virus_For_File_ServerVirus_Anti_Virus_For_File_Server_Scanner.Click;
end;

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

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

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

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

procedure TMainForm.TrayMenuPopup(Sender: TObject);
begin
  mnuPCRun.Enabled := MonitorForm.StartPC.Enabled;
  mnuPCPause.Enabled := MonitorForm.PausePC.Enabled;
  mnuPCStop.Enabled := MonitorForm.StopPC.Enabled;
  if inAnti_Virus_For_File_Server_Scan then
mnAnti_Virus_For_File_Server_Options.Enabled := False else
mnAnti_Virus_For_File_Server_Options.Enabled := True;
end;

procedure TMainForm.Anti_Virus_For_File_Server_ScanMenuPopup(Sender: TObject);
begin
  mnuPCRun.Enabled := MonitorForm.StartPC.Enabled;
  mnuPCPause.Enabled := MonitorForm.PausePC.Enabled;
  mnuPCStop.Enabled := MonitorForm.StopPC.Enabled;
  mnuSaveReport.Enabled := SaveBTN.Enabled;
  if inAnti_Virus_For_File_Server_Scan then
mnuAnti_Virus_For_File_Server_ScanStart.Enabled := False else
mnuAnti_Virus_For_File_Server_ScanStart.Enabled := True;
  if inAnti_Virus_For_File_Server_Scan then
mnuStopAnti_Virus_For_File_Server_Scan.Enabled := True else
mnuStopAnti_Virus_For_File_Server_Scan.Enabled := False;
end;

procedure TMainForm.mnuAnti_Virus_For_File_Server_ScanStartClick(Sender:
TObject);
begin
  Anti_Virus_For_File_Server_ScanBTN.Click;
end;

procedure TMainForm.mnuStopAnti_Virus_For_File_Server_ScanClick(Sender:
TObject);
begin
  Anti_Virus_For_File_Server_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 Anti_Virus_For_File_Server_AddPath.pas – додавання шляхів сканування

```
unit Anti_Virus_For_File_Server_AddPath;

interface

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

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

var
  AddUserPathForm: TAddUserPathForm;

implementation

uses Anti_Virus_For_File_Server_Main, Anti_Virus_For_File_Server_Options,
uSelInfo;

{$R *.dfm}

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

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

procedure TAddUserPathForm.ShellTreeViewClick(Sender: TObject);
begin
  if DirectoryExists(ShellTreeView.Path+'\') then
    ApplyBTN.Enabled := True else
    ApplyBTN.Enabled := False;
end;

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

```
end;

end.
```

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

```
unit Anti_Virus_For_File_Server_InfectedAction;

interface

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

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

var
  ActionForm: TActionForm;

implementation

uses Anti_Virus_For_File_Server_Main, Anti_Virus_For_File_Server_Options;

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

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

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

end.
```

Файл Anti_Virus_For_File_Server_About.pas – довідка

```
unit Anti_Virus_For_File_Server_About;

interface

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

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

var
  AboutForm: TAboutForm;

implementation

uses Anti_Virus_For_File_Server_Main;

{$R *.dfm}

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

end.
```