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 діагностування у корпоративній мережі”

Виконав здобувач вищої освіти
IV курсу, групи КБ-20
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Щорба Р.В.
«____» _____ 2024 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Смірнов С.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

Щорбі Роману Віталійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі*

2. Керівник роботи *Смірнов Сергій Анатолійович, канд. техн. наук, доцент*
(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 діагностування у корпоративній мережі*

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 С.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2024 р.

Підпис здобувача
_____ Щорба Р.В.
(прізвище та ініціали)

АНОТАЦІЯ

Щорба Р.В. Програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі. 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 діагностування у корпоративній мережі.

Метою розробки є програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі.

Результат роботи – програмна реалізація системи кібербезпеки антивірусного діагностування у корпоративній мережі.

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, корпоративна мережа

ABSTRACT

Shchorba R.V. Software of the cyber security system of antivirus diagnostics in the corporate network. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this final qualification work for the first (bachelor) level of higher education, software is developed, which is intended for the cyber security system of antivirus diagnostics in the corporate network.

The purpose of the development is the software of the cyber security system of antivirus diagnostics in the corporate network.

The result of the work is the software implementation of the cyber security system of antivirus diagnostics in the corporate network.

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, corporate network

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БММ	–	база математичних моделей
ЕОМ	–	електрона обчислювальна машина
КВ	–	коефіцієнт варіації
КЗ	–	канал зв'язку
НСД	–	несанкціонований доступ
ОД	–	об’єкт діагностування
ПС	–	програмна середа
СВВ	–	система виявлення вторгнень
СеМО	–	експонентна мережа масового обслуговування
СМО	–	система масового обслуговування
СПД	–	система передачі даних

ВСТУП

Актуальність теми. Оскільки кіберзагрози продовжують розвиватися із загрозливою швидкістю, традиційним антивірусним рішенням важко встигати за ними. Саме тут вступають у гру антивірусні технології нового покоління, які пропонують новий рівень захисту ваших цифрових активів. У цій роботі ми розглянемо ключові функції та переваги антивірусних технологій наступного покоління, а також те, як вони можуть захистити вашу організацію від загроз, що постійно зростають.

Традиційне антивірусне програмне забезпечення вже давно є основним рішенням для захисту від відомих шкідливих програм і вірусів. Однак ці рішення в основному покладаються на виявлення на основі сигнатур, що означає, що вони можуть виявляти лише загрози, які були раніше ідентифіковані та додані до їхніх баз даних. Такий підхід робить організації вразливими до атак «нульового дня» та нових шкідливих програм, які ще не розпізнані постачальниками антивірусів.

Крім того, традиційне антивірусне програмне забезпечення часто споживає значні системні ресурси, що призводить до проблем з продуктивністю та негативного впливу на продуктивність. Потреба в більш ефективному та проактивному підході до кібербезпеки проклала шлях для антивірусних технологій наступного покоління.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі.

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

– Огляд існуючих систем антивірусного діагностування у корпоративній мережі.

– Дослідження системи кібербезпеки антивірусного діагностування у корпоративній мережі.

– Програмна реалізація системи кібербезпеки антивірусного діагностування у корпоративній мережі.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі антивірусного діагностування у корпоративній мережі.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі, 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

Система, яка розробляється у бакалаврській роботі, призначена для реалізації програмного забезпечення системи антивірусного діагностування у корпоративній мережі.

Антивірусні технології наступного покоління використовують передові методи та передові технології, щоб забезпечити комплексний захист від відомих і невідомих загроз. Ось деякі ключові особливості, які відрізняють їх від інших:

– Аналіз поведінки: антивірусні рішення наступного покоління використовують аналіз поведінки для моніторингу й аналізу поведінки файлів і процесів у режимі реального часу. Виявляючи підозрілі дії та аномалії, вони можуть виявляти та блокувати раніше невідомі шкідливі програми.

– Машинне навчання та ШІ: ці технології дозволяють антивірусам наступного покоління постійно навчатися та адаптуватися до нових загроз. Аналізуючи величезні масиви даних, вони можуть ідентифікувати шаблони та ознаки компрометації, підвищуючи точність виявлення.

– Ізольоване програмне середовище: антивірусні рішення нового покоління часто включають можливості ізольованого програмного середовища, що дозволяє ізолювати та запускати підозрілі файли в контрольованому середовищі. Ця техніка допомагає виявляти й аналізувати потенційно зловмисну поведінку без ризику для хост-системи.

Антивірусні технології нового покоління пропонують кілька переваг перед традиційними антивірусними рішеннями:

– Покращене виявлення загроз: поєднуючи кілька методів виявлення, антивірус наступного покоління забезпечує кращий захист як від відомих, так і від нових загроз.

– Зменшення кількості помилкових спрацьовувань: розширені можливості аналізу антивірусу наступного покоління зводять до мінімуму помилкові спрацьовування, гарантуючи, що законні файли та процеси помилково не позначаються як шкідливі.

– Покращена продуктивність: антивірусні рішення наступного покоління створені для мінімального впливу на системні ресурси, що дозволяє організаціям підтримувати продуктивність без шкоди для безпеки.

Однак під час впровадження антивірусу нового покоління важливо враховувати кілька факторів:

– Сумісність: забезпечте сумісність із наявною IT-інфраструктурою та програмним забезпеченням, щоб уникнути потенційних конфліктів або проблем із продуктивністю.

– Проактивний моніторинг. Незважаючи на те, що антивірус наступного покоління забезпечує розширений захист, дуже важливо доповнити його проактивним моніторингом і регулярними оновленнями безпеки, щоб випереджати нові загрози.

– Вибір постачальника: Вибір правильного постачальника антивірусу наступного покоління є критичним. Враховуйте такі фактори, як репутація, підтримка та здатність адаптуватися до нових загроз.

Антивірусні технології нового покоління знаменують значний прогрес у боротьбі з кіберзагрозами. Використовуючи такі інноваційні підходи, як поведінковий аналіз, машинне навчання та пісочниця, ці рішення пропонують покращений захист і вдосконалені можливості виявлення загроз. Оскільки ландшафт загроз продовжує розвиватися, застосування антивірусних технологій нового покоління стає обов’язковим для захисту цифрових активів вашої організації.

1.2 Область застосування

Областю застосування системи, яка розробляється у бакалаврській роботі, є корпоративна мережа.

Корпоративна мережа — це логічно відокремлена група комп’ютерів, маршрутизаторів та інших частин IT-інфраструктури, які функціонують поза традиційними межами Інтернету. Її часто називають інтранет. Термін інтранет описує мережу, яка, на відміну від Інтернету, призначена для доступу лише певної групи людей [1].

Існує безліч методів, які можна використати для побудови ексклюзивної комп’ютерної мережі компанії. Пристрої можна з’єднувати разом за допомогою криптографії та VPN через відкритий Інтернет. Абревіатура VPN розшифровується як Virtual Private Network – абстрактна мережа, яка об'єднує лише певні машини, які мають підключення до глобальної мережі Інтернет. Вона може бути недоступною з-за меж штаб-квартири компанії або без відповідних облікових даних [2]

Основною метою створення корпоративних мереж є забезпечення безпеки та надійності інформаційної системи управління компанією, системи обробки даних, передачі даних і зв’язку – може не знадобитися співробітникам використовувати програмне забезпечення з обмеженими даними, коли вони знаходяться поза робочим місцем. Крім того, розміщення додатків, які стосуються лише членів компанії виключно в корпоративних мережах, може бути набагато менш складним [3]. Корпоративні мережі використовують різні типи пристроїв LAN та WAN, а також широко доступні протоколи та стандарти зв’язку для надання таких функцій.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі, 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

Огляд програми AVG Internet Security

Це антивірус, що може похвастатися невеликим запитом системних параметрів, можна ставити 9 балів. Самі розроблювачі не дуже те розписують free версію. Ви скачаєте файл, що біля одного мегабайта, але не дивуйтеся, коли ви його запустите почнеться завантаження (130 Мб) на ваш ПК. Потім буде установка, у якій їсти можливість відключити тулбар та інші не потрібні опції, які намагається встановити антивірус. Наприкінці установки пропозиція вказати E-mail і зареєструватися зовсім безкоштовно. Також вам установиться гаджет на робочому столі.

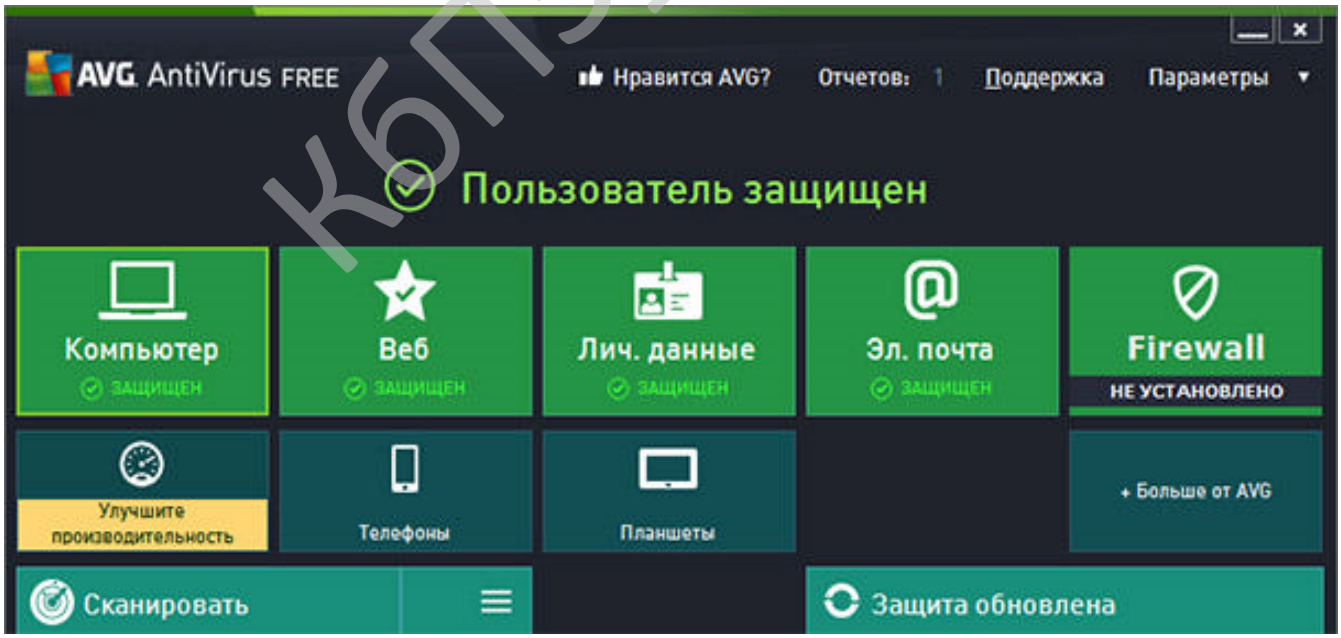

Рисунок 2.1 – Інтерфейс користувача AVG Internet Security

Під час роботи програма буде вимагати багато оперативної пам'яті (ОЗП), виправляється повним скануванням системи, після чого ненажерливість знижується в рази. Інтерфейс приємний, отут немає до чого при колупатися. Зробив відразу після установки сканування, пройшло швидко, але це на Windows 7 і на віртуальному диску.

Вирішив використати функцію поліпшення продуктивності, швидкість аналізу можна оцінити на 10, тому що деякі програми оптимізації роблять це набагато повільніше, але це швидше за все від різниці в зчитувальних параметрів, тут їх усього 4. І отут розчарування виправити не виходить, треба щось там десь качати. Навіщо тоді ця функція? Просто показати й розбудувати користувача? Пробував шукати сайти з вірусами, навіть на ті, де вони були раніше, він їх не знайшов. Може видалили. Простояв він на віртуальній машині рівно 1 тиждень і був видалений разом з віртуальним диском.

Огляд програми Avast Free Antivirus

От деякі основні можливості цього антивірусного ПЗ:

– Sandbox – Ізольоване віртуальне, тимчасово створюване середовище, у якій сумнівні сайти й додатки не можуть нанести яку-небудь шкоду вашому комп'ютеру.

– Антивірус – Захищає вашу систему від вірусів, шпигунського програм і інших шкідливих програм.

– Віддалена допомога – Дозволяє вашому другові допомогти вам у випадку виникнення проблем з комп'ютером.

– SafeZone – Чисте віртуальне вікно браузера запобігає крадіжці банківських даних.

– Антиспам – Якісний захист від спама.

– Брандмауер – Захищає ваші дані, блокуючи атаки зловмисників, досить таки непогано справляється.

Рисунок 2.2 – Інтерфейс користувача Avast Free Antivirus

Про цю антивірусну програму можна багато говорити й писати, Але для цього треба робити огляд по кожній антивірусній програмі окремо. А так я хочу лише винести свою думку щодо тих програм з якими доводилося працювати. Щоб ви мали подання, що й чим відрізняється друг від друга.

Більшим плюсом Avast є його можливість запускати все підозріле у віртуальній оболонці. За це я його й використовую, тому що ніщо із програм шкідливого характеру не зробить вам шкоди без вашого прямого на те згоди. Також із усього часу, що я проводжу в Інтернеті, він не раз мене застерігав від шкідливих і вірусних сайтів. Реально гарний безкоштовний захист. який зневажати не треба. Перше сканування системи буде виконано після перезапуску системи й у під системному режимі. Але це якщо ви не відключите цю можливість сканування. А систему реально вантажить у порівнянні з іншими програмами менше. Так що тверде 11 з 12 і моя рекомендація.

Огляд програми Avira Free Antivirus

Якщо ввести в ПК от такий запит avira vs avast, то співвідношення на користь Avast відчутно відразу. Але є й деякі плюси в Avira і ми зараз усе розглянемо. Звичайно розроблювачі постаралися зробити роботу програми більше компактної й не вимогливої до параметрів ПК. Але чесно працює аона більш повільніше за AVG. Давайте переглянемо основні можливості даної програми:

– Центр контролю для моніторингу, адміністрування й керування програмами.

– Централізоване налаштування в стандартному й експортному режимах із чутливої до контексту Довідкою.

– Інтегрований в Windows Vista модуль керування обліковими записами користувачів (User Account Control) для виконання завдань, що вимагають прав адміністратора.

– Guard для постійного відстеження спроб доступу до файлів.

– Убудований менеджер карантину для ізоляції підозрілих файлів і роботи з ними.

– Захист від руткіт-програм дозволяє виявити ПЗ, сховане встановлене в системі (руткіт) (тільки для 32-битн. системи).

– Прямий доступ до докладної інформації про виявлені віруси й шкідливому ПЗ (Інтернет).

– Просте й швидке відновлення програми, файлу вірусних сигнатур (VDF), а також пошукового ядра за допомогою відновлення одним файлом і інкрементного VDF-відновлення з веб-сервера в Інтернет.

– Убудований Планувальник для планування одноразових або повторюваних завдань відновлення, перевірки та ін.

– Найвищий рівень виявлення вірусів і шкідливих програм, гарантируємий новою технологією пошуку (пошукове ядро) із застосуванням евристики.

– Розпізнавання всіх популярних типів архівів, включаючи вкладені, із застосуванням списків небезпечних розширень файлів.

– Висока продуктивність багатопоточної технології (одночасне сканування декількох файлів).

Список не маленький, але по якості, не все виправдує очікуване.

Огляд програми Comodo Internet Security

От одна із програм, що воістину мене здивувала. Справа в тому, що я не знав про неї, точніше не використовував ніколи, а отут довелося ставити й тестувати.

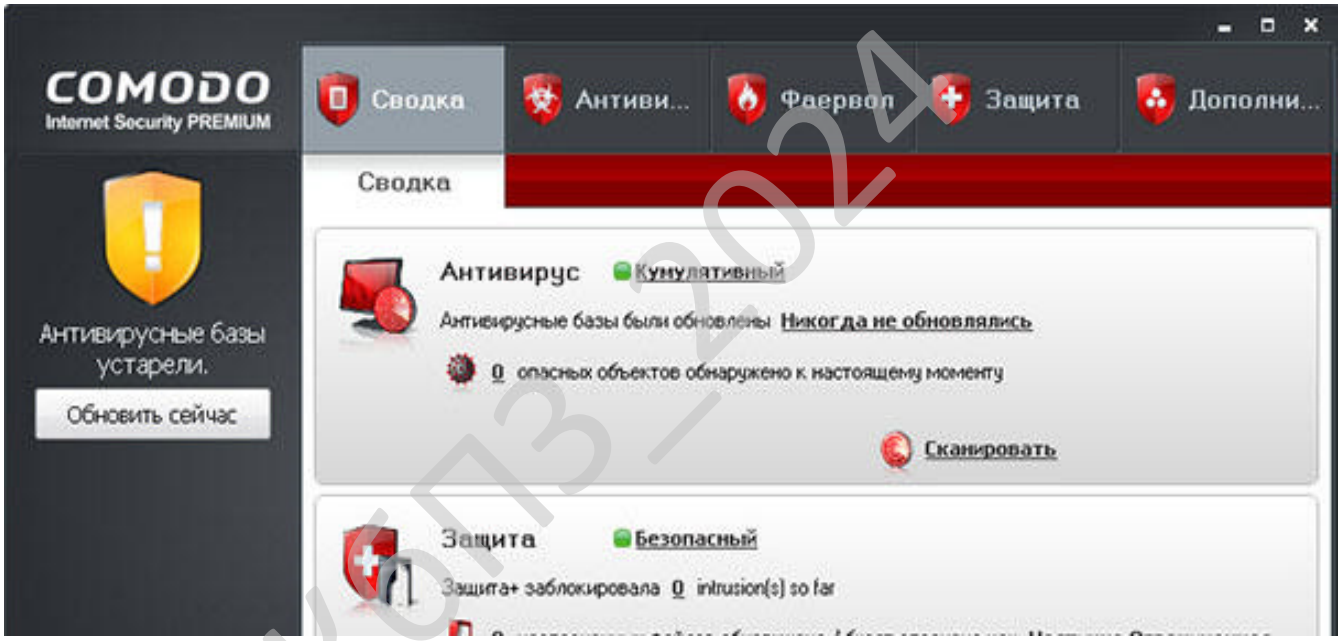

Рисунок 2.3 – Інтерфейс користувача Comodo Internet Security

Мені сподобалася дана продукції, вона як внутрішні файли добре сканує, так і зовнішні з мережі. В останньої 6 версії програми зовсім новий, поліпшений інтерфейс, що розрахований на сенсорні екрани в основному.

Користувався я нею десь пару тижнів, вона так і залишилася стояти на віртуальній машині. Легко ставиться на всі моделі Windows. Відразу після установки починається автоматичне відновлення, а потім відразу й сканування

системи. Взагалі меню програми дуже й дуже зручне. Є в Комоді така функція як ігровий режим, що немаловажно для вашої системи. І чим не може похвастатися кожна антивірусна програма. Є в ній і додаткові можливості.

Огляд утиліти Dr.Web LiveCD

Про цей диск варто знати кожному, навіть не просто знати, а мати його собі в колекції, так про всякий випадок.

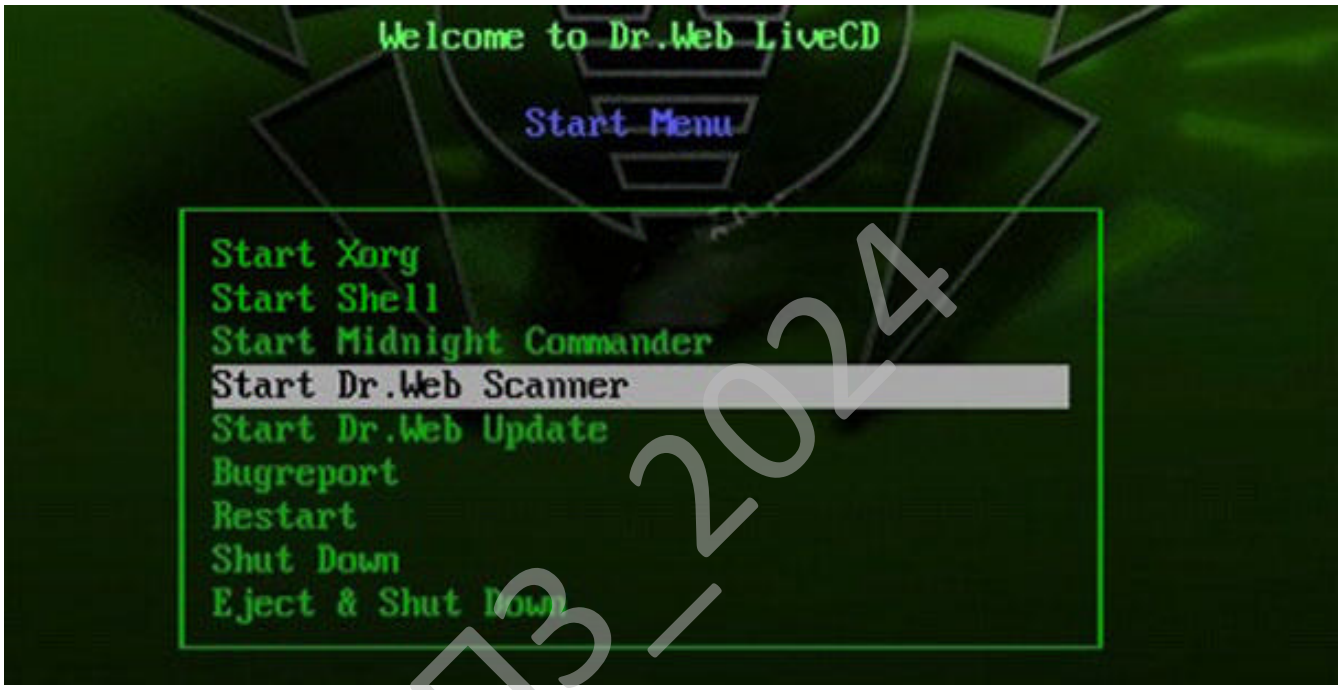

Рисунок 2.4 – Інтерфейс користувача Dr.Web LiveCD

Компанія "Доктори Веб" далеко ступнула розробивши дану утиліту й безсумнівно заробила не малу популярність і повагу. Ця утиліта може не тільки якісно від сканувати вашу систему, але й у випадку виходу з ладу вашої системи, через віруси, провести ряд реанімаційних робіт для відновлення й видалення останніх.

Працює вона із завантажувального диску, і для цього все що вам потрібно, це запустити систему із завантаженням з носія CD\DVD Rom. Ви можете не тільки перевіряти систему, але й у випадку непрацездатності Windows витягти корисні файли й перемістити на флеш накопичувач наприклад. Компанія

ступнула далеко вперед і тепер утиліту можна встановить на флеш накопичувач, тобто ви можете використовувати продукт на будь-якому флеш накопичувачі.

Dr.Web LiveUSB – продукт, що дозволяє провести аварійне відновлення операційної системи за допомогою завантажувального USB-накопичувача. Призначений для роботи в операційних системах Windows (32– і 64-бітні версії). Для завантаження із флеш-накопичувача BIOS комп'ютера повинен підтримувати пристрій USB-HDD у якості завантажувального.

Огляд програми Microsoft Security Essentials

Це буде остання програма по захисту ПК із серії безкоштовних.

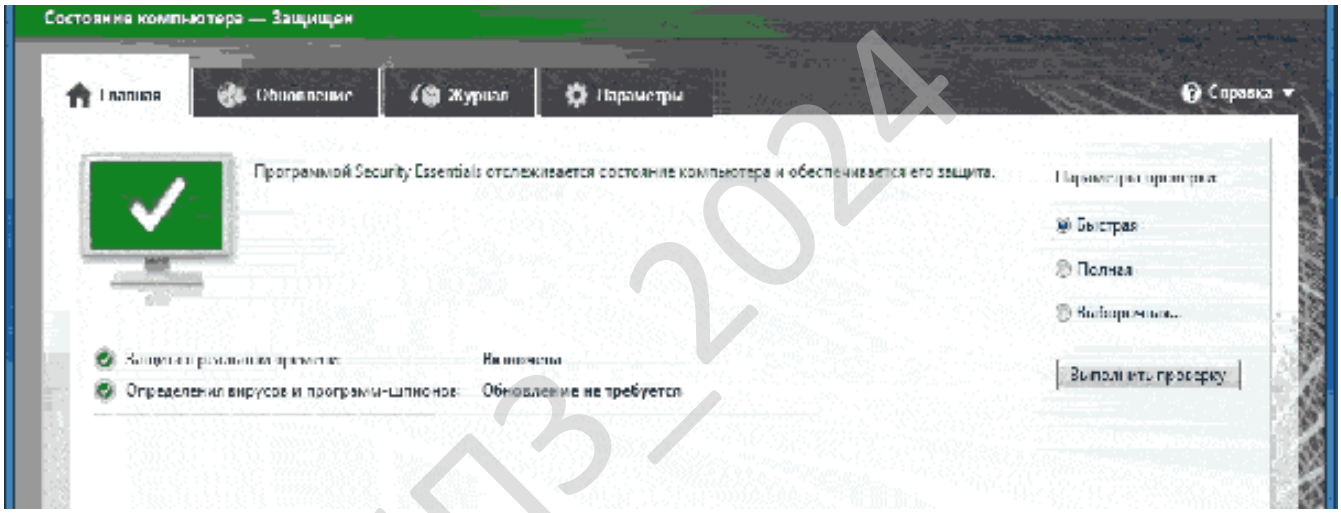

Рисунок 2.5 – Інтерфейс користувача Microsoft Security Essentials

У чому безсумнівна перевага цієї програми. Вона одна з найпростіших програм які я тільки бачив. Немає ніяких додаткових налаштувань і інструментів, якими напхані більшість програм по захисту комп'ютерів. Швидко встановлюється й так само швидко працює. Хоча при повному скануванні, час вона займе неймовірно багато. Особливим тут є її унікальна здатність зносити всілякі кряки з вашого ПК і кряк активації Windows у тому числі. Взагалі в мене зложилася думка, що компанія Microsoft приділила більше уваги виявленню не ліцензійного ПЗ, ніж захисту вашої системи.

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 діагностування у корпоративній мережі.

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

Більшість компаній думають про перемогу над потенційними атаками на безпеку, але лише деякі з них справді уявляють набір загроз безпеці, які можуть загрожувати компанії. Багато з них описані в корпоративних стандартах безпеки, таким чином допомагаючи компаніям організувати систему захисту IT-безпеки. У такому контексті антивірусний захист відіграє важливу роль у всій сфері безпеки. Крім того, сучасні антивірусні рішення стають більш досконалими та зрілими. Сьогодні вони включають не тільки антивірусний механізм для робочих станцій і консоль адміністрування, але й багато додаткових функцій, таких як антивірус для системи захисту пошти, шлюз, базу даних інцидентів і розширену систему звітів і журналів. Тим не менш, реалізація багатьох таких рішень далеко не вирішує всіх питань корпоративної безпеки. Тому недостатньо встановити лише персональні антивірусні продукти в корпоративній мережі, а цілий корпоративний пакет, щоб впоратися з усіма загрозами на різних рівнях мережі. Це допоможе побудувати корпоративне безпечне IT-середовище.

Що стосується майбутнього у сфері захисту безпеки корпоративних користувачів, зростає тенденція до включення більш складного інтерфейсу адміністрування, який надає детальну інформацію про стан мережі в реальному часі. Він може бути представлений у вигляді розширеного графічного інтерфейсу з діаграмами або навіть як окремий продукт. Це може бути інтелектуальний агент, який може обробляти величезну кількість інформації з тисяч комп’ютерів і підказує адміністратору, що робити в такому випадку.

Наприклад, Blue Medora розробила спеціального агента для корпоративного рішення Symantec, що забезпечує «меншу складність, більш уніфіковане управління операціями та значне скорочення витрат завдяки

усуненню надлишкової інфраструктури та кількох інструментів, що стосуються платформи» [2]. Це підтверджує думку про те, що є область для подальшого вдосконалення корпоративного антивірусного рішення навіть для видатного вендора.

Серед розширюваних функцій можна виділити наступні:

– Покращено моніторинг інцидентів зі шкідливим програмним забезпеченням.

– Покращено моніторинг вторгнення користувача в ключові процеси антивіруса.

– Моніторинг збоїв в оновленнях і завданнях сканування шкідливих програм.

Нові функції, які будуть включені в продукт:

– Монітор статусу та доступності в реальному часі.

– Монітори журналів.

– Система звітування та вжиття дій, яка допоможе адміністратору виконати необхідні дії щодо будь-якого типу загроз.

Основна ідея полягає в тому, щоб підвищити рівень чутливості осіб, які відповідають за безпеку корпоративної мережі, і скоротити час реакції на виниклу небезпеку. Таким чином, корисне та вичерпане представлення даних дійсно може допомогти у боротьбі зі шкідливим програмним забезпеченням.

Окрім реєстрації та моніторингу, важливою частиною рішення безпеки є цілісність. Сучасні корпоративні антивірусні рішення включають не тільки низку – антивірус, антишпигун, брандмауер, антиспам із системою керування, але й багато додаткових функцій, таких як системи резервного копіювання, менеджери паролів і ключів і утиліти шифрування для організації безпечного зберігання конфіденційних даних.

Що стосується корпоративних мереж, Symantec надає Protection Suite for Endpoints, де зібрано шифрування, зберігання конфіденційних даних та інші функції, спрямовані на підтримку IT-безпеки в компанії [5]. Ще один цікавий

продукт прийшов від Sophos [6]. Endpoint Security and Data Protection містить у своєму пакеті інтегровані інструменти DLP (запобігання втраті даних) і шифрування.

Крім того, в корпоративному рішенні повинні підтримуватися мобільні та не-Windows платформи через величезну різноманітність робочих пристроїв: ноутбуків, КПК, смартфонів і т.д. Багато виробників антивірусів мають такі рішення в лінійці продуктів.

Важливим моментом, який слід враховувати, є безпека як послуга. Безпека – це не тільки програмне забезпечення, а й стан системи. Важливо мати цілодобову службу технічної підтримки для вирішення найновіших проблем безпеки, таких як нові версії шкідливих програм, експлойти нульового дня. Часто проактивний захист не може впоратися з величезною різноманітністю нових модифікацій шкідливого програмного забезпечення, які щодня випускають генератори хакерів. Так само адміністратор не може підтримувати все програмне забезпечення в актуальному стані з установленими новими виправленнями. У такому контексті бажано розгортати систему пошуку вразливостей, щоб виявити порушення програмного забезпечення та вчасно повідомити про встановлення нових оновлень.

Тут порушується проблема якості служби підтримки. Ні для кого не секрет, що якісну службу підтримки може надати лише команда кваліфікованих експертів зі зловмисного програмного забезпечення, а не «пісочниці»-роботи. Багато компаній надають колонку аналізу шкідливих програм на своїх веб-сайтах або навіть в окремих доменах безпеки, де публікуються описи найпопулярніших загроз, як це робиться на virusradar.com від Eset.

Інша сторона медалі – здатність усунути наслідки інфекції. Не всі антивірусні механізми дозволяють належним чином знезаражувати систему чи мережу після інциденту, який уже стався. У цьому випадку аналітики випускають спеціальні утиліти та скрипти видалення, щоб допомогти адміністраторам очистити свої IT-ферми. Є такі сервіси від Symantec [7].

Фішинг стає серйозною проблемою для всіх користувачів у кіберсвіті. Що можуть запропонувати виробники антивірусів для захисту корпоративних користувачів від цієї проблеми, крім стандартних антифішингових модулів, які блокують небезпечні веб-сайти з чорного списку? Ці модулі можуть бути корисними для підтримки більш безпечного спілкування з фінансовими установами, що може бути важливим у корпоративному середовищі.

Нарешті, дотримання корпоративних стандартів безпеки – це те, що роблять деякі постачальники AV. Великі компанії намагаються організувати корпоративну IT-безпеку відповідно до політик, що відповідають стандартам безпеки. Серед них:

1) X509 – це стандарт ITU-T, який визначає формати для сертифікатів відкритих ключів, списків відкликаних сертифікатів, сертифікатів атрибутів і алгоритму перевірки шляху сертифікації [10],

2) LDAP (Lightweight Directory Access Protocol) – це прикладний протокол для запиту та зміни даних за допомогою служб каталогів, що працюють через TCP/IP [11],

3) Microsoft IWA (Integrated Windows Authentication) – забезпечує автентифікаційні з'єднання між Microsoft IIS, Internet Explorer та іншими додатками, що підтримують Active Directory [12].

У попередньому розділі був проведений аналіз вірусних програм та антивірусних засобів, а також відомих методів розробки антивірусних програм. Аналіз вірусних програм розкрив характерні особливості алгоритмів їх роботи та необхідність захисту персональних комп'ютерів від них. Дослідження антивірусних методів та засобів показало, що вони не забезпечують надійного захисту персональних комп'ютерів від вірусних програм, а також знешкодження тих з них, які не містяться в їх базах даних сигнатур.

Основними методами, які використовуються при роботі антивірусних програмних засобів є:

– сигнатурний пошук вірусного коду;

– емуляція роботи процесора;

– евристичний аналіз;

– трасування переривань.

Перераховані методи розробки антивірусних програм окремо, навіть їх комбінація, не дають можливості виявляти всі вірусні програми. В зв'язку з цим постала задача розробки нових методів, які дозволяли б виявляти та знешкоджувати вірусні програми, які з'являтимуться в майбутньому.

В роботі досліджувались вірусні програми, які в процесі розмноження записуються в початок, чи середину, чи кінець файлу, і крім цього можуть здійснювати шифрування свого тіла, чи запис у завантажувальний сектор жорсткого диску повністю або частково, чи маскування свого перебування в персональному комп'ютері корпоративної мережі, чи їх комбінацій.

На основі проведеного аналізу структур вірусних програм встановлено, що в процесі свого функціонування вони повинні виконувати певні обов'язкові дії, для реалізації яких розробляються такі модулі:

– активізації вірусу;

– забезпечення попадання в оперативний запам'ятовуючий пристрій;

– розмноження вірусу;

– шифрування тіла вірусу;

– маскування в персональному комп'ютері корпоративної мережі;

– виконання деструктивних дій.

Деякі з цих модулів можуть бути відсутніми. Тип вірусної програми залежить від наявності певних модулів в її складі.

З метою розробки математичних моделей вірусних програм проведено їх поділ за поведінкою при розмноженні та маскуванні в персональному комп'ютері корпоративної мережі. В залежності від можливих середовищ існування вірусних програм їх поділено на дві підмножини:

– До першої підмножини (позначимо через Q) віднесено ті з них, які заражують файли на жорсткому диску при розмноженні, впроваджуючись в них в початок, середину чи кінець, і при цьому містять сталі ділянки коду.

– До другої (позначимо через P) віднесено ті з них, які в процесі розмноження чи виконання змінюють свої ділянки коду повністю або частково, або маскують своє перебування в операційній системі, підставляючи замість заражених ділянок файлу незаражені, або записуються повністю чи частково у завантажувальний сектор жорсткого диску.

Для розробки математичних моделей антивірусних засобів розроблено математичні моделі типів вірусних програм згідно проведеного поділу. Нехай X – множина всіх файлів на жорсткому диску, включаючи також файл, що містить інформацію завантажувального сектора (позначимо його через x_0). Всі файли на жорсткому диску позначимо через x_i, i = 1, 2, 3,..., n, де n – кількість файлів, то $X = \{x_0, x_1,..., x_п\}$. Виділимо підмножину $X' \subset X$, $X' = \{x'_0, x'_1, x'_2,..., x'_k\}$, $к <= n$ усіх тих файлів (певного типу або декілька типів), які можуть бути заражені вірусними програмами, а також X'', $X'' = X' \setminus \{x'_0\}$. Нехай V – множина всіх вірусних програм, $V = \{v_1, v_2,..., v_m\}$, де $m \in N$, $m \to \infty$, $V = P \cup Q$. Множина V розбивається на класи в залежності від типів файлів, які заражують вірусні програми. Але оскільки в основу класифікації покладено алгоритм розмноження та алгоритм маскування в персональному комп'ютері корпоративної мережі, то дії вірусних програм по відношенню до файлів різних типів будуть однаковими. Тому недоцільно виділяти в класи по типу файлів. Математична модель роботи вірусних програм задається так:

$$M_1 = <X,V,G>,$$

де G – двохдольний інформаційний граф:

Граф G має множину вершин G_v, що складається з двох множин, які не перетинаються: підмножини V вершин, яка відповідає множині вірусних програм, і підмножини X вершин, що відповідає множині всіх файлів жорсткого диску. Дуги графа відображають зв'язок (алгоритм взаємодії) між множинами V і

X. Між вершиною двохдольного графа $v_i \epsilon V$ та інформаційною змінною v_i математичної моделі M_1 існує взаємнооднозначна відповідність. Аналогічно є взаємнооднозначна відповідність між вершиною графа $x_j \epsilon X$ та інформаційною змінною x_j. Таким чином, для двохдольного інформаційного графу математичної моделі M_1 справедливі співвідношення:

$$V \cup X = G_v,\ V \cap X = \varnothing.$$

Інформаційний граф є орієнтованим. Вірусна програма $v_i \epsilon V$ після її запуску здійснює пошук файлів з множини X для зараження. У випадку належності x_j множині X′ вірусна програма заражує x_j.

Враховуючи множини X', P, Q математична модель роботи вірусних програм запишеться так:

$$M_2 = <X', P, Q, G_1>,$$

де G_1 – граф зв'язків множин P і Q з множиною X'.

З метою виділення визначальних особливостей для ідентифікації вірусних програм множини P і Q представлено так:

$$P = \bigcup_{i=1}^{3} A_i,\quad Q = \bigcup_{i=4}^{10} A_i,$$

де A_i – множина вірусних програм, які визначаються однаковою поведінкою при розмноженні та маскуванні своїх дій в персональному комп'ютері корпоративної мережі. Множину предикатів, з допомогою яких в алгоритмі роботи вірусних програм реалізуються умовні переходи, позначимо через F_{A_i}, $i = \overline{1,10}$. Ознака коректності вірусних програм alg_i ($i = \overline{1,10}$) дорівнює True і некоректності – дорівнює False. Граф-схему алгоритму роботи вірусних програм та їх взаємодії з множиною X′ позначимо через S_{A_i} ($i = \overline{1,10}$). Математичну модель, що описує вірусні програми множини A_i на алгоритмічному рівні представлено так:

$$M_{A_i} = <X', A_i, F_{A_i}, S_{A_i}>,\ i = \overline{1,10}.$$

Для представлення роботи кожного типу вірусних програм використовувались скінчені орієнтовані зв'язні графи. Математично граф

однозначно відображається матрицею інцидентності, яку зручно представляти в пам’яті персонального комп’ютера.

Математичні моделі на алгоритмічному рівні $M_{A_i}(i = \overline{1,10})$ утворюють базу математичних моделей (БММ) вірусних програм. Вона містить достатньо складну та різнорідну інформацію і виникає проблема її внутрімашинного зображення.

3.2 Розробка структурної схеми

Структурна схема розробленої системи зображена на рисунку 3.1. Розроблена математична модель процесу антивірусного комбінованого діагностування у корпоративній мережі дає змогу аналізу цих ланцюгів символів з метою виявлення та виділення вірусних ділянок.

В процесі аналізу виділено такі етапи:

– лексичний аналіз;

– синтаксичний аналіз;

– евристичний аналіз;

– заповнення таблиць символів;

– знешкодження вірусних програм в ОД;

– втілення антивірусного корегуючого коду в ОД.

Входом модуля, що реалізує лексичний аналіз, служить ланцюг символів деякого алфавіту. Комбінації символів можуть розглядатись як єдині об’єкти. Робота лексичного аналізатора полягає в тому, щоб згрупувати певні термінальні символи в єдині синтаксичні об’єкти, які називаються лексемами. На етапі лексичного аналізу досліджується ланцюг лексем і встановлюється структура файлу. Лексичний аналізатор готує дані для синтаксичного аналізатора.

Результатом роботи синтаксичного аналізатора є дерево, яке представляє синтаксичну структуру, притаманну об’єкту діагностування.

Евристичний аналізатор вирішує питання про наявність вірусної програми у файлі шляхом аналізу результату роботи синтаксичного аналізатора та математичних моделей вірусних програм.

При знешкодженні вірусної програми в ОД здійснюється видалення її частин. Для цього використовується інформація евристичного аналізатора про тип та структуру вірусної програми.

Для втілення антивірусного корегуючого коду (далі антивірусного коду) в ОД здійснюється втілення ланцюга символів (лексем) певної структури в ланцюг символів (лексем) файлу з метою автоматизованого діагностування ОД і знешкодження вірусних програм у випадку їх виявлення.

Для ефективної організації роботи антивірусної програми на кожному етапі використовується таблиця символів для зберігання інформації.

Нехай всі символи певної мови програмування низького рівня складають множину S. Послідовності символів поставимо у взаємнооднозначну відповідність регулярну множину в алфавіті S. Порожній регулярній множині {e} в алфавіті S відповідає порожній ланцюг символів, тобто порожній файл. Будь-якому непорожньому файлу відповідає непорожня множина, що складається з символів файлу. Алфавіт S є скінченим. Довжина ланцюга символів дорівнює кількості символів в ній. Оскільки довжина файлу на жорсткому диску обмежена, то розглядатимемо лише скінченні регулярні множини. Регулярні множини в алфавіті S ототожнюються з регулярними виразами.

Розділимо кожен блок моделі на підблоки до необхідного рівня деталізації опису системи. Модель функціонально поділиться на підмоделі. Тобто проведемо декомпозицію системи і застосуємо структурний підхід для моделювання внутрішнього механізму кожного блоку. Математична модель відображатиме механізм взаємодії елементів кожного блоку. Критерієм правильності структури блоку буде виконання ним заданої в ході моделювання функції.

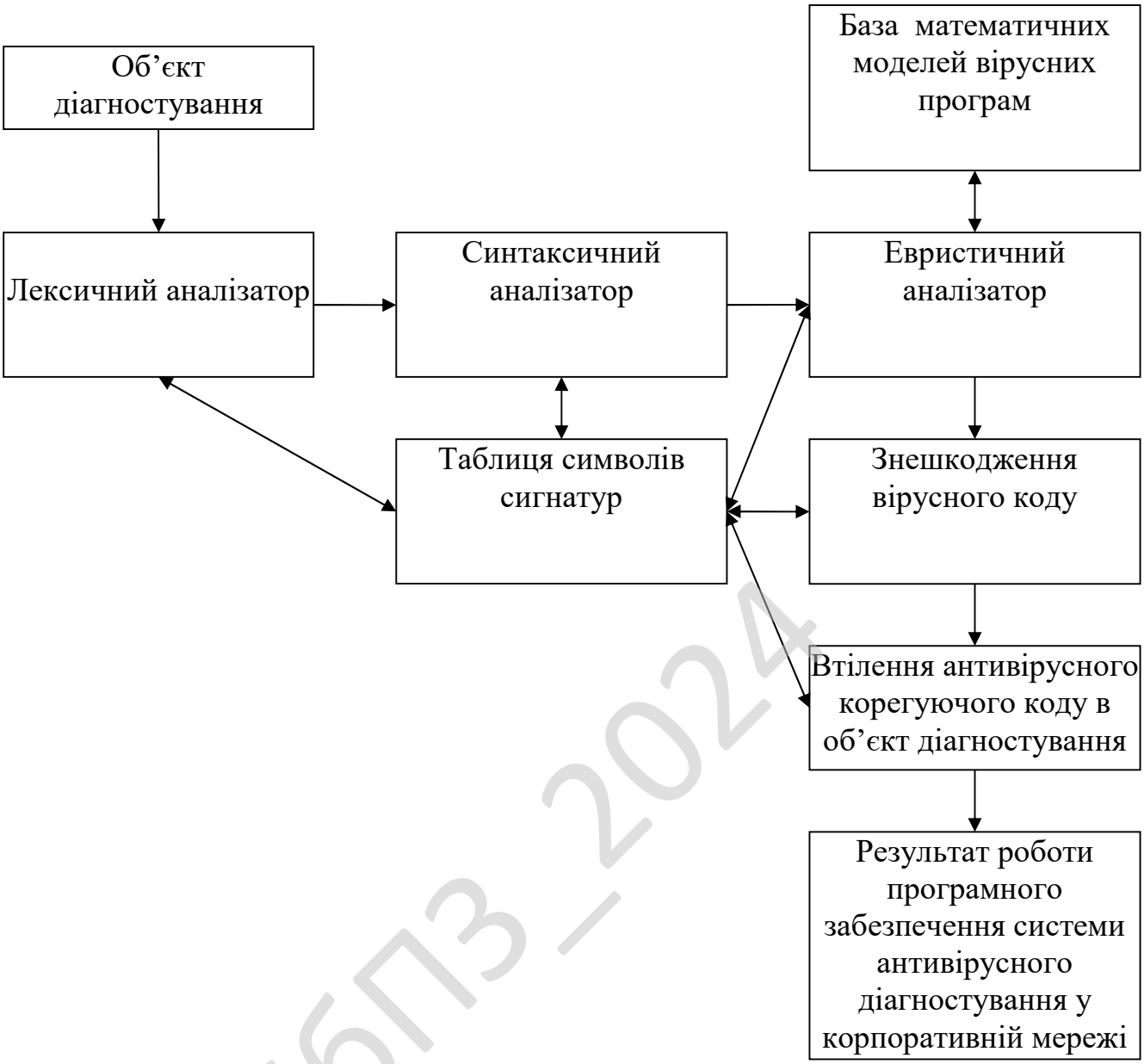

Рисунок 3.1 – Структурна схема системи

Лексичний аналізатор будується як детермінований скінчений автомат по заданому регулярному виразу. Поділ символів на лексеми програмно реалізується побудовою програмної моделі процесора, який є емулятором роботи процесора. Результатом його роботи буде список виконуваних команд об’єкту діагностування, а також порядкові номери кожного символу в файлі і лексеми. В загальному вигляді математичну модель лексичного аналізатора задано функцією:

$$y_i = f(x'_i),$$

де f – функція, яка здійснює поділ об'єктів діагностування $X'_i(i = \overline{0,к})$ на лексеми і результатом виступають елементи $y_i(i = \overline{0,к}) \in Y$, які є ланцюгами лексем, множини X′ та Y складаються з регулярних виразів. Лексичний аналізатор заносить в таблицю символів ланцюг символів лексем разом з порядком їх запису в об'єкті діагностування.

На етапі синтаксичного аналізу досліджується ланцюг лексем і перевіряється, чи задовільняє він структурним умовам, сформульованим в означенні синтаксису мови. По сукупності синтаксичних правил автоматично будується синтаксичний аналізатор, який буде перевіряти, чи має вхідний ланцюг синтаксичну структуру, що визначається цими правилами. Результатом роботи синтаксичного аналізатора виступає деревовидна структура. Внутрішні вершини дерева представляють ті дії, які потрібно виконати. Прямі нащадки кожної вершини представляють документи, до яких потрібно застосувати дії, або визначають тип виконуваної дії. Таке дерево будується в пам'яті персонального комп'ютера і заноситься в таблицю символів. В основу даного методу покладено алгоритм Ерлі. Математичну модель синтаксичного аналізатора задано функцією виду:

$$t_i = \varphi(y_i),$$

де φ – функція, що здійснює розбір елементів $y_i(i = \overline{0,к})$ з множини розширених регулярних виразів Y і результатом виступають синтаксичні дерева $t_i(i = \overline{0,к})$, які утворюють множину T.

Евристичний аналізатор являє собою головний блок антивірусної програми, який здійснює діагностування розширених регулярних виразів множини X′ на зараженість вірусними програмами. Для цього використовується база математичних моделей, в якій зберігається інформація про поведінку вірусних програм в процесі розмноження.

Знешкодження вірусних програм в ОД полягає у виділенні з ланцюга символів підланцюга, що визначає вірусну програму, або втілення на місце

вірусних частин ланцюга символів для неможливості виконання вірусної програми. Модель блоку знешкодження вірусного коду в об'єкті діагностування подано у вигляді:

$$M_Л = <X,V,F_З,G_З>,$$

де $F_З$ – множина алгоритмів знешкодження, $G_З$ – граф зв'язків множин X та V.

Для втілення антивірусного корегуючого коду в ОД використовуємо втілення ланцюга символів у вхідний ланцюг з метою автоматизованого діагностування. Впроваджений ланцюг символів міститиме інформацію про початковий стан вхідного ланцюга. Крім того, він також буде містити засоби (ланцюг лексем) для виявлення та знешкодження в ОД маскуючих вірусних програм. Нехай антивірусний код ОД формується функцією $p(l_i)$ для ОД $x_i'(i = \overline{0,к})$, а результат позначимо через $r_i \in R$, $i = \overline{0,к}$. Математична модель блоку втілення антивірусного коду в ОД втіленням ланцюга символів представлена так:

$$M_З = <R,L,S_B>,$$

де S_B – граф зв'язків множин R і L і описується функцією

$$r_i = q(l_i,p(l_i)),$$

де q – функція приєднання антивірусного коду до ОД l_i, $i = \overline{0,к}$.

Функції φ, f, q, p реалізуються відповідними алгоритмами, які задають взаємнооднозначну відповідність.

Розроблена математична модель процесу антивірусного комбінованого діагностування у корпоративній мережі є основою для розробки методів та алгоритмів антивірусного комбінованого діагностування. Отримана модель процесу антивірусного комбінованого діагностування передбачає використання математичних моделей вірусних програм. При її розробці основна увага приділяється вибору математичних моделей вірусних програм, які представляються у відповідній базі даних матрицями інцидентності. В складі евристичного аналізатора міститься підсистема автоматичної генерації математичних моделей вірусних програм. Вона включає в себе такі блоки: базу

математичних моделей вірусних програм, систему керування базами даних, блок генерації математичних моделей об'єктів діагностування.

Для ефективної організації процесу діагностування щодо персональних комп'ютерів згідно його математичної моделі розроблено метод антивірусного комбінованого діагностування. Він полягає у комбінації методів антивірусного діагностування на основі сигнатур вірусних програм, матриць інцидентності та антивірусного корегуючого коду. Крім того, суть методу антивірусного комбінованого діагностування не тільки у комбінації вищеназваних методів, але й у розроблених з цією ж метою нових методиках, які передбачають нові підходи до здійснення діагностування. Розроблений також метод антивірусного діагностування на основі матриць інцидентності дозволяє створювати антивірусні засоби, які виявлятимуть вірусні програми за їх типами згідно проведеного поділу.

Кожен ОД проходить етап евристичного аналізу. Інформація про відомі вірусні програми зберігається в їх базі даних сигнатур. Якщо сигнатура в ОД знайдена, то відбувається перехід на модуль знешкодження відомих вірусних програм. Але ОД може бути зараженим не тільки однією вірусною програмою, а й декількома. Тому після видалення частин вірусних програм в ОД, він знову перевіряється евристичним аналізатором. І лише після моделювання ОД, якщо стає відомо що він не заражений, то робота евристичного аналізатора закінчується. ОД обов'язково моделюється, навіть якщо в ньому не знайдено сигнатури вірусної програми. Евристичний аналізатор використовує програмний емулятор процесора і здійснює на ньому моделювання ОД з метою отримання його реакцій. Після цього, враховуючи інформацію синтаксичного аналізатора, відбувається порівняння реакцій ОД з еталонами поведінки вірусних програм.

Суть методики антивірусного комбінованого діагностування, яка базується на ідентифікації невідомих вірусних програм за їх автоматично згенерованими математичними моделями:

– Здійснюємо поділ ОД на лексеми. Якщо такий поділ не здійснюється, то встановлюємо факт зараження ОД некоректною вірусною програмою.

– Здійснюємо моделювання ОД до тих пір, поки співпадатимуть відповідні елементи генерованої програмним емулятором матриці інцидентності з елементами матриць баз моделей вірусних програм та деструктивних дій. Якщо матриці співпадуть, то здійснювати перехід на модуль знешкодження вірусного коду. Процес моделювання закінчуємо до досягнення найбільшого розміру матриці інцидентності баз моделей вірусних програм та деструктивних дій.

Дослідження сучасних засобів антивірусного діагностування показало, що в їх складі міститься модуль евристичного аналізатора. Про те, він тільки видає повідомлення про ймовірність зараження об’єкту діагностування. Евристичний аналізатор системи антивірусного комбінованого діагностування є основним модулем, що оцінює ймовірність зараження об’єкту діагностування, як у випадку зараження вірусними програмами, сигнатури яких є в базі даних, так і тими, сигнатури яких відсутні в базі даних. Для випадку зараження об’єкту діагностування невідомими вірусними програмами розроблено алгоритм автоматичної генерації їх моделей та поповнення бази даних сигнатур. Підвищення ефективності антивірусного комбінованого діагностування здійснюється з допомогою використання бази сигнатур вірусних програм, яка має відкриту архітектуру.

Однією із важливих можливостей антивірусних програм є їх здатність до знешкодження вірусних програм в ОД у випадку зараження. Модуль, що реалізує цю частину антивірусної програми починає свою роботу, якщо виявлено наявність вірусу в ОД.

Втілення антивірусного коду в ОД призначено в першу чергу для виявлення факту зараження при запуску ОД на виконання, а також з метою автоматизованого знешкодження вірусного коду.

Суть використаного методу оцінки ефективності алгоритмів антивірусного комбінованого діагностування полягав в необхідності розв’язати дану задачу без врахування логічної структури обчислювальних засобів корпоративної мережі, на яких реалізується алгоритм у вигляді програми, а також при відсутності конкретних даних про обмеження на часткові параметри

алгоритму. Отримані розрахунки ефективності алгоритму є задовільними для практичної реалізації процесу антивірусного комбінованого діагностування у корпоративній мережі.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2. У програмному продукті, що розроблений у ході виконання бакалаврської роботи реалізовані наступні функції:

– Файловий захист у реальному часі.

– Перевірка за розкладом.

– Перевірка кореспонденції по популярних поштових протоколах (POP3, SMTP і ін.).

– Перевірка архівів.

– Перевірка знімних носіїв.

– Веб-антивірус.

– Безпечне спілкування в соціальних мережах.

– Перевірка повідомлень, отриманих через IM.

– Перевірка файлів, отриманих через P2P, IM.

– Автоматичний брандмауер.

– Поведінковий аналіз.

– Евристичний аналіз.

– Сигнатурний аналіз.

– Карантин.

– Хмарні технології (поширення описів нових погроз, імпульсні відновлення).

– Хмарні технології (керування продуктом із хмари).

– Запуск програм в ізольованому середовищі (SandBox).

– Безпека онлайн платежів.

– Віртуальна клавіатура.

– Безпечне уведення даних з апаратної клавіатури.

– Анти-спам.

– Анти-баннер.

– Анти-фішинг.

– Перевірка репутації сайтів.

– Батьківський контроль.

– Пошук і закриття уразливостей.

– Резервне копіювання даних.

– Ранній запуск захисту від шкідливих програм (Early-Launch Anti-Malware, ELAM).

– Контроль цілісності продукту.

Розглянемо деякі з перерахованих функції.

Евристичний аналіз

Евристичний аналіз (евристичне сканування) – це сукупність функцій антивірусу, націлених на виявлення невідомих вірусним базам шкідливих програм, але в той же час цей же термін позначає один з конкретних способів.

Практично всі сучасні антивірусні засоби застосовують технологію евристичного аналізу програмного коду. Евристичний аналіз нерідко використовується разом із сигнатурним скануванням для пошуку складних вірусів, що шифруються й поліморфних вірусів. Методика евристичного аналізу дозволяє виявляти раніше невідомі інфекції, однак, лікування в таких випадках практично завжди виявляється неможливим. У такому випадку, як правило, потрібне додаткове відновлення антивірусних баз для одержання останніх сигнатур і алгоритмів лікування, які, можливо, містять інформацію про раніше невідомий вірус. У противному випадку, файл передається для дослідження антивірусним аналітикам або авторам антивірусних програм.

Методи евристичного сканування не забезпечують якого-небудь гарантованого захисту від нових, відсутніх у сигнатурному наборі комп'ютерних

вірусів, що обумовлено використанням як об'єкт аналізу сигнатур раніше відомих вірусів, а як правила евристичної верифікації – знань про механізм поліморфізму сигнатур. У той же час, оскільки цей метод пошуку базується на емпіричних припущеннях, повністю виключити помилкові спрацьовування не можна.

У ряді випадків евристичні методи виявляються надзвичайно успішними, наприклад, у випадку дуже коротких програмних частин у завантажувальному секторі: якщо програма робить запис у сектор 1, доріжку 0, сторону 0, те це приводить до зміни розділу накопичувача. Але крім допоміжної програми fdisk ця команда більше ніде не використовується, і тому у випадку її несподіваної появи мова йде про завантажувальний вірус.

У процесі евристичного аналізу виробляється перевірка емулюємої програми аналізатором коду. Наприклад, програма інфікована поліморфним вірусом, що складається із зашифрованого тіла й розшифровувача. Емулятор коду зчитує інструкції в буфер антивірусу, розбирає їх на інструкції й робить їхнє виконання по одній інструкції, після цього аналізатор коду підраховує контрольну суму й звіряє її з тою, котра зберігається в базі. Емуляція буде тривати доти, поки необхідна для підрахунку контрольної суми частина вірусу не буде розшифрована. Якщо сигнатура збіглася – програма визначена.

Надмірна підозрілість евристичного аналізатора може викликати помилкові спрацьовування при наявності в програмі фрагментів коду, що виконує дії й/або послідовності, у тому числі й властиві деяким вірусам. Зокрема, розпаковник у файлах, запакованих PE-Пакувальником (Win)Upack викликає помилкові спрацьовування цілого ряду антивірусних засобів, що не визнають такої проблеми.

Наявність простих методик обману евристичного аналізатора. Як правило, перш ніж поширювати шкідливу програму (вірус), її розроблювачі досліджують існуючі розповсюджені антивірусні продукти, різними методами уникаючи її детектування при евристичному скануванні. Наприклад, видозмінюючи код,

використовуючи елементи, виконання яких не підтримується емулятором коду даних антивірусів, використовуючи шифрування частини коду й ін.

Незважаючи на заяви й рекламні проспекти розроблювачів антивірусних засобів щодо вдосконалювання евристичних механізмів, ефективність евристичного сканування на даний момент далека від очікуваної. Незалежні тести компонентів евристичного аналізу показують, що рівень виявлення нових шкідливих програм становить не більш ніж 40-50 % від їхнього числа.

Навіть при успішному визначенні, лікування невідомого вірусу практично завжди є неможливим. Як виключення, деякими продуктами можливе лікування однотипних і ряду поліморфних, що шифруються вірусів, що не мають постійного вірусного тіла, але впровадження, що використовують єдину методику. У такому випадку, для лікування десятків і сотень вірусів може існувати один запис у вірусній базі.

Проактивні технології

Проактивні технології – сукупність технологій і методів, використовуваних в антивірусному програмному забезпеченні, основною метою яких, на відміну від реактивних (сигнатурних) технологій, є запобігання зараження системи користувача, а не пошук уже відомого шкідливого програмного забезпечення в системі. При цьому проактивний захист намагається блокувати потенційно небезпечну активність програми тільки в тому випадку, якщо ця активність являє реальну загрозу. Серйозний недолік проактивного захисту – блокування легітимних програм. Проактивні технології почали розвиватися практично одночасно із класичними (сигнатурними) технологіями. Однак, перші реалізації проактивних технологій антивірусного захисту вимагали високого рівня кваліфікації користувача, тобто не були розраховані на масове використання простими користувачами персональних комп'ютерів. Через десятиліття антивірусної індустрії стало очевидно, що сигнатурні методи виявлення вже не можуть забезпечити ефективний захист користувачів. Цей факт і підштовхнув до відродження проактивних технологій.

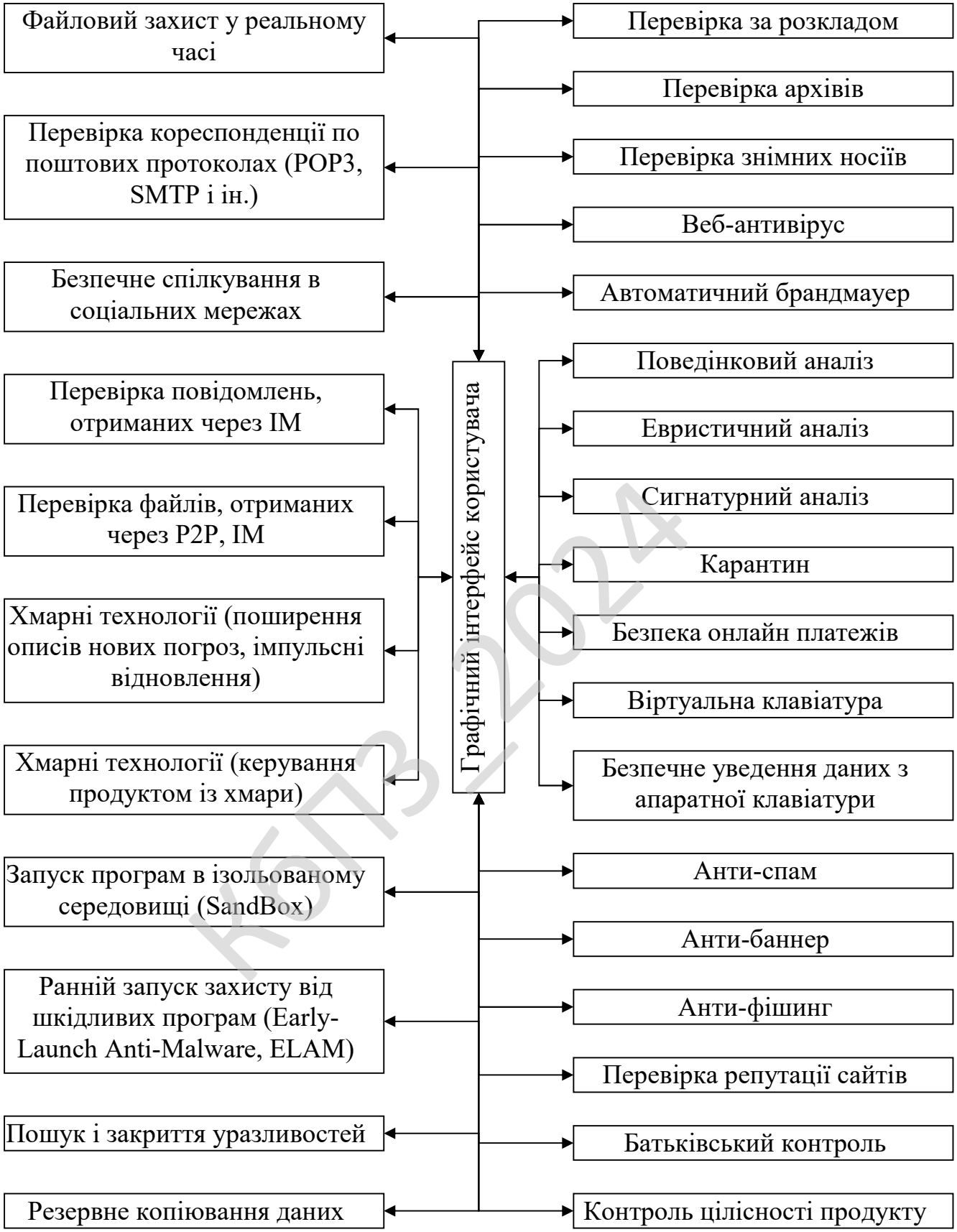

Рисунок 3.2 – Функціональна схема системи

Технології проактивного захисту:

– Евристичний аналіз. Технологія евристичного аналізу дозволяє на основі аналізу коду виконуваного додатка, скрипту або макросу виявити ділянки коду, відповідальні за шкідливу активність. Ефективність даної технології не є високої, що обумовлено більшою кількістю помилкових спрацьовувань при підвищенні чутливості аналізатора, а також більшим набором технік, використовуваних авторами шкідливого ПЗ для обходу евристичного компонента антивірусного ПЗ.

– Емуляція коду. Технологія емуляції дозволяє запускати додаток у середовищі емуляції, емулюючі поводження ОС або центрального процесора. При виконанні додатка в режимі емуляції додаток не зможе нанести шкоди системі користувача, а шкідлива дія буде детектована емулятором. Незважаючи на гадану ефективність даного підходу, він також не позбавлений недоліків – емуляція займає занадто багато часу й ресурсів комп'ютера користувача, що негативно позначається на швидкодії при виконанні повсякденних операцій, також, сучасні шкідливі програми здатні виявляти виконання в емульованому середовищу й припиняти своє виконання в ній.

– Аналіз поводження. Технологія аналізу поводження ґрунтується на перехопленні всіх важливих системних функцій або установці т.зв. міні-фільтрів, що дозволяє відслідковувати всю активність у системі користувача. Технологія поведінкового аналізу дозволяє оцінювати не тільки одиничну дію, але й ланцюжок дій, що багаторазово підвищує ефективність протидії вірусним погрозам. Також, поведінковий аналіз є технологічною основою для цілого класу програм – поведінкових блокаторів (HIPS – Host-based Intrusion Systems).

– Sandboxing (Пісочниця) – обмеження привілеїв виконання. Технологія Пісочниці працює за принципом обмеження активності потенційно шкідливих додатків таким чином, щоб вони не могли нанести шкоди системі користувача. Обмеження активності досягається за рахунок виконання невідомих додатків в обмеженому середовищі – властиво пісочниці, звідки додаток не має прав доступу до критичних системних файлів, віткам реєстру й іншої важливої

інформації. Технологія обмеження привілеїв виконання є ефективною технологією протидії сучасним погрозам, але, варто розуміти, що користувач повинен мати знання, необхідними для правильної оцінки невідомого додатка.

– Виртуалізація робітника оточення. Технологія виртуалізації робітника оточення працює за допомогою системного драйвера, що перехоплює всі запити на запис на жорсткий диск і замість виконання запису на реальний жорсткий диск виконує запис у спеціальну дискову область – буфер. Таким чином, навіть у тому випадку, якщо користувач запустить шкідливе програмне забезпечення, воно проживе не далі чим до очищення буфера, що за замовчуванням виконується при вимиканні комп'ютера. Однак, варто розуміти, що технологія виртуалізації робітника оточення не зможе захистити від шкідливих програм, основною метою яких є крадіжка конфіденційної інформації, тому що доступ на читання до жорсткого диска не заборонений.

У цей час проактивні технології є важливим і невід'ємним компонентом антивірусного програмного забезпечення. Більше того, як правило, в антивірусних продуктах використовується сполучення відразу декількох технологій проактивного захисту, наприклад евристичний аналіз і емуляція коду успішно сполучаються з поведінковим аналізом, що може дозволити досвідченому користувачеві багаторазово підвищити ефективність сучасних антивірусних продуктів проти нових, усе більше й більше витончених шкідливих програм.

Руткіт

Руткіт – набір програмних засобів (наприклад, файлів що виконуються, скриптів, конфігураційних файлів), для забезпечення:

– маскування об'єктів (процесів, файлів, директорій, драйверів);

– контролю (подій які відбуваються у системі);

– збору даних (параметрів системи).

Термін Rootkit історично прийшов з миру UNIX, і під цим терміном розуміється набір утиліт або спеціальний модуль ядра, які зловмисник установлює на зламаній їм комп'ютерній системі відразу після одержання прав

суперкористувача. Цей набір, як правило, містить у собі різноманітні утиліти для «замітання слідів» вторгнення в систему, робить непомітними сніффери, сканери, кейлоггери, троянські програми, що заміщають основні утиліти UNIX (у випадку неядерного руткіту). Rootkit дозволяє зломщикові закріпитися в зламаній системі й сховати сліди своєї діяльності шляхом приховання файлів, процесів, а також самої присутності руткіту в системі.

У систему руткіт може бути встановлений різними способами: завантаження за допомогою експлойта, після одержання шелл-доступу (у такому випадку, може використовуватися засіб типу wget або вихідний FTP-клієнт для завантаження руткіту з вилученого пристрою), у вихідному коді або ресурсах програмного продукту.

Файєрвол

Міжмережний екран, мережний екран, файєрвол, брандмауер – комплекс апаратних або програмних засобів, що здійснює контроль і фільтрацію минаючих через нього мережних пакетів відповідно до заданих правил. Основним завданням мережного екрана є захист комп'ютерних мереж або окремих вузлів від несанкціонованого доступу. Також мережні екрани часто називають фільтрами, тому що їхнє основне завдання – не пропускати (фільтрувати) пакети, що не підходять під критерії, визначені в конфігурації. Деякі мережні екрани також дозволяють здійснювати трансляцію адрес – динамічну заміну внутримережних (сірих) адрес або портів на зовнішні, використовувані за межами ЛОМ.

Пісочниця

Пісочниця (sandbox) – у комп'ютерній безпеці механізм для безпечного виконання програм.

Пісочниця звичайно являє собою жорстко контрольований набір ресурсів для виконання гостьової програми – наприклад, місце на диску або в пам'яті. Доступ до мережі, можливість повідомлятися з головною операційною системою або зчитувати інформацію із пристроїв уведення звичайно або частково емулюють, або сильно обмежують. Пісочниці являють собою приклад

віртуалізації. Підвищена безпека виконання коду в пісочниці найчастіше пов'язана з великим навантаженням на систему – саме тому деякі види пісочниць використовують тільки для неналагодженого або підозрілого коду.

Як правило, пісочниці використовують для запуску неперевіреного коду з невідомих джерел, як засіб проактивного захисту від шкідливого коду, а також для виявлення й аналізу шкідливих програм. Також найчастіше пісочниці використовуються в процесі розробки програмного забезпечення для запуску «сирого» коду, що може випадково ушкодити систему або зіпсувати складну конфігурацію. Такі «тестувальні» пісочниці копіюють основні елементи середовища, для якої пишеться код, і дозволяють розроблювачам швидко й безболісно експериментувати з неналагодженим кодом.

Анти-спам

Анти-спам – метод, частково або цілком запобігає влучення небажаної інформації (спаму) у кругозір користувача або групи користувачів ПК. Дія методу здійснюється шляхом настроювання програмного забезпечення як на стороні клієнта, так і на стороні сервера, що може служити посередником при доступі до мережі Інтернет.

Анти-спам використовується:

– В інтернет-браузерах. Застосовуються настроювання модуля браузера й/або програмного забезпечення, які запобігають влученню небажаного контенту в кругозір користувача або групи користувачів.

– В електронній пошті. Застосовуються настроювання програмного забезпечення (модулів) сервера електронної пошти (поштовий сервер) для інтерактивного аналізу (у реальному часі) отриманих листів і визначення (маркування) цих листів у категорію «спам».

– У програмах обміну повідомленнями, а також різних програмах, що мають таку можливість (наприклад, у клієнтах DC++).

Фішинг

Фішинг – вид інтернет-шахрайства, метою якого є одержання доступу до конфіденційних даних користувачів – логінам і паролям.

Це досягається шляхом проведення масових розсилань електронних листів від імені популярних брендів, а також особистих повідомлень усередині різних сервісів, наприклад, від імені банків або усередині соціальних мереж.

У листі часто втримується пряме посилання на сайт, зовні неотличимый від сьогодення, або на сайт із редиректом.

Після того, як користувач попадає на підроблену сторінку, шахраї намагаються різними психологічними прийомами спонукати користувача ввести на підробленій сторінці свої логін і пароль, які він використовує для доступу до певного сайту, що дозволяє шахраям одержати доступ до аккаунтами банківських рахунків.

Фішинг – один з різновидів соціальної інженерії, заснована на незнанні користувачами основ мережної безпеки: зокрема, багато хто не знають простого факту: сервіси не розсилають листів із проханнями повідомити свої облікові дані, пароль та інше.

Для захисту від фішинга виробники основних інтернет-браузерів домовилися про застосування однакових способів інформування користувачів про те, що вони відкрили підозрілий сайт, що може належати шахраям.

Нові версії браузерів уже мають таку можливість, що відповідно йменується «антифішинг».

Екранна клавіатура

Екранна клавіатура (віртуальна клавіатура) – клавіатура, зображена на екрані комп'ютера.

Натискання на клавіші здійснюється курсором миші, або, для сенсорних екранів, пальцем користувача або стілусом.

Віртуальні клавіатури можуть використовуватися для зменшення ризику реєстрації натискання клавіш.

При цьому шкідливим програмам складніше одержувати дані, чим при реальних натисканнях клавіш.

Однак є ризик, що шкідлива програма буде робити скріншоти з регулярними інтервалами часу або після кожного клацання миші.

Для рятування від цієї проблеми в Японії винайшли "фальшиві курсори" для віртуальних клавіатур, так званий алгоритм Symmetric Cursors, при якому по віртуальній клавіатурі крім справжнього курсору рухається велика кількість фальшивих курсорів.

У результаті сканування екрана не визначить, куди дійсно спрямований курсор миші.

Батьківський контроль

Батьківський контроль – комплекс правил і мер по запобіганню негативного впливу мережі Інтернет і комп'ютера на опікувану людину (звичайно дитини).

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. Після початку роботи розробленого ПЗ ми потрапляємо до інтерфейсу ПЗ звідки проводиться налаштування ПЗ, перегляд довідкової інформації та через обробник помилок проходе обрання об'єкту діагностування.

Далі проходе робота з виявлення вірусів за допомогою бази математичних моделей вірусних програм з автоматичним оновлення БД чи ручним. Виявлення проходить спочатку через лексичний аналізатор з використанням таблиці символів сигнатур, синтаксичний аналізатор та остаточно через евристичний аналізатор з подальшим знешкодженням вірусного коду та втіленням антивірусного корегуючого коду в об'єкт діагностування.

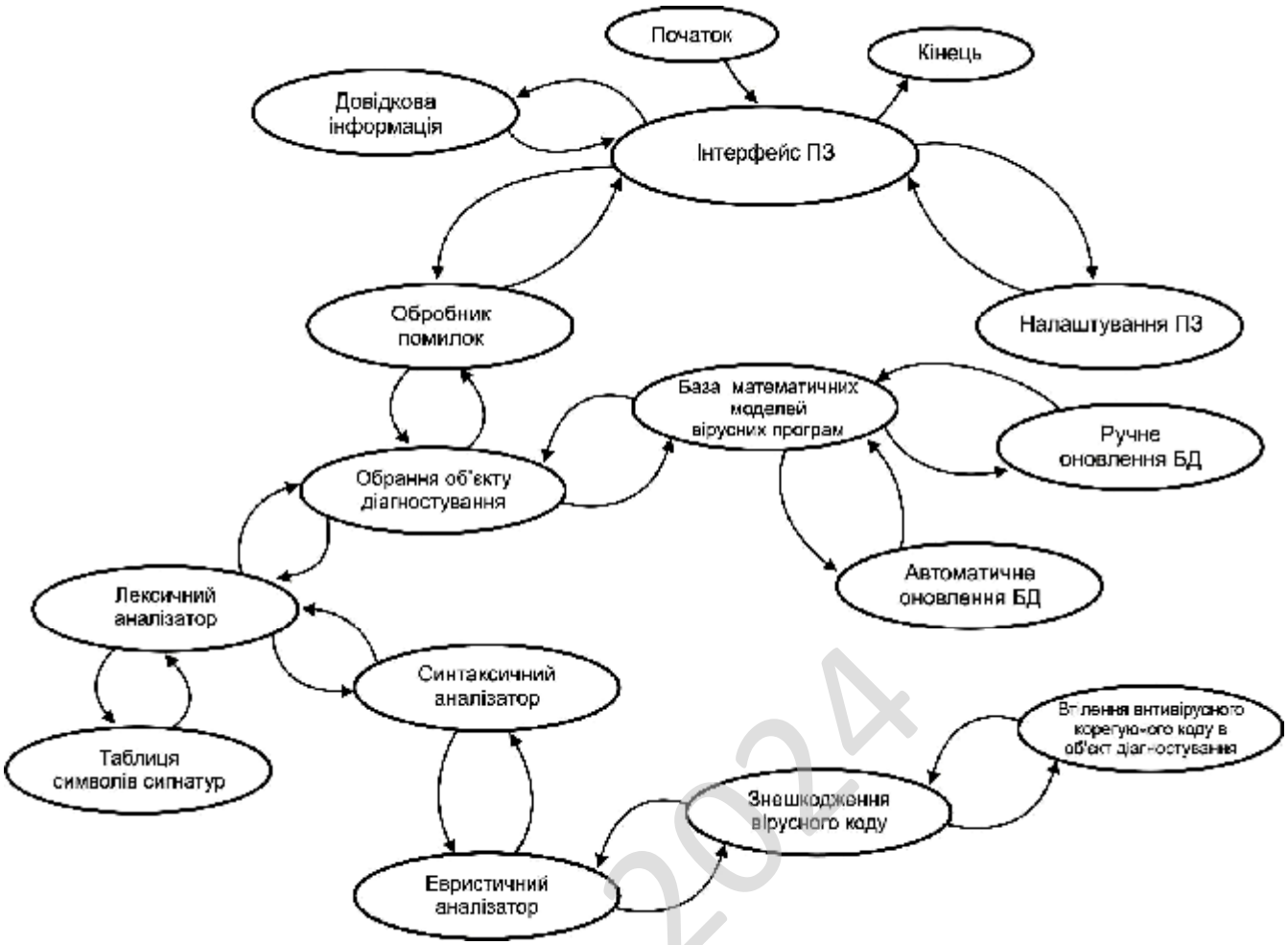

Рисунок 3.3 – Діаграма взаємодії процесів

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

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо алгоритм роботи основної програми. Його блок-схема зображена на рисунку 4.1.

З рисунку видно, що після запуску програми спочатку відбувається виділення пам’яті ПЗ. Потім здійснюється:

– Ініціалізація початкових змінних ПЗ.

– Підключення файлу налаштування ПЗ.

– Підключення модулів реєстру та файлової підсистеми.

– Тест цілісності ПЗ та доступу до файлової підсистеми.

– Тест пройдено (запит).

– Очікування запиту початку сканування об'єкту діагностування.

– Запит WM_CLOSE?

– Є запит сканування ОД?

– Підпрограма антивірусного комбінованого діагностування.

– Є запит оновлення мат. мод. вірусного ПЗ?

– Підпрограма оновлення математичних моделей вірусного ПЗ.

– Проведення оновлення бази, запис дати та часу оновлення.

– Запит WM_CLOSE?

– Виявлено помилки роботи ПЗ (запит).

– Виведення повідомлення тіла помилки.

– Звільнення виділених ресурсів програми.

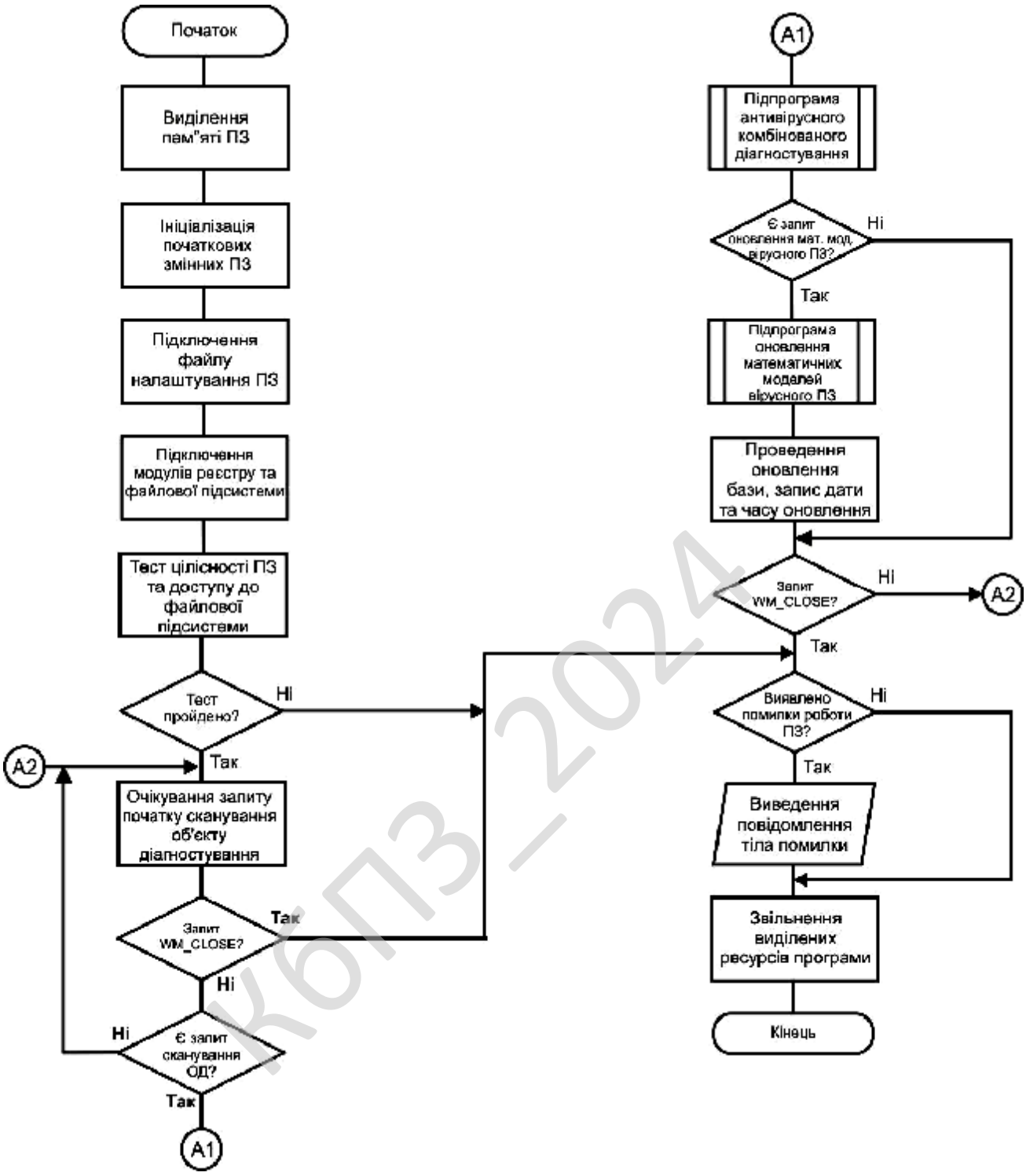

Рисунок 4.1 – Блок-схема основної програми

З рисунку 4.2 видно роботу підпрограми антивірусного комбінованого діагностування. Де проходе:

– Читання файлу бази математичних моделей вірусного ПЗ.

– Створення масиву сигнатур комбінованого діагностування.

– Читання файлової підсистеми.

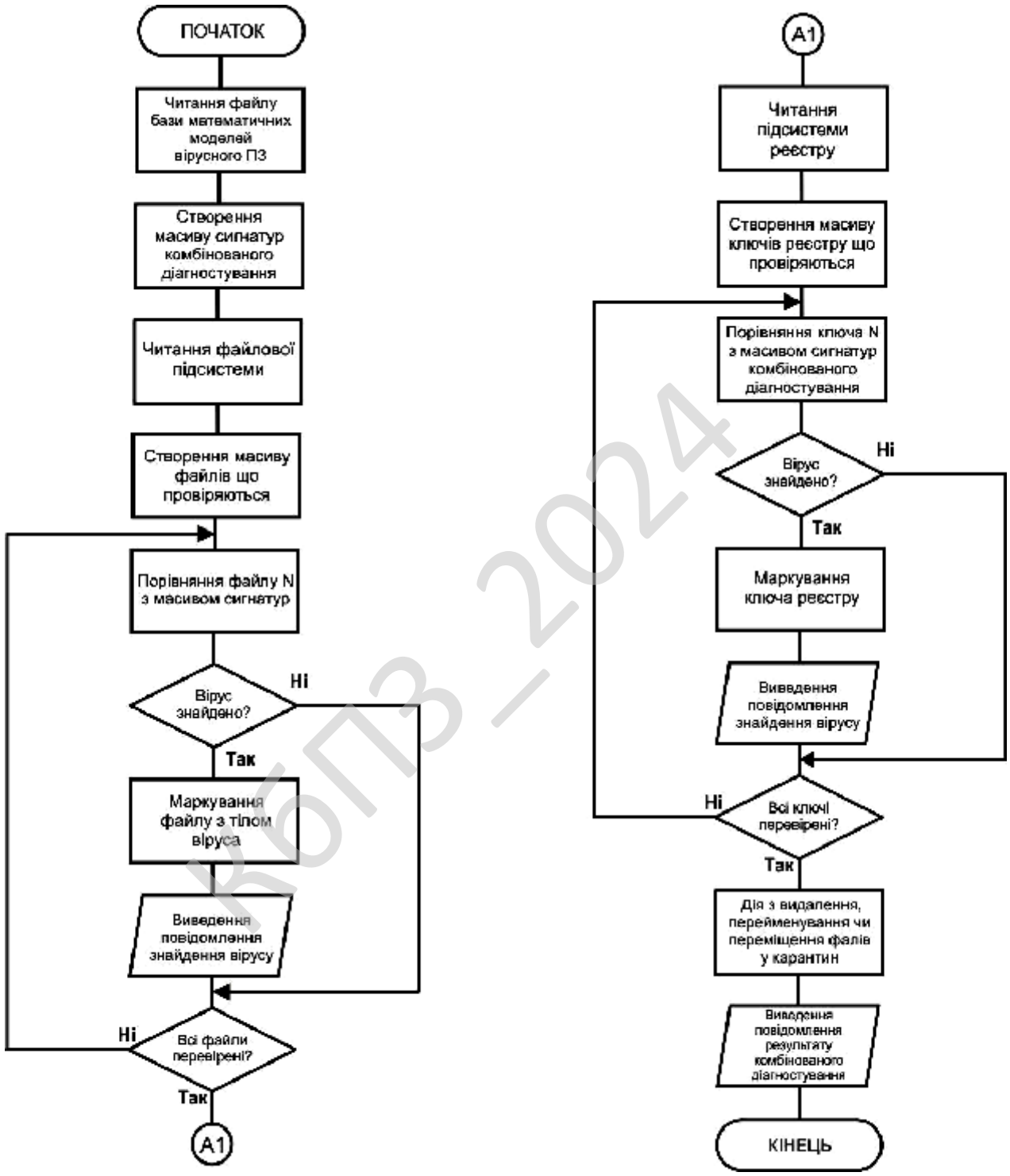

Рисунок 4.2 – Блок-схема підпрограми антивірусного комбінованого діагностування

– Створення масиву файлів що провіряються.

– Порівняння файлу N з масивом сигнатур .

– Вірус знайдено (запит).

– Маркування файлу з тілом віруса.

– Виведення повідомлення знайдення вірусу.

– Всі файли перевірені (запит).

– Читання підсистеми реєстру.

– Створення масиву ключів реєстру що провіряються.

– Порівняння ключа N з масивом сигнатур комбінованого діагностування.

– Вірус знайдено (запит) .

– Маркування ключа реєстру.

– Виведення повідомлення знайдення вірусу.

– Всі ключі перевірені (запит).

– Дія з видалення, перейменування чипереміщення фалів у карантин.

– Виведення повідомлення результату комбінованого діагностування.

З рисунку 4.3 видно роботу підпрограми оновлення математичних моделей вірусного ПЗ. Де проходе:

– Виведення вікна оновлення математичних моделей вірусного ПЗ.

– Читання файлу бази математичних моделей вірусного ПЗ.

– Встановлення версії БД .

– Ручне оновлення (запит).

– Введення шляху знаходження файлу БД математичних моделей вірусного ПЗ.

– Версія файлу БД нова (запит).

– Задіяти файл бази математичних моделей вірусного ПЗ.

– Виведення повідомлення оновлення БД.

– Автоматичне оновлення (запит).

– Запит доступу до глобальної мережі Інтернет.

– Є доступ до мережі Інтернет (запит).

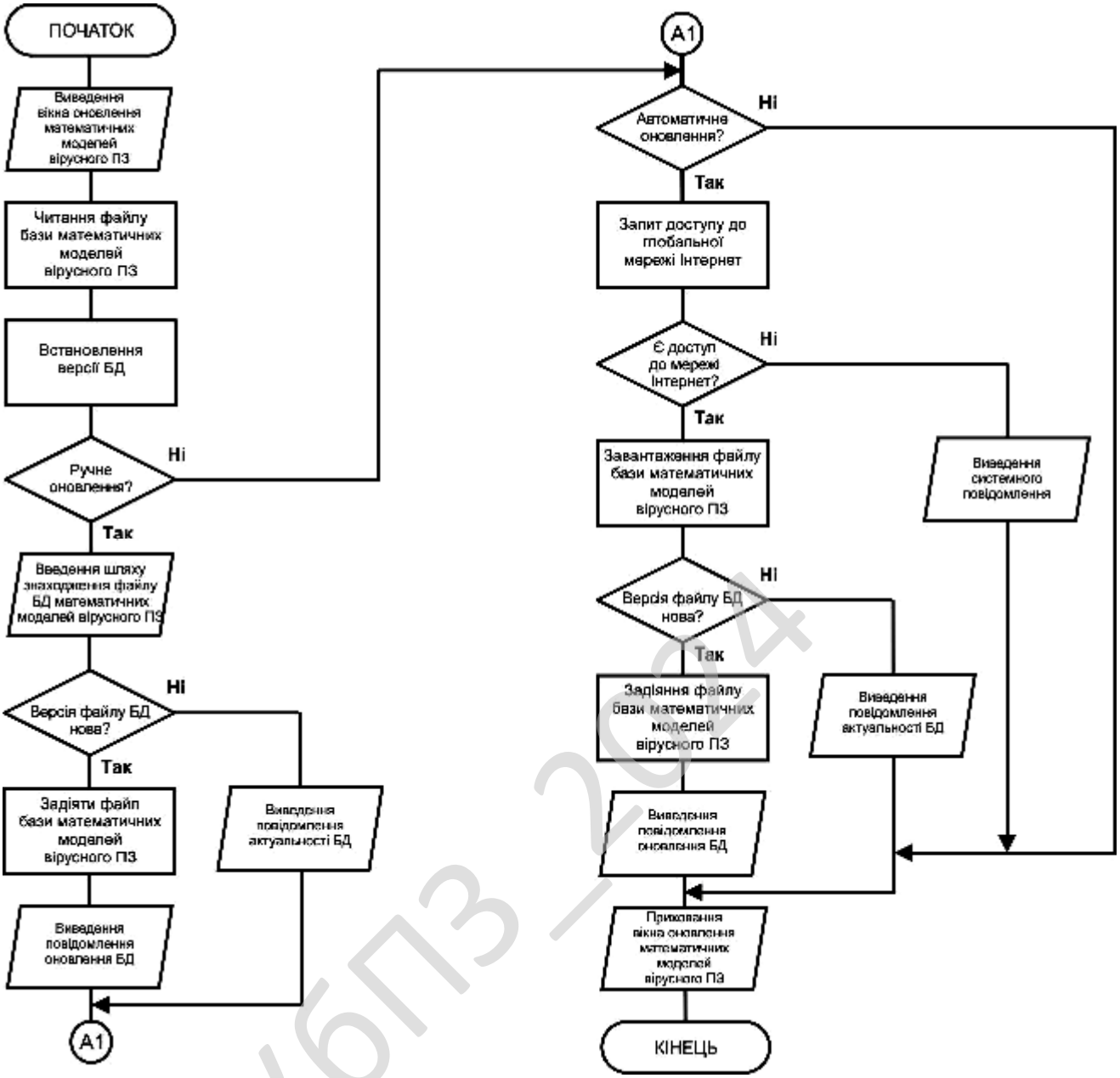

Рисунок 4.3 – Блок-схема підпрограми оновлення математичних моделей вірусного ПЗ

– Завантаження файлу бази математичних моделей вірусного ПЗ.

– Версія файлу БД нова (запит).

– Задіяти файл бази математичних моделей вірусного ПЗ.

– Виведення повідомлення оновлення БД.

– Приховання вікна оновлення математичних моделей вірусного ПЗ.

Опис алгоритмів функціонування системи

Розглянемо розроблений клас TMemory який забезпечує збереження даних в адресному просторі

При цьому методи доступу до цих даних залишаються тими ж, що й при роботі з файловими потоками. Це дозволяє використовувати адресний простір для зберігання проміжних результатів роботи ПЗ, а також за допомогою стандартних методів здійснювати обмін даними між пам'яттю й іншими фізичними носіями.

Розглянемо розроблений вихідний код порівняння двох файлів по вмісту використовуємо тільки модуль Windows – головний алгоритм порівняння сигнатури з файлом який перевіряється.

Вихідний код дозволяє зрівняти два файли по змісту. Якщо є відмінності – видає false, якщо файли однакові – true.

Ціль написання даної функції – уникнути використання додаткових модулів Delphi, таких як sysutils, classes, тому що вони працюють украй нестабільно.

Додаткові функції: fileexists(перевірка наявності файлу), Tempdir(довідаємося тимчасову папку windows mobile), Createtemporaryfile (створення тимчасового файлу на диску, якщо це потрібно).

```
function Tempdir: string;
{функція повертає шлях до папки тимчасових файлів}
var Dir: array[0..MAX_PATH - 1] of char;
begin
 Gettemppath(Sizeof(Dir),Dir);
 Result := Dir;
end;

function Fileexists(const Filename: pchar): Bool;
{імпортуємо функції перевірки наявності файлу з модуля sysutils}
 function Fileage(const Filename: pchar): Integer;
 var
  Handle: Thandle;
  Finddata: Twin32Finddata;
  Localfiletime: Tfiletime;
 type
```

```
  Longrec = packed record
   case Integer of
    0: (Lo, Hi: Word);
    1: (Words: array[0..1] of Word);
    2: (Bytes: array[0..3] of Byte);
  end;
 begin
  Handle := Findfirstfile(Pchar(Filename), Finddata);
  if Handle <> INVALID_HANDLE_VALUE then
   begin
    Windows.Findclose(Handle);
    if (Finddata.dwfileattributes and FILE_ATTRIBUTE_DIRECTORY) = 0 then
     begin
      Filetimetolocalfiletime(Finddata.ftlastwritetime, Localfiletime);
      if Filetimetodosdatetime(Localfiletime, Longrec(Result).Hi,
       Longrec(Result).Lo) then Exit;
     end;
   end;
  Result := -1;
 end;
begin
 result := false;
 if filename = '' then exit;
 Result := Fileage(Filename) <> -1;
end;

function Createtemporaryfile(Filename: string): string;
{створення тимчасової копії порівнюваних файлів,
  повертає ім'я свіжоспеченого файлу.}
const S: string = '_Qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjk
                   lzxcvbnm1234567890';
var i, N: integer;
X: string;
label A;
begin
 Randomize;
 A:
 X := '';
 for i := 0 to 7 do {генеруємо ім'я файлу довжиною 8 знаків}
  begin
   N := Random(Length(S)+1);
   if N = 0 then Goto A;
```

```
    X := X + S[N];
   end;
  X := Tempdir + X + '.tmp';
  if Fileexists(pchar(X)) = true then goto a else
   begin
    if Copyfile(pchar(Filename), pchar(X), true) = true then
     Result :=X else Result := '';
   end;
end;

{ функція порівняння }
function Comparefiles(File1, File2: string;
                      Createtempfile: Boolean): Boolean;
{Createtempfile – параметр задає, чи необхідно копіювати порівнювані
файли чи ні. Зроблене для запобігання помилок, якщо файл,
приміром використовується запущений exe файл}
var F1, F2: file;
 B1, B2: array[0..1023] of Char;
 i1,i2:Integer;
begin
Result:=false;
 if (Fileexists(pchar(File1)) = false) or
     (Fileexists(pchar(File2)) = false) then
  Exit;
{якщо один з файлів відсутній, то виходимо}
 if Createtempfile = true then
{якщо треба – створюємо тимчасові копії в папці Temp}
  begin
   File1 := Createtemporaryfile(File1);
   File2 := Createtemporaryfile(File2);
  end;
 Assign(F1, File1);
 Assign(F2, File2);
 Reset(f1, 1);
 Reset(f2, 1);
 if Filesize(f1) <> Filesize(f2) then
{якщо розміри файлів не збігаються, то вони(файли) у
кожному разі не ідентичні}
  begin
   Closefile(F1);
   Closefile(F2);
   if Createtempfile = true then
```

```
    begin
     Deletefile(pchar(File1));
{забираємо сміття за собою}
     Deletefile(pchar(File2));
    end;
   Exit;
  end;
 repeat
{повторюємо операції поки файл не закінчиться}
  Blockread(F1, B1, Sizeof(B1),i1);
  Blockread(F2, B2, Sizeof(B2),i2);
{блочно читаємо й порівнюємо блоки}
{як тільки попадуться два, що різняться блоку,
відразу виходимо, result:=false}
  if B1 <> B2 then
   begin
    Result := false;
    Closefile(F1);
    Closefile(F2);
    if Createtempfile = true then
     begin
      Deletefile(pchar(File1));
      Deletefile(pchar(File2));
     end;
    Exit;
   end else Result := true;
 until Eof(F2);
{кінець файлу}
 Closefile(F1);
 Closefile(F2);
 if Createtempfile = true then
{якщо ми створювали тимчасові копії, то їх потрібно вилучити}
  begin
   Deletefile(pchar(File1));
   Deletefile(pchar(File2));
  end;
end;
```

Для видалення тимчасових файлів використовувалася функція Deletefile у вигляді коду:

```
if not Deletefile('c:\Data\input\tmpdat.rar') then Showmessage
                  ('Помилка видалення');
```

Також при роботі з файлами було необхідне додавати рядок тексту у файл роботи програми (LOG файл).

Розглянемо реалізацію:

```
proceduretform1.Button1Click(Sender: Tobject);
Varf:Textfile;
// оголошення файлової змінної для LOG файлу
st:String;
// строкова змінна
begin
// прив'язка назви файлу до файлової змінної
Assignfile(f,'c:\Data\log\log.txt');
{ Команда Assignfile здійснює прив'язку рядка шляху файлу
до файлової змінної. Усі подальші операції з файлової
змінної автоматично здійснюються із зазначеним файлом.}
{$I-}
// відключення контролю помилок введення / виведення
Reset(f);
// відкриття файлу для читання
{$I+}
// включення контролю введення / виведення
ifioresult<>0 then
// якщо є помилка відкриття, то ….
begin
Showmessage('Помилка відкриття файлу log.txt');
Exit;
// вихід із процедури при помилці відкриття файлу
end;
While not EOF(f) do
// поки не кінець файлу робити цикл:
begin
   Readln(f,st);
// читати з файлу рядок
   Showmessage(st);
// виведення рядка
end;
Closefile(f); // закрити файл
end;
```

{$I-} і {$I+} є директивами компіляторові, що в цьому місці відповідно слід відключити й включити контроль помилок введення / виведення.

У цьому випадку при невдалій спробі відкриття файлу log.txt(файл відсутній або відкритий для запису іншою програмою) програма не видасть аварійної помилки й продовжить виконання даної процедури. Ця властивість корисна для обробки всіх можливих випадків у роботі програми.

Ioresult – змінна, яка зберігає в собі код помилки останньої операції введення / виведення. Якщо вона дорівнює нулю, то остання операція була успішно виконана.

EOF(Файл) – функція, що повертає ознаку кінця файлу, показує, досягнуться чи ні кінець відкритого файлу.

Readln (Файл, Змінна) – процедура зчитування змінної з файлу. На відміну від команди Read робить зчитування рядка із символом, що завершується, перекладу рядка під кодами 13 і 10 (клавіша Enter).

Closefile(Файл) – процедура закриття раніше відкритого файлу.

Розглянемо основні моменти при написанні програми для бакалаврської роботи.

При початковій стадії проектування було з'ясовано, що написання програми буде проходити винятково за допомогою Remote API (RAPI). RAPI у делфі реалізується використанням бібліотеки RAPI.DLL і інтерфейсу бібліотеки RAPI.pas.

Розглянемо розроблений код для копіювання файлів (з використанням бібліотеки RAPI.DLL).

```
procedure Tform1.Button1Click(Sender: Tobject);
var
// оголошення змінних
stemplate:Tstringlist;
Hdes:Thandle;
o1:Poverlapped;
Byteswritten:DWORD;
begin
stemplate:=Tstringlist.Create;
stemplate.Loadfromfile('test.xml');
cerapiinit;
// створення  й копіювання файлу
```

```
        Hdes   :=   Cecreatefile   ('\ipaq   file   store\test.xml',   Generic_Write,
                                   File_Share_Write,nil, Create_Always,0,0);
        Cewritefile(Hdes,Pchar(stemplate.Text),Length(stemplate.Text),
                    Byteswritten,@o1);
        Ceclosehandle(Hdes);
        Cerapiuninit;
        showmessage('Скопійоване: '+inttostr(Byteswritten) + ' байт');
        //виведення результату копіювання
        end;
```

На основі цього коду й враховуючи особливості копіювання файлів я написав функцію для копіювання файлу з параметрами.

```
        Function COPYFILE (Dest,Source:Pchar;
                           Nomessage: boolean=false) :integer;
     var Filehdl,Cehandle,Handle:  Thandle;
     Cefilename:Pwidechar; Res:integer;
     Boolres:boolean; Buffercount,Readcount:cardinal;
     buffer:array [1..1024] of char;
     writesbyte:cardinal; Msgstr:String;
begin
     Res:=0;
     Handle:=Loadlibrary('rapi.dll');
// Підключення бібліотеки
     if Handle<>0 then
      begin
// Створення файлу
      Cewritefile := Getprocaddress(Handle, 'Cewritefile');
      Cecreatefile := Getprocaddress(Handle, 'Cecreatefile');
//  Створення директорії
      Cecreatedirectory := Getprocaddress(Handle, 'Cecreatedirectory');
// Одержання адресного простору
      Ceclosehandle := Getprocaddress(Handle, 'Ceclosehandle');
      Cerapiinit := Getprocaddress(Handle, 'Cerapiinit');
      Cerapiuninit := Getprocaddress(Handle, 'Cerapiuninit');
// Перевірка стану
      Cegetlasterror := Getprocaddress(Handle, 'Cegetlasterror');
      if Assigned(Cerapiinit) then Cerapiinit;
// Виділення пам'яті
      Getmem(Cefilename,512);
      Msgstr:=Extractfiledir(Dest);
      Cefilename:=Stringtowidechar(Msgstr,Cefilename,length(Dest)+1);
      Boolres:=Cecreatedirectory(Cefilename,nil);
```

```
//Перетворення типу даних
      Cefilename:=Stringtowidechar(Dest,Cefilename,length(Dest)+1);
// Створення файлу з параметрами
      Cehandle:=Cecreatefile(Cefilename,GENERIC_READ       or       GENERIC_WRITE,
                             0,nil, CREATE_ALWAYS,
                             FILE_ATTRIBUTE_NORMAL,0);
 if (Cehandle<>INVALID_HANDLE_VALUE) then
     begin
     Buffercount:=1024;Readcount:=1024;
     Filehdl:=Createfile(Pchar(Source),GENERIC_READ,0,nil,OPEN_EXISTING,
                         FILE_ATTRIBUTE_NORMAL,0);
  if Filehdl<>INVALID_HANDLE_VALUE then
   begin
   while Readfile(Filehdl,Buffer,Buffercount,Readcount,nil)
                  and (Readcount<>0) do
   Cewritefile(Cehandle,buffer,Readcount,writesbyte,nil);
   Closehandle(Filehdl);
   end
  else
   begin
// Виникли помилки під час виконання
   Res:=Getlasterror;
   Msgstr:='Не вдалося відкрити файл'+Source;
  // Якщо є помилки вивести її на екран
if Nomessage=false then Showmessage(Msgstr+'. Код помилки:'+Inttostr(Res));
end; //if Filehdl...
// Закриття файлу звільнення пам'яті
  Ceclosehandle(Cehandle);
  Freememory(Cefilename);
  end
 else
  begin
// Якщо є помилки вивести її на екран
  Res:=Cegetlasterror;
  Msgstr:='Не вдалося створити файл '+Cefilename;
  if Nomessage=false then Showmessage(Msgstr+'. Код помилки:'+Inttostr(Res));
  end;//if Cehandle...
 if Assigned(Cerapiuninit) then Cerapiuninit;
// Звільнення ресурсів виділеної бібліотеки
 Freelibrary(Handle);
 end//if Handle<>0...
else
```

```
 begin
 Res:=1;
// Якщо бібліотека RAPI не знайдено вивести повідомлення
if Nomessage=false then Showmessage('Не вдалося відкрити бібліотеку RAPI.');
 end;
Result:=Res;
end;
```

Розглянемо виклик і роботу бакалаврської роботи. Для виклику ПЗ необхідно виконати рядок коду:

```
device=Createobject ("DATA_WRAPI.Device").
```

Необов'язкові параметри: Errorcode (число) – код помилки; Errorstring (рядок) – опис помилки. У ці попередньо створені параметри програма поверне код і опис помилки. Якщо, повертається значення (тип Boolean) неправда або 0 – програма працює, якщо істина або будь-яке інше значення – програма не працює.

Розглянемо основний розроблений (Device) кореневий клас для керування ПК, його властивості й методи:

– Connect() As Boolean – підключення інтерфейсу RAPI;

– Disconnect() As Boolean– відключення;

– Isconnected([Statusstring As String]) As Boolean – істина, якщо підключений;

– Runcmd(cmd_line As String, parameters As String, [Errorcode As Long], [Errorstring As String]) As Boolean – запускає зазначену в cmd_line програму із параметрами з parameters;

– Terminate() – завершення роботи програми, зручно застосовувати перед обнуліннням змінної device="".

Розроблений Remoteregistry клас для керування регістрами, властивості й методи:

– Createkey(Regkey As String, Valuename As String, Value, [dwtype As Long = -1], [Errorcode As Long], [Errorstring As String]) As Boolean створити ключ Regkey, якщо Valuename"" те буде створене значення Value;

– Deletekey(Regparentkeys As String, Subkeytodelete As String, [Errorcode As Long], [Errorstring As String]) As Boolean вилучити ключ Regkey з вітки Regparentkeys;

– Readkey(Regkey As String, Valuename As String, [Errorcode As Long], [Errorstring As String]) читати значення Valuename з Regkey;

– Setvalue(Regkey As String, Valuename As String, Value, [dwtype As Long = -1], [Errorcode As Long], [Errorstring As String]) As Boolean установити значення Value з іменем Valuename у вітці Regkey;

– Deletevalue(Regparentkeys As String, Valuenametodelete As String, [Errorcode As Long], [Errorstring As String]) As Boolean видалити значення з іменем Valuenametodelete у вітці Regkey.

При створенні ключа/значення створюються по всій ієрархії рядка Regkey, якщо відсутні, а якщо ні, то помилки не відбувається Regkey повинен починатися з "HKEY_CLASSES_ROOT", "HKEY_CURRENT_USER", "HKEY_LOCAL_ MACHINE" – інших віток реєстру.

– Fileexistondevice(Filename As String, [lhandle As Long], [Errorcode As Long = 0], [Errorstring As String]) As Boolean перевіряє існування файлу Filename;

– Copyfile(Existingfilename As String, Newfilename As String, [Replace As Boolean False], [Errorcode As Long], [Errorstring As String]) As Boolean копіює Existingfilename в Newfilename, якщо у Replace = 0 то заміняє, інакше – помилка;

– Movefile(Existingfilename As String, Newfilename As String, [Replace As Boolean = False], [Errorcode As Long], [Errorstring As String]) As Boolean переміщає Existingfilename в Newfilename, якщо Replace=0 то заміняє, інакше – помилка;

– Deletefile(Filetodevice As String, [Errorcode As Long], [Errorstring As String]) As Boolean видаляє Filetodevice;

– Createdirectory(Directoryname As String, [Errorcode As Long], [Errorstring As String]) As Boolean створює Directoryname, батьківські папки кінцевої в Directoryname повинні існувати;

– Removedirectory(Directoryname As String, [Errorcode As Long], [Errorstring As String]) As Boolean видаляє Directoryname;

– Copytodevice(Filetopc As String, Filetodevice As String, [Replace As Boolean = True], [Errorcode As Long], [Errorstring As String]) As Boolean копіює Filetopc із іменем Filetodevice, якщо Replace=0 то заміняє, інакше – помилка;

– Copytopc(Filetodevice As String, Filetopc As String, [Replace As Boolean = True], [Errorcode As Long], [Errorstring As String]) As Boolean копіює Filetodevice у PC із іменем Filetopc, якщо Replace=0 те заміняє, інакше – помилка;

– Getfilesize(Filename As String, [Errorcode As Long], [Errorstring As String]) As Long повертає розмір файлу Filename;

– Addfiletoexchange(Filetodevice As String, Filetopc As String, [Direction As Integer = 1], [Replace As Boolean = False], [Errorcode As Long], [Errorstring As String]) As Boolean додати файл Filetodevice/Filetopc до колекції підготовлених до обміну Direction – напрямок обміну, 1 – копіювати Filetopc із PC в Filetodevice 2 – копіювати Filetodevice в Filetopc у PC;

– Replace: якщо Replace=0 то заміняє, інакше – помилка;

– Filestoexchange Direction As Integer – напрямок обміну, Filetodevice As String – ім'я файлу;

– Filetopc As String – ім'я файлу в PC Replace As Boolean якщо Replace=0 то заміняє, інакше – помилка;

– Size As Long після обміну в Size буде записаний розмір фінального файлу Successfully As Boolean якщо Successfully=0 то обмін не зроблений;

– Exchange([Direction As Integer], [Errorcode As Long], [Errorstring As String]) As Boolean запускає обмін файлами з колекції Filestoexchange Clearfilestoexchange() очищає колекцію Filestoexchange;

– Lasterrorcode() As Long повертає код останньої помилки;

– Lasterrorstring() As String повертає опис останньої помилки.

Розглянемо копіювання файлів у бакалаврській роботі. Для копіювання файлів застосовується функція Copyfile. Формат її такої:

```
Copyfile(Вихідний_файл, Файл_запису,Прапор_перезапису);
```

де:

Вихідний_файл – повний шлях і назва файлу, який копіюється.

Файл_запису – повний шлях і назва файлу, куди копіюється вихідний файл.

Прапор_перезапису – буде чи ні перезаписаний файл, якщо такий уже існує (true – не буде, false – буде перезаписаний).

Copyfile є функцією, яка повертає прапор успішної чи ні операції копіювання.

У бакалаврській роботі я її реалізував як:

```
if not Copyfile('c:\Data\dat.rar', 'c:\Data\outputdata\dat.rar ',true)
       then Showmessage('Помилка копіювання');
```

Скопіює файл dat.rar у файл dat.rar тільки в тому випадку, якщо останнього ні, а якщо ні, то буде видаватися повідомлення про помилку копіювання.

У деяких випадках при реалізації було необхідно не копіювати файл, а переміщати його, тоді використовувалася функція Movefile.

Формат: Movefile(Вихідний_файл, файл_запису);.

Її параметри аналогічні команді Copyfile за винятком відсутності прапора перезапису.

Я її реалізував як:

```
if not Movefile ('c:\Data\dat.rar', 'c:\Data\outputdata\send\dat.rar') then
Showmessage('Помилка переміщення');
```

Коли файл знаходиться в архіву застосовувалася функція Renamefile (перейменування файлів) у вигляді запису:

```
if not Renamefile('c:\Data\input\dat04.rar', 'c:\Data\arxiv\AR_dat04.rar ')
                    then Showmessage ('Помилка перейменування');
```

У всіх розглянутих командах параметри вихідного й кінцевого файлу мають тип Pchar. Це строковий тип з нулем, що завершується. Її розмір обмежується наявністю нульового байта наприкінці. Теоретично така змінна може мати нескінченний розмір. На практиці вона обмежується розмірами виділюваної пам'яті для програми (2Гб) у бакалаврській роботі її розмір змінювався відповідно до прийнятого розміру файлу.

Коли було необхідно перетворити звичайну строкову змінну, типу String в Pchar я використовував функцію:

Pchar(Строкова_змінна).

При об'єднанні рядків виконувався код:

```
procedure Tform1.Button1Click(Sender: Tobject);
Var Indir,Outdir:String;
// оголошення строкових змінних
begin
Indir:= 'c:\Data\output\dat.rar'';
// каталог вихідних файлів
Outdir:= c:\Data';
// каталог файлів запису
Copyfile(Pchar(Indir+'1.txt'),Pchar(Outdir+'1.txt'),false);
Copyfile(Pchar(Indir+'2.txt'),Pchar(Outdir+'2.txt'),false);
end;
```

4.2 Захист розробленого програмного забезпечення

Дані у програмному забезпеченні я захищаю за допомогою NTRU. NTRUEncrypt (абревіатура Nth-degree TRUncated polynomial ring або Number Theorists aRe Us) – це криптографічна система з відкритим ключем, що раніше називалася NTRU.

Криптосистема NTRUEncrypt, заснована на ґратчастій криптосистемі, створена як альтернатива RSA і криптосистемам на еліптичних кривих (ECC). Стійкість алгоритму забезпечується труднощами пошуку найкоротшого вектора ґрати, що більше стійка до атак, здійснюваним на квантових комп'ютерах. На відміну від своїх конкурентів RSA,ECC, Elgamal, алгоритм використовує операції над кільцем:

$$\mathbb{Z}[X]/(X^N - 1),$$

усічених багаточленів ступеня, що не перевершує $N - 1$:

$$\mathbf{a}(X) = \mathbf{a} = a_0 + a_1 X + a_2 X^2 + \cdots + a_{N-2} X^{N-2} + a_{N-1} X^{N-1}.$$

Такий багаточлен можна також представити вектором:

$$\vec{a}(X) = \vec{a} = \sum_{i=0}^{N-1} a_i X^i = [a_0, a_1, a_2, \cdots, a_{N-2}, a_{N-1}]$$

Як і будь-який молодий алгоритм, NTRUEncrypt погано вивчений, хоча й був офіційно затверджений для використання в сфері фінансів комітетом Accredited Standards Committee X9.[1]

Існує реалізації NTRUEncrypt з відкритим вихідним кодом.[2]

NTRUEncrypt, що споконвічно називався NTRU, був винайдений в 1996 році й представлений на конференціях CRYPTO, Конференція RSA, Eurocrypt. Причиною, що послужила початком розробки алгоритму в 1994 році, стала стаття [3], у якій говорилося про легкість злому існуючих алгоритмів на квантових комп'ютерах, які, як показало час, не за горами[4]. У цьому ж році, математики Jeffrey Hoffstein, Jill Pipher і Joseph H. Silverman, що розробили систему разом із засновником компанії NTRU Cryptosystems, Inc. (пізніше перейменованої в SecurityInnovation), Даніелєм Лієманом (Daniel Lieman) запатентували свій винахід.[5]

Кільця усічених багаточленів

NTRU оперує над багаточленами ступеня не переважаючої $N-1$:

$$\mathbf{a} = a_0 + a_1 X + a_2 X^2 + \cdots + a_{N-2} X^{N-2} + a_{N-1} X^{N-1},$$

де коефіцієнти $a_0, \cdots, a_{N-1}$ – цілі числа. Щодо операцій додавання й множення за модулем багаточлена $X^N - 1$.Такі багаточлени утворять кільце *R*, називане кільцем усічених багаточленів, що ізоморфно кільцю відносин:

$$\mathbb{Z}[X]/(X^N - 1).$$

NTRU використовує кільце усічених багаточленів *R* разом з діленням за модулем на взаємно прості числа *p* і *q* для зменшення коефіцієнтів багаточленів.

У роботі алгоритму також використовуються зворотні багаточлени в кільці усічених багаточленів. Слід зазначити, що не всякий багаточлен має зворотний, але якщо зворотний поліном існує, то його легко обчислити.[6][7]

Генерація відкритого ключа

Для передачі повідомлення від Аліси до Боба необхідні відкритий і закритий ключі. Відкритий знають як Боб, так і Аліса, закритий ключ знає тільки Боб, що він використовує для генерації відкритого ключа. Для цього Боб вибирає два «маленьких» поліноми *f g* з *R*. «Малість» поліномів мається на увазі в тому розумінні, що він маленький щодо довільного полінома за модулем *q*: у довільному поліномі коефіцієнти повинні бути приблизно рівномірно розподілені за модулем *q*, а в малому поліномі вони багато менше *q*. Малість поліномів визначається за допомогою чисел *df* і *dg*:

Поліном *f* має *df* коефіцієнтів рівних «1» і *df*-1коефіцієнтів рівних «-1», а інші – «0». У цьому випадку говорять, що:

$$\mathbf{f} \in \mathcal{L}_f$$

Поліном *g* має *dg* коефіцієнтів рівних «1» і стільки ж рівних «-1», інші – «0» У цьому випадку говорять, що:

$$\mathbf{g} \in \mathcal{L}_g$$

Причина, по якій поліноми вибираються саме таким чином, полягає в тому, що *f* , можливо, буде мати зворотний, а *g* – однозначно немає (*g* (1) = 0, а нульовий елемент не має зворотного).

Боб повинен зберігати ці полінома в секреті. Далі Боб обчислює зворотні поліноми $\mathbf{f}_p$ й $\mathbf{f}_q$, тобто такі, що:

$$\mathbf{f} \cdot \mathbf{f}_p \equiv 1 \pmod{p} \text{ й } \mathbf{f} \cdot \mathbf{f}_q \equiv 1 \pmod{q}$$

Якщо *f* не має зворотного полінома, то Боб вибирає інший поліном *f*.

Секретний ключ – це пари $(\mathbf{f}, \mathbf{f}_p)$, а відкритий ключ *h* обчислюється за формулою:

$$\mathbf{h} = (p\mathbf{f}_q \cdot \mathbf{g}) \bmod q.$$

Приклад:

Для приклада візьмемо *df*=4, а *dg*=3. Тоді як поліноми можна вибрати

$$\mathbf{f} = -1 + X + X^2 - X^4 + X^6 + X^9 - X^{10}$$
$$\mathbf{g} = -1 + X^2 + X^3 + X^5 - X^8 - X^{10}$$

Далі для полінома *f* шукаються зворотні поліноми за модулем *p*=3 иq=32:

$$\mathbf{f}_p = 1 + 2X + 2X^3 + 2X^4 + X^5 + 2X^7 + X^8 + 2X^9$$

$$\mathbf{f}_q = 5+9X+6X^2+16X^3+4X^4+15X^5+16X^6+22X^7+20X^8+18X^9+30X^{10}$$

Заключним етапом є обчислення самого відкритого ключа *h*:

$$\mathbf{h} = (p\mathbf{f}_q \cdot \mathbf{g}) \bmod 32 = 8+25X+22X^2+20X^3+12X^4+24X^5+15X^6+19X^7+12X^8+19X^9+16X^{10}.$$

Шифрування

Тепер, коли в Аліси є відкритий ключ, вона може відправити зашифроване повідомлення Бобові. Для цього потрібно повідомлення представити у вигляді полінома m з коефіцієнтами за модулем *p*, обраними з діапазону (-*p*/2, *p*/2]. Тобто *m* є «малим» поліномом за модулем *q*. Далі Алісі необхідно вибрати інший «малий» поліном r, що називається «сліпучої», обумовлений за допомогою числа *dr*:

Поліном r має *dr* коефіцієнтів рівних «1» і стільки ж рівних «-1», інші – «0». У цьому випадку говорять, що:

$$\mathbf{r} \in \mathcal{L}_r.$$

Використовуючи ці поліноми, зашифроване повідомлення виходить за формулою:

$$\mathbf{e} = (\mathbf{r} \cdot \mathbf{h} + \mathbf{m}) \bmod q.$$

При цьому кожній, хто знає (або може обчислити) осліплюючий поліном r, зможе прочитати повідомлення *m*.

Приклад:

Припустимо, що Аліса хоче послати повідомлення, представлене у вигляді полінома:

$$\mathbf{m} = -1 + X^3 - X^4 - X^8 + X^9 + X^{10},$$

і вибрала «осліплюючий» поліном, для якого *dr*=3:

$$\mathbf{r} = -1 + X^2 + X^3 + X^4 - X^5 - X^7.$$

Тоді шифротекст *e*, готовий для передачі Бобові буде:

$$\mathbf{e} = (\mathbf{r}\cdot\mathbf{h}+\mathbf{m}) \bmod 32 = 14+11X+26X^2+24X^3+14X^4+16X^5+30X^6+7X^7+25X^8+6X^9+19X^{10}$$

Розшифрування

Тепер, одержавши зашифроване повідомлення *e*, Боб може його розшифрувати, використовуючи свій секретний ключ. Спочатку він одержує новий проміжний поліном:

$$\mathbf{a} = (\mathbf{f} \cdot \mathbf{e}) \bmod q.$$

Якщо розписати шифротекст, то одержимо ланцюжок:

$$\mathbf{a} = (\mathbf{f} \cdot \mathbf{e}) \bmod q = (\mathbf{f} \cdot (\mathbf{r} \cdot \mathbf{h} + \mathbf{m})) \bmod q = (\mathbf{f} \cdot (\mathbf{r} \cdot p\mathbf{f}_q \cdot \mathbf{g} + \mathbf{m})) \bmod q$$

і остаточно:

$$\mathbf{a} = (p\mathbf{r} \cdot \mathbf{g} + \mathbf{f} \cdot \mathbf{m}) \bmod q.$$

Після того, як Боб обчислив поліном a за модулем *q*, він повинен вибрати його коефіцієнти з діапазону (-*q*/2, *q*/2] і далі обчислити поліном b, одержуваний з полінома a приведенням за модулем *p*:

$$\mathbf{b} = \mathbf{a} \bmod p = (\mathbf{f} \cdot \mathbf{m}) \bmod p,$$

так як:

$$(p\mathbf{r} \cdot \mathbf{g}) \bmod p = 0.$$

Тепер, використовуючи другу половину секретного ключа й отриманий поліном *b*, Боб може розшифрувати повідомлення:

$$\mathbf{c} = (\mathbf{f}_p \cdot \mathbf{b}) \bmod p.$$

Неважко бачити, що:

$$\mathbf{c} \equiv \mathbf{f}_p \cdot \mathbf{f} \cdot \mathbf{m} \equiv \mathbf{m} \pmod p.$$

У такий спосіб отриманий поліном *c* дійсно є вихідним повідомленням *m*.

Приклад:

Боб одержав від Аліси шифроване повідомлення *e*:

$$\mathbf{e} = 14+11X+26X^2+24X^3+14X^4+16X^5+30X^6+7X^7+25X^8+6X^9+19X^{10}$$

Використовуючи секретний ключ *f* Боб одержує поліном *a*:

$$\mathbf{a} = \mathbf{f}\cdot\mathbf{e} \pmod{32} = 3-7X-10X^2-11X^3+10X^4+7X^5+6X^6+7X^7+5X^8-3X^9-7X^{10} \pmod{32},$$

з коефіцієнтами, що належать проміжку (-*q*/2, *q*/2]. Далі перетворить поліном *a* у поліном *b*, зменшуючи коефіцієнти за модулем *p*.

$$\mathbf{b} = \mathbf{a} \pmod 3 = -X - X^2 + X^3 + X^4 + X^5 + X^7 - X^8 - X^{10} \pmod 3$$

Заключний крок – перемножування полінома *b* із другою половиною закритого ключа $\mathbf{f}_p$:

$$\mathbf{c} = \mathbf{f}_p \cdot \mathbf{b} = \mathbf{f}_p \cdot \mathbf{f} \cdot \mathbf{m} \pmod 3 = \mathbf{m} \pmod 3$$

$$\mathbf{c} = -1 + X^3 - X^4 - X^8 + X^9 + X^{10}$$

Який є вихідним повідомленням, що передавала Аліса.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програма має простий та інтуїтивно зрозумілий інтерфейс, який зображений на рисунку 5.1. З нього видно, що інтерфейс користувача програми складається з таких логічних блоків:

– Меню: Файл; Мат. Модель; Налаштування; Допомога.

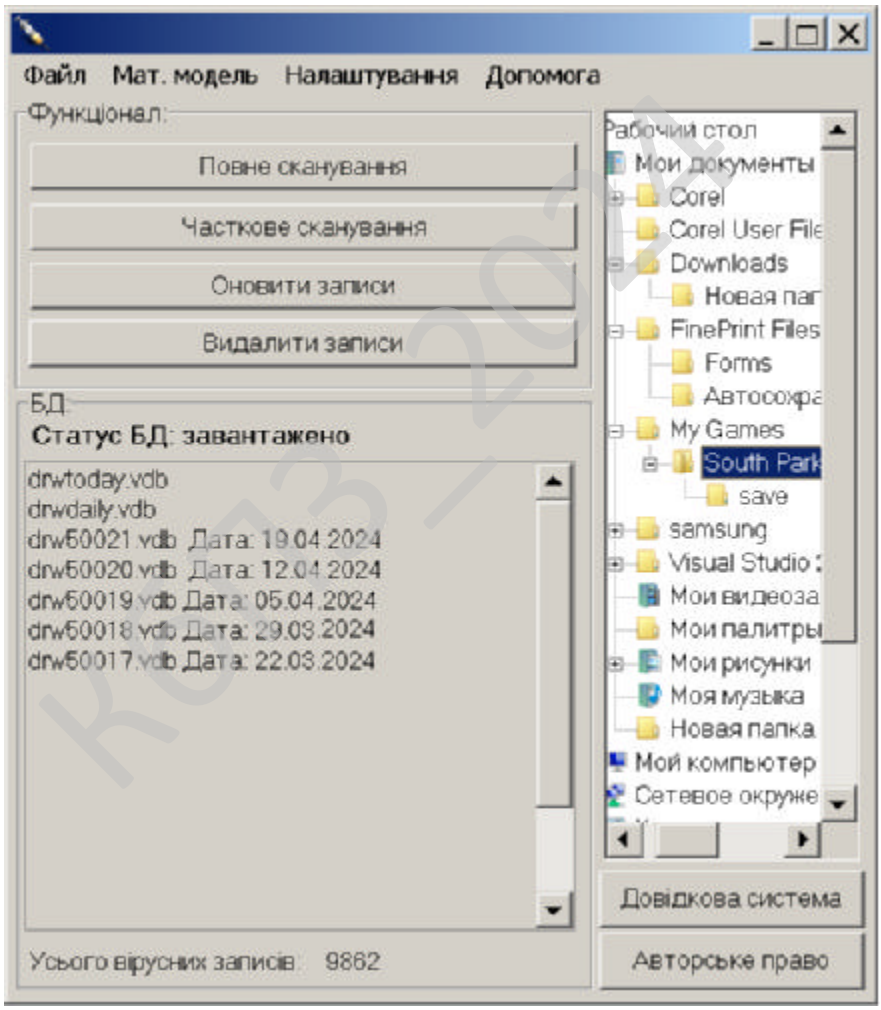

Рисунок 5.1 – Головне вікно програми

– Функціонал: Повне сканування; Часткове сканування; Оновити записи; Видалити записи.

– БД: Список файлів та статус БД.

– Кнопок: Довідкова система; Авторське право.

На рисунку 5.2 зображено форму авторського права. Була вибрана ліцензія Freeware. При розробці ПЗ була створена форма авторського права з ліцензій на використання програмного забезпечення. Ліцензія на використання програмного забезпечення це вид ліцензії, що визначає умови використання виробу, яким є комп'ютерне програмне забезпечення. Ліцензія може надавати дозвіл робити з ним речі, які були б інакше заборонені законом про авторське право. Наприклад, ліцензія на використання програмного забезпечення може дати дозвіл робити копії програмного забезпечення. Власник авторського права може запропонувати ліцензію на використання ПЗ односторонньо, або як частину ліцензійної угоди на використання ПЗ з іншою стороною.

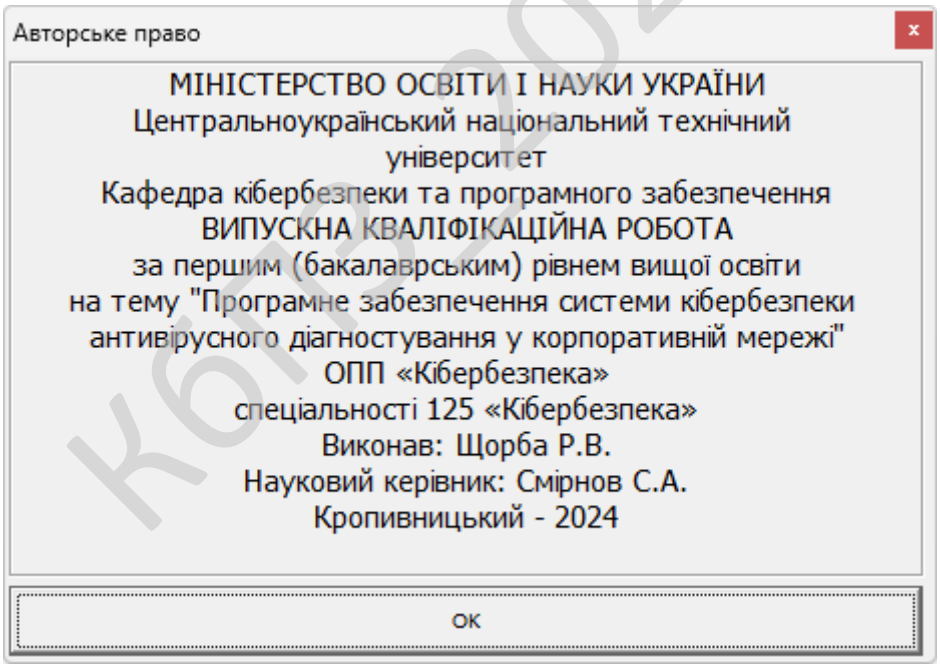

Рисунок 5.2 – Довідка

6 ОСНОВНІ ВИСНОВКИ

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

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 діагностування у корпоративній мережі.

– Досліджена система антивірусного діагностування у корпоративній мережі.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки антивірусного діагностування у корпоративній мережі.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання антивірусного діагностування у корпоративній мережі.

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 діагностування у корпоративній мережі. 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 NTRUEncrypt.

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. Loren Kohnfelder. Designing Secure Software. No Starch Press. 2022. 332 p.

2. Samir Kumar Rakshit. Ethical Hacker's Penetration Testing Guide. BPB Online. 2022. 509 p.

3. Corey J. Ball. Hacking APIs. No Starch Press. 2022. 353 p.

4. Kevin Beaver. Hacking for Dummies. John Wiley & Sons. 2022. 419 p.

5. Mark S. Merkow. Practical Security for Agile and DevOps. CRC Press. 2022. 236 p

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

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

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

9. 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.

10. 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.

11. 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.

12. 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,

13. 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.

14. 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.

15. 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

16. 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.

17. 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.

18. 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.*

19. 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.

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

21. 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.

22. 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.

23. 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.

24. 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.

25. 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.

26. 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.

27. 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.

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

35. 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.

36. 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.

37. 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.

38. 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.

39. 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.

40. 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.

41. 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.

42. 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.

43. 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.

44. 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.

45. 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.

46. 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.

47. 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.

48. 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.

49. 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.

50. 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.

51. 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.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки антивірусного діагностування у корпоративній мережі.

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 діагностування у корпоративній мережі.

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 діагностування у корпоративній мережі;

– 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 – 81 аркуш.

8 Етапи розробки

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

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

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

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

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

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 6.06.2024 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 С.А.

Програмне забезпечення системи кібербезпеки антивірусного діагностування у корпоративній мережі

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 50

Літера: РП

Кропивницький – 2024 року

ОСНОВНА ПРОГРАМА
ФАЙЛ РОЗРОБЛЕНОГО ПРОЕКТУ ПЗ KOMBO_KOMBO_ANTIVIRUS.DPR

```
program KOMBO_KOMBO_ANTIVIRUS; // Початок файлу проекту ПЗ

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки
антивірусного діагностування у корпоративній мережі
Виконав: студент 4 курсу,
Щорба Роман Віталійович, КБ-20
Керівник:
Смірнов С.А.
2024 рік
}

Uses
// Підключення бібліотек
  Forms,
  Messages,Graphics,
  Windows,
  Unit1 in 'Unit1.pas' {Form1},
  Unit2 in 'Report.pas' {Form2},
  Unit3 in 'fset.pas' {Form3},
  KOMBO_KOMBO_ANTIVIRUS_splash in 'U7.pas' {KOMBO_KOMBO_ANTIVIRUS_Form_Splash}.

{$R *.res}
// Підключення ресурсів

begin
  Application.HintPause:=10;
  // Чекання при виведенні довідки ПЗ
  Application.HintHidePause:=1000;
  Application.Initialize; //Ініціалізіція ПЗ
try
  // Створення Splash форми
KOMBO_KOMBO_ANTIVIRUS_Form_Splash:=TKOMBO_KOMBO_ANTIVIRUS_Form_Splash.
                                    Create(Application);
    KOMBO_KOMBO_ANTIVIRUS_Form_Splash.Show;
// виведення Splash форми
    KOMBO_KOMBO_ANTIVIRUS_Form_Splash.Update;
// оновлення Splash форми
  // відправлення системного повідомлення
  Sendmessage(KOMBO_KOMBO_ANTIVIRUS_Form_Splash.Handle,WM_MY,0,'Start');
  Application.CreateForm(TForm1, Form1);
// Підключення форми 1
  Application.CreateForm(TForm2, Form2);
// Підключення форми 2
  Application.CreateForm(TForm3, Form3);
// Підключення форми 3
  // відправлення системного повідомлення
```

```
  Sendmessage(KOMBO_KOMBO_ANTIVIRUS_Form_Splash.Handle,WM_MY,0,'End');
Application.CreateForm(TForm5, Form4);
// Підключення форми 4
Finally KOMBO_KOMBO_ANTIVIRUS_Form_Splash.free;
// звільнення Splash форми
end;
  Application.Run;
// виведення на екран головної форми ПЗ
end.
// Кінець файлу проекту ПЗ
```

ГОЛОВНИЙ ФАЙЛ ФОРМИ ПЗ FORM1.PAS

```
unit Form1;
// Початок файлу форми

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки антивірусного діагностування у
корпоративній мережі
Виконав: студент 4 курсу,
Щорба Роман Віталійович, КБ-20
Керівник:
Смірнов С.А.
2024 рік
}

interface // Секція інтерфейсу (зовнішніх визначень модуля)

Uses // Підключення бібліотек
SysUtils, IniFiles, Forms, registry, Windows, Dialogs, Menus,
IdAntiFreezeBase, IdAntiFreeze, IdCustomTCPServer, IdCustomHTTPServer,
IdHTTPServer, IdBaseComponent, IdComponent, IdTCPConnection, IdTCPClient,
IdHTTP, StdCtrls, ExtCtrls, Controls, ComCtrls, Classes, idContext,
IdAuthentication, srchunit, FileCtrl, TLHelp32, ShellApi, psapi, Graphics,
StrUtils, shlobj, idstack, mapi, Messages;

const //введення констант
  WM_ICONTRAY = WM_USER + 1; // системне повідомлення

Type // Визначення типів
  pfilestream = ^TFileStream;

Type // Визначення типів
 TKOMBO_KOMBO_ANTIVIRUS_main = class (TForm)
    pc:      TPageControl;
    TabSheet2: TTabSheet;
    Label1:  TLabel; // текстові дані
    Label2:  TLabel; // текстові дані
    netbase: TEdit; // компонент текстового введення даних
    locbase: TEdit; // компонент текстового введення даних
    Button2: TButton; // компонент кнопки
    Button1: TButton; // компонент кнопки
    TabSheet3: TTabSheet;
    Label3:  TLabel; // текстові дані
    msk:     TEdit; // компонент текстового введення даних
    TabSheet4: TTabSheet;
    TabSheet5: TTabSheet;
    id:      TIdHTTP;
    TabSheet1: TTabSheet;
    Button3: TButton; // компонент кнопки
logwindow: TRichEdit;
    htest:   TButton; // компонент кнопки
```

```
hstop:   TButton; // компонент кнопки
Label8:  TLabel; // текстові дані
over:    TCheckBox; // компонент введення (так/ні)
_ver:    TEdit; // компонент текстового введення даних
notdel:  TCheckBox; // компонент введення (так/ні)
http:    TIdHTTPServer;
TabSheet6: TTabSheet;
GroupBox3: TGroupBox; // Група компонентів
_uselochttp: TCheckBox; // компонент введення (так/ні)
Label19: TLabel; // текстові дані
_lochttpport: TEdit; // компонент текстового введення даних
deli:    TButton; // компонент кнопки
ignor:   TComboBox;
addit:   TButton; // компонент кнопки
Label6:  TLabel; // текстові дані
always:  TComboBox;
addb:    TButton; // компонент кнопки
Button8: TButton; // компонент кнопки
_http:   TCheckBox; // компонент введення (так/ні)
Status:  TStatusBar;
maxcon:  TEdit; // компонент текстового введення даних
Label4:  TLabel; // текстові дані
TabSheet7: TTabSheet;
GroupBox4: TGroupBox; // Група компонентів
Label10: TLabel; // текстові дані
Label11: TLabel; // текстові дані
Label7:  TLabel; // текстові дані
Label9:  TLabel; // текстові дані
proxy:   TEdit; // компонент текстового введення даних
pauth:   TCheckBox; // компонент введення (так/ні)
ppass:   TEdit; // компонент текстового введення даних
plogin:  TEdit; // компонент текстового введення даних
pport:   TEdit; // компонент текстового введення даних
GroupBox5: TGroupBox; // Група компонентів
hauth:   TCheckBox; // компонент введення (так/ні)
hlogin:  TEdit; // компонент текстового введення даних
Label20: TLabel; // текстові дані
hpass:   TEdit; // компонент текстового введення даних
Label22: TLabel; // текстові дані
huser:   TEdit; // компонент текстового введення даних
Label23: TLabel; // текстові дані
IdAntiFreeze1: TIdAntiFreeze;
TabSheet8: TTabSheet;
GroupBox6: TGroupBox; // Група компонентів
fnoerr:  TCheckBox; // компонент введення (так/ні)
fevent:  TCheckBox; // компонент введення (так/ні)
fnoerrlocb: TCheckBox; // компонент введення (так/ні)
ferr:    TCheckBox; // компонент введення (так/ні)
GroupBox7: TGroupBox; // Група компонентів
flnoerr: TCheckBox; // компонент введення (так/ні)
flevent: TCheckBox; // компонент введення (так/ні)
flnoerrlocb: TCheckBox; // компонент введення (так/ні)
flerr:   TCheckBox; // компонент введення (так/ні)
TabSheet9: TTabSheet;
Memo2:   TMemo;
Label25: TLabel; // текстові дані
```

```
Label26: TLabel; // текстові дані
Label27: TLabel; // текстові дані
detect:  TButton; // компонент кнопки
findweb: TButton; // компонент кнопки
fullsearch: TCheckBox; // компонент введення (так/ні)
sresults: TListBox;
sstop:   TButton; // компонент кнопки
where:   TComboBox;
PopupMenu1: TPopupMenu; // Контексне меню ПЗ
N1:      TMenuItem; // Елемент меню
N2:      TMenuItem; // Елемент меню
OpenDialog: TOpenDialog;
PopupMenu2: TPopupMenu; // Контексне меню ПЗ
MenuItem1: TMenuItem; // Елемент меню
MenuItem2: TMenuItem; // Елемент меню
txt:     TCheckBox; // компонент введення (так/ні)
err:     TCheckBox; // компонент введення (так/ні)
errd:    TCheckBox; // компонент введення (так/ні)
GroupBox9: TGroupBox; // Група компонентів
Label21: TLabel; // текстові дані
GroupBox10: TGroupBox; // Група компонентів
newhttp: TCheckBox; // компонент введення (так/ні)
imp:     TButton; // компонент кнопки
exp:     TButton; // компонент кнопки
Button10: TButton; // компонент кнопки
SaveDialog: TSaveDialog;
TabSheet10: TTabSheet;
GroupBox11: TGroupBox; // Група компонентів
run:     TEdit; // компонент текстового введення даних
multin:  TCheckBox; // компонент введення (так/ні)
mdirs:   TComboBox;
chkpath: TButton; // компонент кнопки
zapinet: TCheckBox; // компонент введення (так/ні)
zapnet:  TCheckBox; // компонент введення (так/ні)
getrec:  TButton; // компонент кнопки
traymenu: TPopupMenu; // Контексне меню ПЗ
N8:      TMenuItem; // Елемент меню
N9:      TMenuItem; // Елемент меню
N10:     TMenuItem; // Елемент меню
N11:     TMenuItem; // Елемент меню
N12:     TMenuItem; // Елемент меню
N13:     TMenuItem; // Елемент меню
N14:     TMenuItem; // Елемент меню
N15:     TMenuItem; // Елемент меню
N16:     TMenuItem; // Елемент меню
N17:     TMenuItem; // Елемент меню
N18:     TMenuItem; // Елемент меню
N19:     TMenuItem; // Елемент меню
N20:     TMenuItem; // Елемент меню
N21:     TMenuItem; // Елемент меню
TabSheet19: TTabSheet;
avpath:  TEdit; // компонент текстового введення даних
Label28: TLabel; // текстові дані
avscanbrowse: TButton; // компонент кнопки
servname: TEdit; // компонент текстового введення даних
Label5:  TLabel; // текстові дані
```

```
    artyp:   TRadioGroup;
    Button4: TButton; // компонент кнопки
    Button5: TButton; // компонент кнопки
    aredit:  TEdit; // компонент текстового введення даних
    GroupBox1: TGroupBox; // Група компонентів
    agshow:  TRadioGroup;
    firstz:  TCheckBox; // компонент введення (так/ні)
  end;

Var // Розділ визначення змінних
  logfilepath: string = '';
  isauto: boolean = False;
  shutdown: boolean = False;
  trayst: boolean = False;
  nt: boolean = False;
  KOMBO_KOMBO_ANTIVIRUS: TKOMBO_KOMBO_ANTIVIRUS_main ;
  st: TSearchThread;
  LogFile: TextFile;
  uselog: boolean = False;
  maxlog: longint = 2097152;
  logonlyerrors: boolean = False;
  rewr:   boolean = False;
  mypath: string = '';
  started: tdatetime = -1;
  marked: boolean = False;
  updateok: boolean = True;
  Colors: array[False..True] of TColor;
  Windowsdir: string = '';
  Tempdir: string = '';
  sessiontotal: integer = 0;

Var // Розділ визначення змінних
  NotifyIconData: TNotifyIconData;
  nowintray: boolean = False;
  buildstamp: string = '100';

implementation // Секція реалізації

Uses // Підключення бібліотек
sntpsend, blcksock;

{$R *.dfm} // Підключення ресурсів модуля

Var // Розділ визначення змінних
  stop: boolean = False;

function InternetGetConnectedState(lpdwFlags: LPDWORD; dwReserved:DWORD):BOOL;
stdcall; external 'wininet.dll' Name 'InternetGetConnectedState';

function FileSizeByName(const AFilename: string): int64;
begin
  with TFileStream.Create(AFilename, fmOpenRead or fmShareDenyWrite) do
    try
      Result:= Size;
    finally
      Free;
```

```
     end;
end;

function InternetConnected: boolean;
Var // Розділ визначення змінних
  dwConnectionTypes: DWORD;
begin
  dwConnectionTypes:= 7;
  if KOMBO_KOMBO_ANTIVIRUS.nonet.Checked then
    Result:= True
  else
    Result:= InternetGetConnectedState(@dwConnectionTypes, 0);
end;

function SyncSysTime;
Var // Розділ визначення змінних
  sn: TSntpSend;
begin
  if not InternetConnected then
   begin
    Log_KOMBO_KOMBO_ANTIVIRUS(' Не виявлене підключення до мережі!', 255, 3);
    status.Panels[1].Text:= 'Помилка.';
    Result:= False;
    exit;
   end;
  sn:= TSntpSend.Create;
  sn.TargetHost:= times.Text;
  sn.SyncTime:= not test;
  case socksproxy.ItemIndex
    of
    0:
     begin
      sn.Sock.SocksType:= ST_Socks5;
     end;
    1:
     begin
      sn.Sock.SocksType:= ST_Socks4;
     end;
    2:
     begin
      sn.Sock.SocksType:= ST_Socks4;
      sn.Sock.SocksResolver:= True;
     end;
   end;
  sn.Sock.SocksIP:= proxy.Text;
sn.Sock.SocksPort:= IntToStr(strtointdef(socksport.Text, 1080));
  if socksneedpass.Checked then
   begin
    sn.Sock.SocksUsername:= socksuser.Text;
    sn.Sock.SocksPassword:= sockspass.Text;
   end;
  case prot.ItemIndex of
    0:
      Result:= sn.GetSNTP;
    1:
      Result:= sn.GetNTP;
```

```
   end;
  if Result then
   begin
    if not Test then
Log_KOMBO_KOMBO_ANTIVIRUS('Успішно синхронізований час із сервера ' +
sn.Targethost, 0, 2); Log_KOMBO_KOMBO_ANTIVIRUS(Formatdatetime('Час, отриман із
сервера: dddd, d mmmm yyyy, hh:nn:ss:z', sn.Ntptime), 0, 1);
   end
  else
   begin
Log_KOMBO_KOMBO_ANTIVIRUS('Помилка синхронізації часу із сервера ' +
sn.Targethost, 255, 3);
Log_KOMBO_KOMBO_ANTIVIRUS('Опис помилки: ' + sn.Sock.Lasterrordesc, 255, 3);
    status.Panels[1].Text:= 'Помилка.';
   end;
  sn.Free;
end;

function copyfileadv(from, tof: string): boolean;
begin
  if from = tof then
   begin
    Result:= False;
    exit;
   end;
  if fileexists(tof) then
   begin
    Result:= Filesetreadonly(tof, False);
    if Result then
  Log_KOMBO_KOMBO_ANTIVIRUS(Format('Підготовлений до видалення файл %s', [tof]),
0, 0)
    else
  Log_KOMBO_ANTIVIRUS(Format('Немає доступу до файлу %s', [tof]), 255, 3);
    if Result then
      Result:= Windows.Deletefile(Pchar(tof));
    if Result then
Log_KOMBO_ANTIVIRUS(Format('Вилучений раніше існуючий файл %s', [tof]), 0, 0)
    else
  Log_KOMBO_ANTIVIRUS(Format('Немає доступу до файлу %s', [tof]), 255, 3);
   end
  else
    Result:= True;
  if Result then
    Result:= Windows.Copyfile(Pchar(from), Pchar(tof), False);
  if Result then
Log_KOMBO_ANTIVIRUS(Format('Скопійований %s в %s', [from, tof]), 0, 0)
  else
Log_KOMBO_ANTIVIRUS(Format('Помилка копіювання %s в %s', [from, tof]), 255, 3);
end;

function httpgetfilename(s: string): string;
Var // Розділ визначення змінних
  i: integer;
begin
  Result:= '';
  for i:= length(s)
```

```
    downto 1 do
    if s[i] <> '/' then
      Result:= s[i] + Result
    else
      break;
end;

function cutfirst(s: string): string;
begin
  Result:= s;
  Delete(Result, 1, 1);
end;

function getprocessfilename(Proc: TPROCESSENTRY32): string;
Var // Розділ визначення змінних
  ph: Thandle;
  md: array[0..1023] of Thandle;
  sz: array[0..255] of char;
  n:  dword;
  i:  longint;
begin
  Result:= '';
  ph:= Openprocess(PROCESS_QUERY_INFORMATION or PROCESS_VM_READ,
    False, proc.th32Processid);
  if (ph <> INVALID_HANDLE_VALUE) then
    try
      if Enumprocessmodules(ph, (@md), Sizeof(md), n) then
       begin
        for i:= 0 to n div Sizeof(Thandle) - 1 do
         begin
          if Getmodulefilenameex(ph, md[i], Pchar(@sz), MAX_PATH) <> 0 then
           begin
            if i = 0 then
              Result:= sz;
           end;
         end;
       end;
    finally
      Closehandle(ph);
     end;
end;

procedure .Handleerror(e, l: integer; fn: string; Ex: Exception);

  function oneline(s: string): string;
  begin
    Result:= strutils.Ansireplacetext(s, #13#10, ' ');
  end;

begin
  if ex is Eidsocketerror then
    if Syserrormessage((ex as Eidsocketerror).Lasterror) <> '' then
  Log_KOMBO_ANTIVIRUS(Format('Помилка #%d ''%s'''', [(ex as
Eidsocketerror).Lasterror,
        Syserrormessage((ex as Eidsocketerror).Lasterror)]), e, l)
    else
```

```
  Log_KOMBO_ANTIVIRUS(Format('Помилка #%d ''%s''', [(ex as
Eidsocketerror).Lasterror,
        (oneline((ex as Eidsocketerror).Message))]), e, l)
  else
  if ex is Eidhttpprotocolexception then
Log_KOMBO_ANTIVIRUS(Format('Помилка #%d ''%s'' при одержанні файлу ''%s''',
      [(ex as Eidhttpprotocolexception).Errorcode, oneline(
      (ex as Eidhttpprotocolexception).Message), fn]), e, l)
  else
Log_KOMBO_ANTIVIRUS(Format('Помилка ''%s''', [oneline(ex.Message)]), e, l);
end;

function Spider: string;
Var // Розділ визначення змінних
  cp: cardinal;
  pe: TProcessEntry32;
  n:  string;
begin
  Result:= '';
  pe.dwSize:= sizeof(TProcessEntry32);
  cp:= CreateToolHelp32Snapshot(TH32CS_SnapProcess, 0);
  try
    if Process32First(cp, pe) then
      repeat
        if NT then
          n:= getprocessfilename(pe)
        else
          n:= pe.szExeFile;
        if (AnsiUpperCase(ExtractFileName(n)) = 'SPIDER.EXE') or
          (AnsiUpperCase(ExtractFileName(n)) = 'SPIDERNT.EXE') then
          if fileexists(ExtractFilePath(n)) then
           begin
            Result:= ExtractFilePath(n);
            break;
           end;
      until not Process32Next(cp, pe);
  finally
    CloseHandle(cp);
   end;
  if Result = '' then
   begin
    if FileExists(ExtractFileDrive(WindowsDir) then
Result:= ExtractFileDrive(WindowsDir);
   end;
end;

procedure .RunAndWait(cmdline: string);
Var // Розділ визначення змінних
  si: STARTUPINFO;
  pi: PROCESS_INFORMATION;
begin
  ZeroMemory(@si, sizeof(si));
  si.cb:= SizeOf(si);
  if not CreateProcess(nil, // створення потоку обробки ActiveSync
    PChar(cmdline), nil, nil, False, 0,nil,nil,si,pi) then
   begin
```

```
    ShowMessage('CreateProcess failed.');
    Exit;
   end;
  WaitForSingleObject(pi.hProcess, INFINITE);
  CloseHandle(pi.hProcess);
  CloseHandle(pi.hThread);
end;

procedure .Syntax;
begin
  socksport.Text:= IntToStr(strtointdef(socksport.Text, 1080));
  _maxlog.Text:= IntToStr(strtointdef(_maxlog.Text, 2097152));
  netbase.Text:= Trim(netbase.Text);
  if netbase.Text <> '' then
    netbase.Text:= Sysutils.Includetrailingbackslash(netbase.Text);
  locbase.Text:= Trim(locbase.Text);
  if locbase.Text <> '' then
    locbase.Text:= Sysutils.Includetrailingbackslash(locbase.Text);
  _lochttpport.Text:= Inttostr(strtointdef(_lochttpport.Text, 16555));
  dir.Text:= Trim(Dir.Text);
  dir.Text:= Ansireplacetext(dir.Text, '\', '/');
  if dir.Text <> '' then
    if dir.Text[length(dir.Text)] <> '/' then
      dir.Text:= dir.Text + '/';
end;

procedure .install(Id, Run: string);
Var // Розділ визначення змінних
  reg: Tregistry;
begin
  try
reg:= Tregistry.Create;
reg.Rootkey:= hkey_current_user;
reg.Openkey('\SOFTWARE\Microsoft\Windows\Currentversion\Run', True);
    reg.Writestring(Id, run);
Log_KOMBO_ANTIVIRUS('Інсталяція: ' + reg.Currentpath + ', Ідентифікатор: ' +
      id + ', Рядок запуску: ' + run, 0, 2);
  finally
    reg.Free;
   end;
end;

procedure TKOMBO_ANTIVIRUS_main.deinstall(id: string);
Var // Розділ визначення змінних
  reg: Tregistry;
begin
  try
    reg:= Tregistry.Create;
reg.Rootkey:= hkey_current_user;
reg.Openkey('\SOFTWARE\Microsoft\Windows\Currentversion\Run', True);
    if reg.Valueexists(id) then
     begin
      if reg.Deletevalue(id) then
    Log_KOMBO_ANTIVIRUS('Ключ успішно вилучений.', 0, 2)
      else
    Log_KOMBO_ANTIVIRUS('Не вдається вилучити ключ.', 255, 3);
```

```
     end
    else
  Log_KOMBO_ANTIVIRUS('Ключ не існує.', 1, 3)
  finally
    reg.Free;
   end;
end;

procedure TKOMBO_ANTIVIRUS_main.Log_KOMBO_ANTIVIRUS(s: string; ok, lev: byte);
  procedure Addline(l: string);
  begin
    KOMBO_ANTIVIRUS.logwindow.Lines.Append(l);
    if updex.Checked then
      KOMBO_ANTIVIRUS.logwindow.Update;
  end;

  procedure markdate;
  begin
    if marked then
      exit;
writeln(logfile, '--- Новий сеанс ---');
writeln(logfile, Formatdatetime('Початок звіту: hh:nn:ss, DDDDDD', now));
    marked:= True;
  end;

begin
  if (ok = 0) and (not fnoerr.Checked) and (lev >= loglevel.Itemindex) then
   begin
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Color:= clblack;
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Style:= [fsbold];
    Addline(s);
   end
  else
if (ok = 255) and (not ferr.Checked) and (lev >= loglevel.Itemindex) then
   begin
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Color:= clred;
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Style:= [fsbold];
    Addline(s);
   end
  else
  if (ok = 1) and (not fevent.Checked) and (lev >= loglevel.Itemindex) then
   begin
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Color:= clgreen;
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Style:= [fsbold];
    Addline(s);
   end
  else
  if (ok = 2) and (not fnoerrlocb.Checked) and (lev >= loglevel.Itemindex) then
   begin
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Color:= clblue;
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Style:= [fsbold];
    Addline(s);
   end
  else
  if (ok = 3) and (not fnoerrmult.Checked) and (lev >= loglevel.Itemindex) then
   begin
```

```
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Color:= clpurple;
    KOMBO_ANTIVIRUS.logwindow.Selattributes.Style:= [fsbold];
    Addline(s);
   end;


  if (uselog) and (lev >= filelevel.Itemindex) then
   begin
    case ok of
      0:
        if not flnoerr.Checked then
         begin
          markdate;
          writeln(logfile, s);
         end;
      1:
        if not flevent.Checked then
         begin
          markdate;
          writeln(logfile, s);
         end;

      2:
        if not flnoerrlocb.Checked then
         begin
          markdate;
          writeln(logfile, s);
         end;

      3:
        if not flnoerrmult.Checked then
         begin
          markdate;
          writeln(logfile, s);
         end;

      255:
        if not flerr.Checked then
         begin
          markdate;
          writeln(logfile, s);
         end;

     end;
   end;
end;

procedure TKOMBO_ANTIVIRUS_main.Button1Click(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
  s:= Userselectedfolder('Вкажіть потрібний шлях');
  if s <> '' then
  netbase.Text:= s + '\';
end;
```

```
procedure TKOMBO_ANTIVIRUS_main.lanupdate(Sender: Tobject);
Var // Розділ визначення змінних
  r: boolean;
begin
  logwindow.Clear;
  Log_KOMBO_ANTIVIRUS('Мережне відновлення..', 0, 0);
  syntax;
  button3.Enabled:= False;
  if (trim(netbase.Text) <> '') and (trim(locbase.Text) <> '') then
   begin
    if not pupdate(Pchar(netbase.Text), Pchar(locbase.Text), Pchar(msk.Text))
then
     begin
Log_KOMBO_ANTIVIRUS('Відновлення пройшло з помилками!', 255, 3);
      if errd.Checked then
       begin
Showmessage('Відновлення пройшло з помилками! Необхідно Подивиться звіт.');
        if agshow.Itemindex <> 2 then
          gofromtray;
       end
      else
      if nowintray then
       begin
        Showmessage('Відновлення пройшло з помилками!');
        if agshow.Itemindex <> 2 then
          gofromtray;
       end;
     end
    else
     begin
      if nowintray then
       begin
        Showmessage('Відновлення успішне завершене!');
        if agshow.Itemindex = 1 then
          gofromtray;
       end;
      if arun.Checked then
       begin
        if not ask.Checked then
          r:= True
        else
r:= messagedlg('Відновлення успішне завершене. Запустити програму?',
mtconfirmation, [mbok, mbcancel], 0) = idok;
        if r then
          if wait.Checked then
            runandwait(run.Text)
          else
            winexec(Pchar(run.Text), sw_show);
       end;
      if autoexit.Checked then
       begin
        if not askex.Checked then
          r:= True
        else
r:=messagedlg('Вийти із програми?', mtconfirmation, [mbok,mbcancel], 0) = idok;
        if r then
```

```
         begin
          application.Processmessages;
          Close;
         end;
       end;

     end;

   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Укажіть шлях для відновлення!', 255, 3);
    Showmessage('Укажіть шлях для відновлення!');
Log_KOMBO_ANTIVIRUS('Відновлення пройшло з помилками!', 255, 3);
    if errd.Checked then
     begin
Showmessage('Відновлення пройшло з помилками! Подивитеся звіт.');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end
    else
    if nowintray then
     begin
      Showmessage('Відновлення пройшло з помилками!');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end;
    pc.Activepage:= Tabsheet2;
   end;
  button3.Enabled:= True;
end;
procedure TKOMBO_ANTIVIRUS_main.Formclose(Sender: Tobject; var Action:
Tcloseaction);
Var // Розділ визначення змінних
  ini: Tinifile;
begin

  if (tray.Itemindex > 1) and (not shutdown) and (not isauto) then
   begin
    gototray;
    Action:= canone;
    exit;
   end;
  syntax;
  ini.Writebool('main', 'agentrestoretofirsttab', firstz.Checked);
  ini.Writeinteger('main', 'raisewindowinagentmode', agshow.Itemindex);
  ini.Writestring('main', 'avpath', avpath.Text);
  ini.Writeinteger('main', 'traymode33', tray.Itemindex);
  ini.Writebool('main', 'cheatgetvirrecfornetupdate', zapnet.Checked);
  ini.Writebool('main', 'cheatgetvirrecforinetupdate', zapinet.Checked);

  ini.Writebool('main', 'enablemulticopyforinet', multii.Checked);
  ini.Writebool('main', 'enablemulticopyfornet', multin.Checked);
  ini.Writestring('main', 'logfilepath', logpath.Text);
  ini.Writeinteger('main', 'webgo', webgo.Itemindex);
  ini.Writeinteger('main', 'sockstype', socksproxy.Itemindex);
```

```
  ini.Writestring('main', 'socksport', socksport.Text);
  ini.Writebool('main', 'socksneedpassword', socksneedpass.Checked);
  ini.Writestring('main', 'socksuser', socksuser.Text);
  ini.Writestring('main', 'sockspassword', sockspass.Text);
  ini.Writebool('main', 'disablenetcheck', nonet.Checked);
  ini.Writebool('main', 'syncsystime', synctime.Checked);
  ini.Writestring('main', 'timeserver', times.Text);
  ini.Writeinteger('main', 'timeprotocol', prot.Itemindex);
  ini.Writestring('main', 'timeservers', times.Items.Commatext);
  ini.Writebool('main', 'timecritical', timeerr.Checked);
  ini.Writebool('main', 'advancedwindowupdate', updex.Checked);
  ini.Writebool('main', 'safeautoupdate', comp.Checked);
  ini.Writebool('main', 'downloadother', downa.Checked);
  ini.Writestring('main', 'otherfiles', other.Items.Commatext);
  ini.Writebool('main', 'autorun', arun.Checked);
  ini.Writebool('main', 'autorunask', ask.Checked);
  ini.Writebool('main', 'autorunexitask', askex.Checked);
  ini.Writebool('main', 'autoexit', autoexit.Checked);
  ini.Writebool('main', 'autorunwait', wait.Checked);
  ini.Writestring('main', 'inetservers33', dir.Items.Commatext);
  ini.Writestring('main', 'localhttpignorelist', ignor.Items.Commatext);
  ini.Writestring('main', 'alwaysupdate33', always.Items.Commatext);
  mdirs.Items.Append('(eol)');
  ini.Writestring('main', 'multicopydirs', mdirs.Items.Commatext);
  ini.Writestring('main', 'localpath', locbase.Text);
  ini.Writestring('main', 'autoruncmdline', run.Text);
  ini.Writestring('main', 'remotepath', netbase.Text);
  ini.Writestring('main', 'vdbmask', msk.Text);
  ini.Writestring('main', 'servname', servname.Text);
  ini.Writestring('main', 'httpdir33', dir.Text);
  ini.Writestring('main', 'proxy', proxy.Text);
  ini.Writestring('main', 'proxyport', pport.Text);
  ini.Writestring('main', 'proxyuser', plogin.Text);
  ini.Writestring('main', 'proxypassword', ppass.Text);
  ini.Writebool('main', 'debugautoerrors', err.Checked);
  ini.Writebool('main', 'debugerrors', errd.Checked);
  ini.Writebool('main', 'log', _uselog.Checked);
  ini.Writebool('main', 'httpnewprotocol33', newhttp.Checked);
  ini.Writebool('main', 'httpneedsverifing', hauth.Checked);
  ini.Writestring('main', 'httplogin', hlogin.Text);
  ini.Writestring('main', 'httppassword', hpass.Text);
  ini.Writestring('main', 'httpuseragent', huser.Text);
  ini.Writeinteger('main', 'maxlog', strtointdef(_maxlog.Text, 2097152));
  ini.Writebool('main', 'localhttpdir', _http.Checked);
  ini.Writebool('main', 'overrideversion', over.Checked);
  ini.Writebool('main', 'alwaysrewritelog', _rewr.Checked);
  ini.Writebool('main', 'keepinstalledfilescopies', notdel.Checked);
  ini.Writestring('main', 'useversion33', _ver.Text);
  ini.Writeinteger('main', 'localhttpport', strtointdef(_lochttpport.Text,
16555));
  ini.Writeinteger('main', 'maxserverconnections', strtointdef(maxcon.Text, 0));
  ini.Writebool('main', 'inetgettxt', txt.Checked);
  ini.Writebool('main', 'uselocalhttpserver', _uselochttp.Checked);
  ini.Writebool('main', 'proxyneedsverification', pauth.Checked);
  ini.Writebool('main', 'filternoerror', fnoerr.Checked);
  ini.Writebool('main', 'filterevent', fevent.Checked);
```

```
  ini.Writebool('main', 'filtererror', ferr.Checked);
  ini.Writebool('main', 'filternoerrorlocb', fnoerrlocb.Checked);
  ini.Writebool('main', 'filternoerrormcop', fnoerrmult.Checked);
  ini.Writebool('main', 'logfilternoerror', flnoerr.Checked);
  ini.Writebool('main', 'logfilterevent', flevent.Checked);
  ini.Writebool('main', 'logfiltererror', flerr.Checked);
  ini.Writebool('main', 'logfilternoerrorlocb', flnoerrlocb.Checked);
  ini.Writebool('main', 'logfilternoerrormcop', flnoerrmult.Checked);
  ini.Writeinteger('main', 'loglevel', loglevel.Itemindex);
  ini.Writeinteger('main', 'logfilelevel33', filelevel.Itemindex);
  ini.Free;
  if uselog then
    closefile(logfile);
end;

procedure TKOMBO_ANTIVIRUS_main.Formactivate(Sender: Tobject);
Var // Розділ визначення змінних
  ini:   Tinifile;
  dr:    dword;
  i, dt: integer;
  r:     Tregistry;
begin
  Onactivate:= nil;
  started:= now;
  dr:= getlogicaldrives;
  for i:= 0 to 31 do
    if (dr shr i) and 1 = 1 then
     begin
dt:= getdrivetype(Pchar(char(byte('A') + i) + ':\'));
if (dt <> DRIVE_CDROM) and (dt <> DRIVE_REMOVABLE) and (dt <> DRIVE_RAMDISK)
then
        where.Items.Append(char(byte('A') + i));
     end;
  where.Itemindex:= 0;

  mypath:= extractfilepath(application.Exename);
  pc.Activepageindex:= 0;
  ini:= Tinifile.Create(windowsdir + 'drupdate.ini');
  firstz.Checked:= ini.Readbool('main', 'agentrestoretofirsttab', True);
  agshow.Itemindex:= ini.Readinteger('main', 'raisewindowinagentmode', 0);
  tray.Itemindex:= ini.Readinteger('main', 'traymode33', 0);
  zapnet.Checked:= ini.Readbool('main', 'cheatgetvirrecfornetupdate', False);
  zapinet.Checked:= ini.Readbool('main', 'cheatgetvirrecforinetupdate', False);
  multii.Checked:= ini.Readbool('main', 'enablemulticopyforinet', False);
  multin.Checked:= ini.Readbool('main', 'enablemulticopyfornet', False);
  logpath.Text:= ini.Readstring('main', 'logfilepath', windowsdir +
'drupdate.log');
  webgo.Itemindex:= ini.Readinteger('main', 'webgo', 0);
  socksproxy.Itemindex:= ini.Readinteger('main', 'sockstype', 0);
  socksport.Text:= ini.Readstring('main', 'socksport', '1080');
  socksneedpass.Checked:= ini.Readbool('main', 'socksneedpassword', False);
  socksuser.Text:= ini.Readstring('main', 'socksuser', '');
  sockspass.Text:= ini.Readstring('main', 'sockspassword', '');
  synctime.Checked:= ini.Readbool('main', 'syncsystime', synctime.Checked);
  times.Text:= ini.Readstring('main', 'timeserver', 'clock.via.net');
  prot.Itemindex:= ini.Readinteger('main', 'timeprotocol', 0);
```

```
  times.Items.Commatext:= ini.Readstring('main', 'timeservers',
times.Items.Commatext);
  timeerr.Checked:= ini.Readbool('main', 'timecritical', False);
  comp.Checked:= ini.Readbool('main', 'safeautoupdate', False);
  updex.Checked:= ini.Readbool('main', 'advancedwindowupdate', False);
  downa.Checked:= ini.Readbool('main', 'downloadother', False);
  other.Items.Commatext:= ini.Readstring('main', 'otherfiles',
other.Items.Commatext);
  dir.Items.Commatext:= ini.Readstring('main', 'inetservers33',
dir.Items.Commatext);
  ignor.Items.Commatext:= ini.Readstring('main', 'localhttpignorelist',
'drwebase.vdb');
  always.Items.Commatext:= ini.Readstring('main', 'alwaysupdate33',

'drwrisky.vdb,drwrisky.txt,drwnasty.vdb,drwnasty.txt,drwtoday.vdb,drwtoday.txt,d
wrtoday.vdb,dwrtoday.txt,dwntoday.vdb,dwntoday.txt');
  mdirs.Items.Commatext:= ini.Readstring('main', 'multicopydirs',
    mdirs.Items.Commatext);
  if mdirs.Items.Count > 0 then
    mdirs.Items.Delete(mdirs.Items.Count - 1);
  locbase.Text:= ini.Readstring('main', 'localpath', '');
  if locbase.Text = '' then
    locbase.Text:= spider;
  avpath.Text:= ini.Readstring('main', 'avpath', locbase.Text);
  run.Text:= ini.Readstring('main', 'autoruncmdline', run.Text);
  if run.Text = '' then
    if directoryexists(Sysutils.Excludetrailingbackslash(avpath.Text)) then
      run.Text:= Sysutils.Includetrailingbackslash(avpath.Text);
  arun.Checked:= ini.Readbool('main', 'autorun', False);
  ask.Checked:= ini.Readbool('main', 'autorunask', True);
  askex.Checked:= ini.Readbool('main', 'autorunexitask', askex.Checked);
  autoexit.Checked:= ini.Readbool('main', 'autoexit', False);
  wait.Checked:= ini.Readbool('main', 'autorunwait', False);
  netbase.Text:= ini.Readstring('main', 'remotepath', netbase.Text);
  msk.Text:= ini.Readstring('main', 'vdbmask', '*.vdb');
  servname.Text:= ini.Readstring('main', 'servname', 'drupdate');
dir.Text:= ini.Readstring('main',
  proxy.Text:= ini.Readstring('main', 'proxy', '*');
  if proxy.Text = '*' then
    detectproxy(False)
  else
   begin
    pport.Text:= ini.Readstring('main', 'proxyport', '80');
   end;
  plogin.Text:= ini.Readstring('main', 'proxyuser', '');
  ppass.Text:= ini.Readstring('main', 'proxypassword', '');
  txt.Checked:= ini.Readbool('main', 'inetgettxt', False);
nonet.Checked:= ini.Readbool('main', 'disablenetcheck', False);
  pauth.Checked:= ini.Readbool('main', 'proxyneedsverification', False);
  fnoerr.Checked:= ini.Readbool('main', 'filternoerror', False);
  fevent.Checked:= ini.Readbool('main', 'filterevent', False);
  ferr.Checked:= ini.Readbool('main', 'filtererror', False);
  fnoerrlocb.Checked:= ini.Readbool('main', 'filternoerrorlocb', False);
  fnoerrmult.Checked:= ini.Readbool('main', 'filternoerrormcop', False);
  flnoerr.Checked:= ini.Readbool('main', 'logfilternoerror', False);
  flevent.Checked:= ini.Readbool('main', 'logfilterevent', False);
```

```
  flerr.Checked:= ini.Readbool('main', 'logfiltererror', False);
  flnoerrlocb.Checked:= ini.Readbool('main', 'logfilternoerrorlocb', False);
  flnoerrmult.Checked:= ini.Readbool('main', 'logfilternoerrormcop', False);
  newhttp.Checked:= ini.Readbool('main', 'httpnewprotocol33', False);
  hauth.Checked:= ini.Readbool('main', 'httpneedsverifing', False);
  hlogin.Text:= ini.Readstring('main', 'httplogin', '');
  hpass.Text:= ini.Readstring('main', 'httppassword', '');
  huser.Text:= ini.Readstring('main', 'httpuseragent', '');
  if huser.Text = '' then
   begin
    r:= Tregistry.Create;
    r.Rootkey:= HKEY_CURRENT_USER;
    r.Openkey('\Software\Microsoft\Windows\Currentversion\Internet Settings',
True);
    huser.Text:= r.Readstring('User Agent');
    r.Closekey;
    r.Free;
   end;
  uselog:= ini.Readbool('main', 'log', False);
  rewr:= ini.Readbool('main', 'alwaysrewritelog', False);
  _rewr.Checked:= rewr;
  notdel.Checked:= ini.Readbool('main', 'keepinstalledfilescopies', False);
  logonlyerrors:= ini.Readbool('main', 'logonlyerrors', False);
  _uselog.Checked:= uselog;
  maxlog:= ini.Readinteger('main', 'maxlog', 2097152);
  _maxlog.Text:= Inttostr(maxlog);
  _maxlog.Enabled:= uselog;
  over.Checked:= ini.Readbool('main', 'overrideversion', False);
  _ver.Text:= ini.Readstring('main', 'useversion33', getver);
  http.Defaultport:= ini.Readinteger('main', 'localhttpport', 16555);
  http.Maxconnections:= ini.Readinteger('main', 'maxserverconnections', 0);
  loglevel.Itemindex:= ini.Readinteger('main', 'loglevel', 2);
  filelevel.Itemindex:= ini.Readinteger('main', 'logfilelevel33', 0);
  maxcon.Text:= Inttostr(http.Maxconnections);
  _lochttpport.Text:= Inttostr(http.Defaultport);
  _uselochttp.Checked:= ini.Readbool('main', 'uselocalhttpserver', False);
  _http.Checked:= ini.Readbool('main', 'localhttpdir', False);
  err.Checked:= ini.Readbool('main', 'debugautoerrors', True);
  errd.Checked:= ini.Readbool('main', 'debugerrors', True);
  if _uselochttp.Checked then
   begin
    _http.Checked:= True;
    _http.Enabled:= False;
   end;
  if uselog then
   begin
logfilepath:= logpath.Text;
    if not fileexists(logfilepath) then
     begin
      assignfile(logfile, logfilepath);
      rewrite(logfile);
      uselog:= ioresult = 0;
     end
    else
     begin
      assignfile(logfile, logfilepath);
```

```
      if (filesizebyname(logfilepath) > maxlog) or (rewr) then
       begin
        rewrite(logfile);
        uselog:= ioresult = 0;

       end
      else
       begin
        append(logfile);
        uselog:= ioresult = 0;
       end;
     end;
   end;
  ini.Free;
  if (ansilowercase(Paramstr(1)) = '-time') then
   begin
    isauto:= True;
    if not comp.Checked then
      application.Minimize;
    updateok:= Syncsystime(False);
    if not err.Checked then
     begin
      application.Processmessages;
      Close;
     end
    else
     begin
      if updateok then
       begin
        application.Processmessages;
        Close;
       end
      else
       begin
        application.Restore;
if messagedlg('Помилка синхронізації часу! Однаково вийти із програми?',
mtconfirmation, [mbok, mbcancel], 0) = idok then
         begin
          application.Processmessages;
          Close;
         end;
       end;

     end;

   end;

  if (ansilowercase(Paramstr(1)) = '-auto') or
    ((ansilowercase(Paramstr(1)) = '/go') and (webgo.Itemindex = 1)) then
   begin
    isauto:= True;
    if not comp.Checked then
      application.Minimize;
    updateok:= pupdate(Pchar(netbase.Text), Pchar(locbase.Text),
Pchar(msk.Text));
```

```
    if not err.Checked then
     begin
      application.Processmessages;
      Close;
     end
    else
     begin
      if updateok then
       begin
        application.Processmessages;
        Close;
       end
      else
       begin
        application.Restore;
if messagedlg('Відновлення пройшло з помилками! Однаково вийти із програми?',
mtconfirmation, [mbok, mbcancel], 0) = idok then
         begin
          application.Processmessages;
          Close;
         end;
       end;

     end;

   end;

  if (ansilowercase(Paramstr(1)) = '-autohttp') or
    ((ansilowercase(Paramstr(1)) = '/go') and (webgo.Itemindex = 2)) then
   begin
    isauto:= True;
    if not comp.Checked then
      application.Minimize;
    updateok:= inetupd;
    if not err.Checked then
     begin
      application.Processmessages;
      Close;
     end
    else
     begin
      if updateok then
       begin
        application.Processmessages;
        Close;
       end
      else
       begin
        application.Restore;
if messagedlg('Відновлення пройшло з помилками! Однаково вийти із програми?',
mtconfirmation, [mbok, mbcancel], 0) = idok then
         begin
          application.Processmessages;
          Close;
         end;
       end;
```

```
     end;

   end;
  if _uselochttp.Checked then
    http.Active:= True;

end;

function TKOMBO_ANTIVIRUS_main.inetupd: boolean;
Var // Розділ визначення змінних
  str:   Tfilestream;
  ini:   Tmeminifile;
  sec, upd, new: Tstringlist;
  last, ext: string;
  i, n:  integer;
  Count: longint;
begin
  if locbase.Text = '' then
   begin
Log_KOMBO_ANTIVIRUS('Укажіть шлях до локальних баз!', 255, 3);
    Result:= False;
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end;
  if (not directoryexists(Sysutils.Excludetrailingpathdelimiter(locbase.Text)))
then
   begin
Log_KOMBO_ANTIVIRUS(Format('Каталог локальних баз "%s" не існує. Перевірте
настроювання.',
      [Sysutils.Excludetrailingpathdelimiter(locbase.Text)]), 255, 3);
    Result:= False;
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end;
  if not Internetconnected then
   begin
Log_KOMBO_ANTIVIRUS('Не виявлене підключення до мережі!', 255, 3);
    Result:= False;
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end;
  Result:= True;
  status.Panels[0].Text:= '';
  status.Panels[1].Text:= '';
  Count:= 0;
  sessiontotal:= 0;
  htest.Enabled:= False;
  hstop.Enabled:= True;
  stop:= False;
  logwindow.Lines.Clear;
  if KOMBO_ANTIVIRUS.synctime.Checked then
   begin
Log_KOMBO_ANTIVIRUS('Синхронізація часу...', 0, 0);
    if timeerr.Checked then
     begin
      if not syncsystime(False) then
```

```
        Result:= False;
      end
     else
       syncsystime(False);
   end;
  id.Proxyparams.Proxyport:= strtointdef(pport.Text, 80);
  id.Proxyparams.Proxyserver:= proxy.Text;
  id.Proxyparams.Proxypassword:= ppass.Text;
  id.Proxyparams.Proxyusername:= plogin.Text;
  id.Proxyparams.Basicauthentication:= pauth.Checked;
  id.Request.Useragent:= huser.Text;
  id.Request.Basicauthentication:= hauth.Checked;
  id.Request.Username:= hlogin.Text;
  id.Request.Password:= hpass.Text;
  id.Request.Cachecontrol:= '"no-cache"';
  if newhttp.Checked then
    id.Protocolversion:= pv1_1
  else
    id.Protocolversion:= pv1_0;

  Log_KOMBO_ANTIVIRUS('Адреса відновлення: ' + dir.Text, 0, 1);
  if id.Protocolversion = pv1_0 then
Log_KOMBO_ANTIVIRUS('Версія протоколу: HTTP 1.0', 0, 1)
  else
Log_KOMBO_ANTIVIRUS('Версія протоколу: HTTP 1.1', 0, 1);
  if trim(proxy.Text) <> '' then
Log_KOMBO_ANTIVIRUS(Format('Прокси: %s:%d', [id.Proxyparams.Proxyserver,
id.Proxyparams.Proxyport]),
      0, 0);
  Log_KOMBO_ANTIVIRUS('User-Agent: ' + id.Request.Useragent, 0, 0);
  Log_KOMBO_ANTIVIRUS('Тимчасовий каталог: ' + tempdir, 0, 0);
  str:= Tfilestream.Create(tempdir + 'http_update.tmp', fmcreate);
  try
    Getfile(dir.Text + 'drweb32.lst', @str);
  except
    on E: Exception do
     begin
      Indyreport;
      KOMBO_ANTIVIRUS.Handleerror(255, 3, id.Request.URL, E);
      htest.Enabled:= True;
      hstop.Enabled:= False;
      str.Free;
      Result:= False;
      status.Panels[1].Text:= 'Помилка.';
      exit;
     end;
   end;
  Log_KOMBO_ANTIVIRUS('Отриманий drweb32.lst', 0, 0);
  Inc(sessiontotal, str.Size);
  str.Free;
  Log_KOMBO_ANTIVIRUS('Відповідь: ' + id.Response.Responsetext, 0, 0);
  ini:= Tmeminifile.Create(tempdir + 'http_update.tmp');
  sec:= Tstringlist.Create;
  upd:= Tstringlist.Create;
  new:= Tstringlist.Create;
  ini.Readsections(sec);
```

```
  if sec.Count > 0 then
   begin
    sec.Sort;
Log_KOMBO_ANTIVIRUS('Не знайдені секції: ', 0, 0);
    for i:= 0 to sec.Count - 1 do
  Log_KOMBO_ANTIVIRUS(format('%d: [%s]', [i, sec[i]]), 0, 0);
    if sec.Count = 1 then
      last:= sec[0]
    else
      last:= Inttostr(findnewver(sec));
   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Помилка у форматі drweb32.lst!', 255, 3);
    Result:= False;
   end;

  if over.Checked then
   begin
    last:= _ver.Text;
Log_KOMBO_ANTIVIRUS('Версія встановлена користувачем: ' + last, 0, 1);
   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Знайдена остання версія: ' + last, 0, 1);
   end;

  ini.Readsectionvalues(last, sec);
  ini.Free;
  if notdel.Checked = False then
    deletefile(Pchar(tempdir + 'http_update.tmp'));

  if sec.Count > 0 then
   begin

    for i:= 0 to sec.Count - 1 do
      if sec[i][1] = '+' then
       begin
        sec[i]:= copy(sec[i], 2, pos(',', sec[i]) - 2);
        ext:= ansilowercase(extractfileext(sec[i]));
        if (ext = '.vdb') or ((ext = '.txt') and (txt.Checked)) then
          if not fileexists(locbase.Text + sec[i]) or
            (always.Items.Indexof(sec[i]) <> -1) then
            upd.Append(sec[i])
          else
        Log_KOMBO_ANTIVIRUS('Уже встановлена:' + sec[i], 1, 1)

        else
      Log_KOMBO_ANTIVIRUS('Ігноруємо неабазу:' + sec[i], 1, 0);

       end;
    upd.Sort; //сортування
   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Даних за версією ' + last + ' не знайдене!', 255, 3);
```

```
    Result:= False;
   end;

  sec.Free;
  Log_KOMBO_ANTIVIRUS('Отриманий список файлів:', 0, 1);
  if downa.Checked then
   begin
    for i:= 0 to other.Items.Count - 1 do
      if upd.Indexof(httpgetfilename(other.Items[i])) = -1 then
        upd.Append('+' + other.Items[i])
      else
  Log_KOMBO_ANTIVIRUS('Зігнорований: ' + other.Items[i] + '. Уже в черзі.', 0,
3);

   end;
  for i:= 0 to upd.Count - 1 do
    if upd[i][1] <> '+' then
  Log_KOMBO_ANTIVIRUS(upd[i], 0, 1)
    else
  Log_KOMBO_ANTIVIRUS('Зовнішній: ' + cutfirst(upd[i]), 0, 1);
  if not (directoryexists(tempdir + 'drup_tmp') or Createdir(tempdir +
'drup_tmp')) then
   begin
    stop:= True;
Log_KOMBO_ANTIVIRUS('Помилка ініціалізації тимчасового каталогу ' + tempdir +
'drup_tmp', 255, 3);
   end;
  for i:= 0 to
    upd.Count - 1 do
   begin
    if stop then
     begin
  Log_KOMBO_ANTIVIRUS('Зупинка', 255, 3);
      Result:= False;
      break;
     end;
    if upd[i][1] = '+' then
  Log_KOMBO_ANTIVIRUS('Одержуємо файл: ' + cutfirst(upd[i]), 0, 1)
    else
  Log_KOMBO_ANTIVIRUS('Одержуємо файл: ' + upd[i], 0, 1);
    status.Panels[0].Text:= Format('Одержуємо %d з %d', [i + 1, upd.Count]);
    if upd[i][1] = '+' then
      str:= Tfilestream.Create(tempdir + 'drup_tmp\' +
        httpgetfilename(upd[i]), fmcreate)
    else
      str:= Tfilestream.Create(tempdir + 'drup_tmp\' + upd[i], fmcreate);
    application.Processmessages;
    try
      if upd[i][1] = '+' then
        Getfile(cutfirst(upd[i]), @str)
      else
        Getfile(dir.Text + upd[i], @str);
    except
      on E: Exception do
       begin
        Indyreport;
```

```
        KOMBO_ANTIVIRUS.Handleerror(255, 3, upd[i], E);
        str.Free;
        Result:= False;
        stop:= True;
        break;
       end;
     end;
    Inc(sessiontotal, str.Size);
    str.Free;
    if upd[i][1] = '+' then
     begin
      new.Append(cutfirst(upd[i]));
  Log_KOMBO_ANTIVIRUS('Отриманий файл: ' + cutfirst(upd[i]), 0, 2);
     end
    else
     begin
      new.Append(upd[i]);
  Log_KOMBO_ANTIVIRUS('Отриманий файл: ' + upd[i], 0, 2);
     end;

   end;

  for n:= 0 to new.Count - 1 do
    if new[n][1] = '+' then
      new[n]:= httpgetfilename(cutfirst(new[n]));

  if stop then
   begin
Log_KOMBO_ANTIVIRUS(Format('Отримані не всі файли (%d з %d)', [new.Count,
upd.Count]), 255, 3);
    Result:= False;
   end;


  for i:= 0 to new.Count - 1 do
   begin
    status.Panels[0].Text:= Format('Встановлення %d з %d', [i + 1, new.Count]);
    if copyfileadv(tempdir + 'drup_tmp\' + new[i], locbase.Text + new[i]) then
     begin
  Log_KOMBO_ANTIVIRUS('Встановлено: ' + new[i], 1, 2);
      Inc(Count);
     end
    else
     begin
  Log_KOMBO_ANTIVIRUS('Помилка копіювання: ' + new[i], 255, 3);
      Result:= False;
     end;

    if notdel.Checked = False then
      if deletefile(Pchar(tempdir + 'drup_tmp\' + new[i])) then
    Log_KOMBO_ANTIVIRUS('Вилучений тимчасовий файл: ' + new[i], 0, 0)
      else
       begin
    Log_KOMBO_ANTIVIRUS('Помилка видалення: ' + new[i], 255, 3);
        Result:= False;
       end;
```

```
   end;
  status.Panels[1].Text:= Format('Результат %d з %d', [Count, new.Count]);
  if upd.Count = Count then
Log_KOMBO_ANTIVIRUS('Нових баз усього: ' + Inttostr(upd.Count) +
      ', усі бази успішно встановлені.', 0, 2)
  else
   begin
Log_KOMBO_ANTIVIRUS(Format('Нових баз на сервері: %d Отримане із сервера: %d
Установлене: %d',
      [upd.Count, new.Count, Count]), 255, 3);
    Result:= False;
   end;
  upd.Free;
  new.Free;
  if (_http.Checked) and (Result) then
   begin
Log_KOMBO_ANTIVIRUS('Створюємо каталог для локального http відновлення.', 0, 1);
    makehttpdir(last);
   end;
  if (multii.Checked) and (Result) then
    if not multicopy then
      Result:= False;
  if Result = True then
   begin
    Getupdateflag;
    if zapinet.Checked then
      Getvirrec;
   end;
  Log_KOMBO_ANTIVIRUS(format('Отримане даних: %d байт (%f Кб)', [sessiontotal,
    sessiontotal / 1024]), 0, 2);
  htest.Enabled:= True;
  hstop.Enabled:= False;
end;

procedure TKOMBO_ANTIVIRUS_main.stopinetupdate(Sender: Tobject);
begin
  if id.Connected then
    try
      id.Disconnect;
    except
     end;
  stop:= True;
  Log_KOMBO_ANTIVIRUS('Перериваємо процес...', 255, 3);
end;

function TKOMBO_ANTIVIRUS_main.Userselectedfolder(title: string): string;
Var // Розділ визначення змінних
  Bi:  _browseinfoa;
  idl: Pitemidlist;
  Str: array[0..260 - 1] of char;
begin
  Fillchar(Bi, Sizeof(Bi), 0);
  Bi.hwndowner:= KOMBO_ANTIVIRUS.Handle;
  Bi.lpsztitle:= Pchar(title);
  Bi.ulflags:= 0;
```

```
  idl:= Shbrowseforfolder(Bi);
  Shgetpathfromidlist(idl, @Str[0]);
  Result:= Str;
end;

function TKOMBO_ANTIVIRUS_main.pupdate(fs, fd, mask: string): boolean;
Var // Розділ визначення змінних
  S: Tsearchrec;
  all, new, bad, ignor: integer;
begin
  if (trim(netbase.Text) = '') or (trim(locbase.Text) = '') then
   begin
Log_KOMBO_ANTIVIRUS('Укажіть шляху для відновлення!', 255, 3);
    Result:= False;
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end;
  if (not directoryexists(Sysutils.Excludetrailingpathdelimiter(fd))) then
   begin
Log_KOMBO_ANTIVIRUS(Format('Каталог локальних баз "%s" не існує. Перевірте
налаштування.',
      [Sysutils.Excludetrailingpathdelimiter(fd)]), 255, 3);
    Result:= False;
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end;
  status.Panels[0].Text:= '';
  status.Panels[1].Text:= '';
  Result:= True;
  all:= 0;
  new:= 0;
  ignor:= 0;
  bad:= 0;
  fs:= Pchar(includetrailingbackslash(fs));
  fd:= Pchar(includetrailingbackslash(fd));
  Log_KOMBO_ANTIVIRUS('Використовуємо сервер: ' + fs, 0, 1);
  if not directoryexists(fs) then
   begin
    Result:= False;
Log_KOMBO_ANTIVIRUS('Помилка підключення (каталог не існує): ' + fs, 255, 3);
    status.Panels[1].Text:= 'Помилка.';
    exit;
   end
  else
Log_KOMBO_ANTIVIRUS('Підключений: ' + fs, 0, 0);

  if Findfirst(fs + string(mask), fahidden + fasysfile + fareadonly +
    faarchive, S) = 0 then
    repeat
      Inc(all);
  Log_KOMBO_ANTIVIRUS('Знайдений: ' + s.Name, 0, 0);
      if (not fileexists(Pchar(fd + s.Name))) or
        (always.Items.Indexof(s.Name) <> -1) then

       begin
        if copyfileadv(fs + s.Name, fd + s.Name) then
```

```
         begin
      Log_KOMBO_ANTIVIRUS('Скопійований: ' + s.Name, 0, 2);
          Inc(new);
         end
        else
         begin
Log_KOMBO_ANTIVIRUS('Збій копіювання: ' + fs + s.Name + ' в ' + fd + s.Name,
255, 3);
          Inc(bad);
         end;
        status.Panels[0].Text:= Format('Скопійоване: %d Помилок: %d', [new,
bad]);
       end
      else
       begin
    Log_KOMBO_ANTIVIRUS('Уже встановлений: ' + s.Name, 1, 0);
        Inc(ignor);
       end;
    until Findnext(s) <> 0;
  Sysutils.Findclose(s);
  Log_KOMBO_ANTIVIRUS('Нових баз: ' + Inttostr(new), 0, 2);
  Log_KOMBO_ANTIVIRUS('Усього баз: ' + Inttostr(all), 0, 0);
  status.Panels[1].Text:= Format('Усього: %d Нових: %d Пропущене: %d',
    [all, new, ignor]);
  if all = 0 then
   begin
Log_KOMBO_ANTIVIRUS('Бази не знайдені!', 255, 3);
    Result:= False;
   end;


  if multin.Checked then
    if not multicopy then
      Result:= False;
  if bad > 0 then
    Result:= False;

  if Result = True then
   begin
    Getupdateflag;
    if zapnet.Checked then
      Getvirrec;
   end;
end;


procedure TKOMBO_ANTIVIRUS_main._uselogclick(Sender: Tobject);
begin
  _maxlog.Enabled:= _uselog.Checked;
  _rewr.Enabled:= _uselog.Checked;
  _maxlog.Color:= colors[_uselog.Checked];
  logpath.Enabled:= _uselog.Checked;
  logpath.Color:= colors[_uselog.Checked];
  logp.Enabled:= _uselog.Checked;
  report.Enabled:= _uselog.Checked;
  ;
```

```
end;

procedure TKOMBO_ANTIVIRUS_main.overclick(Sender: Tobject);
begin
  _ver.Enabled:= over.Checked;
  _ver.Color:= colors[over.Checked];
end;

procedure TKOMBO_ANTIVIRUS_main.makehttpdir(ver: string);
Var // Розділ визначення змінних
  s: Tsearchrec;
  list, index: Textfile;
begin
  createdir(mypath + 'http');
  assignfile(index, mypath + 'http\index.html');
  rewrite(list);
  rewrite(index);
  writeln(index, '<html>');
  writeln(index, '<head>');
  writeln(index, '</head>');
  writeln(index, '<body>');
  writeln(index, 'Current version: ', ver, '<br>');
  writeln(index, '<table border="1">');
  writeln(index, '<tr>');
  writeln(index, '<th>File</th>');
  writeln(index, '<th>Size</th>');
  writeln(index, '<th>Date</th>');
  writeln(index, '</tr>');
  writeln(list, '[', ver, ']');
  if findfirst(locbase.Text + msk.Text, faanyfile + fahidden +
    fasysfile + faarchive + fareadonly, s) = 0 then
    repeat
      if ignor.Items.Indexof(s.Name) <> -1 then
        continue;
      writeln(list, '+', s.Name, ', 0');
  Log_KOMBO_ANTIVIRUS('Locb: Копіюємо базу: ' + s.Name, 2, 0);
      if copyfileadv(locbase.Text + s.Name, mypath + 'http\' + s.Name) then
       begin
writeln(index, Format('<tr><td><a
href="\%s">%s</a><br></td><td>%d</td><td>%s</td></tr>',
 [s.Name, s.Name, S.Size, atetostr(Filedatetodatetime(S.Time))]));
    Log_KOMBO_ANTIVIRUS('Locb: Успішно скопійована: ' + s.Name, 2, 1);
       end
      else
    Log_KOMBO_ANTIVIRUS('Locb: Помилка копіювання: ' + s.Name, 255, 3);
    until findnext(s) <> 0;
  Sysutils.Findclose(s);
  closefile(list);
  writeln(index, '</table>');
  writeln(index, '</body>');
  writeln(index, '</html>');
  closefile(index);
end;


procedure TKOMBO_ANTIVIRUS_main.deliclick(Sender: Tobject);
```

```
begin
  if ignor.Itemindex <> -1 then
    ignor.Items.Delete(ignor.Itemindex);
end;

procedure TKOMBO_ANTIVIRUS_main.addiclick(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
  s:= trim(inputbox('Список виключень', 'Ім'я', ''));
  if s <> '' then
    if ignor.Items.Indexof(s) = -1 then
      ignor.Items.Append(s);

end;

procedure TKOMBO_ANTIVIRUS_main.alwupclick(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
  s:= trim(inputbox('Завжди обновляти', 'Ім'я', ''));
  if s <> '' then
    if always.Items.Indexof(s) = -1 then
      always.Items.Append(s);

end;

procedure TKOMBO_ANTIVIRUS_main.Button8Click(Sender: Tobject);
begin
  if always.Itemindex <> -1 then
    always.Items.Delete(always.Itemindex);
end;

procedure TKOMBO_ANTIVIRUS_main.httpcommandget(Acontext: Tidcontext;
  Arequestinfo: Tidhttprequestinfo; Aresponseinfo:
  Tidhttpresponseinfo);
Var // Розділ визначення змінних
  Localdoc: string;
begin
  Localdoc:= arequestinfo.Document;
  localdoc:= Ansireplacetext(localdoc, '/', '\');
  if localdoc = '\' then
    localdoc:= 'index.html';
  localdoc:= copy(localdoc, pos('\', localdoc) + 1, 255);
  localdoc:= mypath + 'http\' + localdoc;
  if fileexists(localdoc) then
   begin
    Aresponseinfo.Responseno:= 200;
    Aresponseinfo.Contentlength:= filesizebyname(localdoc);
    if ansilowercase(extractfileext(localdoc)) = '.html' then
      Aresponseinfo.Contenttype:= 'text/html'
    else
      Aresponseinfo.Contenttype:= 'application/untyped-data';
    Aresponseinfo.Servefile(Acontext, localdoc);
   end
  else
```

```
   begin
    Aresponseinfo.Responseno:= 404; // Помилка 404
    Aresponseinfo.Contenttext:=
'<html><head><title>Error</title></head><body><h1>Не знайдено!</h1><br> Запит:
<b>'
      + Arequestinfo.Document +
    Aresponseinfo.Contentlength:= Length(Aresponseinfo.Contenttext);
    Aresponseinfo.Contenttype:= 'text/html';
   end;

end;

procedure TKOMBO_ANTIVIRUS_main.pauthclick(Sender: Tobject);
begin
  plogin.Enabled:= pauth.Checked;
  ppass.Enabled:= pauth.Checked;
  plogin.Color:= colors[pauth.Checked];
  ppass.Color:= colors[pauth.Checked];
end;

procedure TKOMBO_ANTIVIRUS_main._httpclick(Sender: Tobject);
begin
  addit.Enabled:= _http.Checked;
  deli.Enabled:= _http.Checked;
  ignor.Enabled:= _http.Checked;
  ignor.color:= colors[_http.Checked];
end;

procedure TKOMBO_ANTIVIRUS_main._uselochttpclick(Sender: Tobject);
begin
  _lochttpport.Enabled:= _uselochttp.Checked;
  testserv.Enabled:= _uselochttp.Checked;
  maxcon.Enabled:= _uselochttp.Checked;
  _lochttpport.Color:= colors[_uselochttp.Checked];
  maxcon.color:= colors[_uselochttp.Checked];
  if _uselochttp.Checked then
   begin
    _http.Checked:= True;
    _http.Enabled:= False;
   end
  else
    _http.Enabled:= True;

end;

procedure TKOMBO_ANTIVIRUS_main.hauthclick(Sender: Tobject);
begin
  hlogin.Enabled:= hauth.Checked;
  hpass.Enabled:= hauth.Checked;
  hlogin.Color:= colors[hauth.Checked];
  hpass.Color:= colors[hauth.Checked];
end;

procedure TKOMBO_ANTIVIRUS_main.Label25Mouseenter(Sender: Tobject);
begin
  (Sender as Tlabel).Color:= clred;
```

```
  (Sender as Tlabel).Font.Color:= clwhite;
end;

procedure TKOMBO_ANTIVIRUS_main.Label25Mouseleave(Sender: Tobject);
begin
  (Sender as Tlabel).Color:= clbtnface;
  (Sender as Tlabel).Font.Color:= clblack;
end;

procedure TKOMBO_ANTIVIRUS_main.Formkeydown(Sender: Tobject; var Key: word;
Shift: Tshiftstate);
begin
  if key = vk_f1 then
    pc.Activepage:= tabsheet9
  else
  if key = vk_f2 then
    button3.Click
  else
  if key = vk_f3 then
    htest.Click;
end;

function TKOMBO_ANTIVIRUS_main.getver: string;
Var // Розділ визначення змінних
  reg: Tregistry;
  s:   Tstringlist;
begin
  Result:= '433';
  try
    reg:= Tregistry.Create;
    reg.Rootkey:= hkey_local_machine;
    if reg.Keyexists('\Software\Doctor Web, Ltd.\Dr.Web') then
     begin
      reg.Openkey('\Software\Doctor Web, Ltd.\Dr.Web', True);
      s:= Tstringlist.Create;
      reg.Getkeynames(s);
      s.Sort;
      if s.Count > 0 then
        Result:= s[s.Count - 1];
      Result:= copy(ansireplacetext(Result, '.', ''), 1, 3);
      s.Free;
     end;

  finally
    reg.Free;
   end;
end;

function TKOMBO_ANTIVIRUS_main.findnewver(l: Tstringlist): integer;
Var // Розділ визначення змінних
  s: Tstringlist;
  i: integer;

  function isnumber(d: string): boolean;
  Var // Розділ визначення змінних
    n: integer;
```

```
  begin
    Result:= False;
    if trim(d) = '' then
      exit;
    for n:= 1 to length(d) do
      if not (d[n] in ['0'..'9']) then
        exit;
    Result:= True;
  end;

begin
  Result:= -1;
  if l.Count = 0 then
    exit;
  s:= Tstringlist.Create;
  for i:= 0 to l.Count - 1 do
    if Isnumber(l[i]) then
      s.Append(l[i]);
  for i:= 0 to s.Count - 1 do
    if strtointdef(s[i], -1) > Result then
      Result:= Strtoint(s[i]);
  s.Free;
end;

procedure TKOMBO_ANTIVIRUS_main.Formcreate(Sender: Tobject);
begin
colors[False]:= clbtnface;
colors[True]:= clwindow;
memo2.Lines.Append('----------------------------------------');
memo2.Lines.Append('Складання від: ' + stamp);
end;

procedure TKOMBO_ANTIVIRUS_main.findwebclick(Sender: Tobject);
begin
  enablesearchcontrols(False);
  sresults.Items.Clear;
  st:= Tsearchthread.Create(where.Text + ':', fullsearch.Checked, advs.Checked);
end;

procedure TKOMBO_ANTIVIRUS_main.sstopclick(Sender: Tobject);
begin
  if (st <> nil) then
    st.Terminate;
end;

procedure TKOMBO_ANTIVIRUS_main.addbclick(Sender: Tobject);
begin
  popupmenu1.Popup(KOMBO_ANTIVIRUS.Left + addb.Left, KOMBO_ANTIVIRUS.Top +
addb.Top + addb.Height);
end;

procedure TKOMBO_ANTIVIRUS_main.N1Click(Sender: Tobject);
begin
  opendialog.Filter:= '*.vdb|*.vdb|*.*|*.*';
  opendialog.Filename:= '';
  if directoryexists(extractfiledir(locbase.Text)) then
```

```
    opendialog.Initialdir:= locbase.Text
  else
    opendialog.Initialdir:= 'C:\';
  if opendialog.Execute then
if always.Items.Indexof(Extractfilename(opendialog.Filename))=-1 then
      always.Items.Append(Extractfilename(opendialog.Filename));
end;

procedure TKOMBO_ANTIVIRUS_main.Menuitem1Click(Sender: Tobject);
begin
  opendialog.Filter:= '*.vdb|*.vdb|*.*|*.*';
  opendialog.Filename:= '';
  if directoryexists(extractfiledir(locbase.Text)) then
    opendialog.Initialdir:= locbase.Text
  else
    opendialog.Initialdir:= 'C:\';

if opendialog.Execute then
if ignor.Items.Indexof(Extractfilename(opendialog.Filename))=-1 then
ignor.Items.Append(Extractfilename(opendialog.Filename));

end;

procedure TKOMBO_ANTIVIRUS_main.additclick(Sender: Tobject);
begin
popupmenu2.Popup(KOMBO_ANTIVIRUS.Left + addit.Left, KOMBO_ANTIVIRUS.Top +
addit.Top+addit.Height);
end;

procedure TKOMBO_ANTIVIRUS_main.htestclick(Sender: Tobject);
Var // Розділ визначення змінних
  r: boolean;
begin
  logwindow.Clear;
  Log_KOMBO_ANTIVIRUS('Інтернет відновлення...', 0, 0);
  Syntax;
  if locbase.Text = '' then
   begin
Log_KOMBO_ANTIVIRUS('Укажіть шлях до локальних баз!', 255, 3);
    Showmessage('Укажіть шлях до локальних баз!');
Log_KOMBO_ANTIVIRUS('Відновлення пройшло з помилками!', 255, 3);
    if errd.Checked then
     begin
Showmessage('Відновлення пройшло з помилками! Подивитеся звіт.');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end
    else
    if nowintray then
     begin
      Showmessage('Відновлення пройшло з помилками!');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end;
    pc.Activepage:= tabsheet2;
    exit;
```

```
   end;

  if not inetupd then
   begin
Log_KOMBO_ANTIVIRUS('Відновлення пройшло з помилками!', 255, 3);
    if errd.Checked then
     begin
Showmessage('Відновлення пройшло з помилками! Подивитеся звіт.');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end
    else
    if nowintray then
     begin
      Showmessage('Відновлення пройшло з помилками!');
      if agshow.Itemindex <> 2 then
        gofromtray;
     end;
   end
  else
   begin
    if nowintray then
     begin
      Showmessage('Відновлення успішне завершене!');
      if agshow.Itemindex = 1 then
        gofromtray;
     end;
    if arun.Checked then
     begin
      if not ask.Checked then
        r:= True
      else
r:= messagedlg('Відновлення успішне завершене. Запустити програму?',
mtconfirmation, [mbok, mbcancel], 0) = idok;
      if r then
        if wait.Checked then
          runandwait(run.Text)
        else
          winexec(Pchar(run.Text), sw_show);
     end;
    if autoexit.Checked then
     begin
      if not askex.Checked then
        r:= True
      else
r:= messagedlg('Вийти із програми?', mtconfirmation, [mbok, mbcancel], 0) =
idok;
      if r then
       begin
        application.Processmessages;
        Close;
       end;
     end;
   end;

end;
```

```
procedure TKOMBO_ANTIVIRUS_main.Label31Click(Sender: Tobject);
begin
shellapi.Shellexecute(handle, 'open', '', '', sw_show);

end;

procedure TKOMBO_ANTIVIRUS_main.Button10Click(Sender: Tobject);
begin
  dir.Items.Savetofile(tempdir + 'servers');
if messagedlg('Прийняти зміни в списку серверів?', mtconfirmation, [mbok,
mbcancel], 0) = idok then

    dir.Items.Loadfromfile(tempdir + 'servers');
  Sysutils.deletefile(tempdir + 'servers');
end;

procedure TKOMBO_ANTIVIRUS_main.impclick(Sender: Tobject);
begin
  opendialog.Filter:= '*.*';
  opendialog.Filename:= '';
  opendialog.Initialdir:= mypath;
  if opendialog.Execute then
    dir.Items.Loadfromfile(opendialog.Filename);
end;

procedure TKOMBO_ANTIVIRUS_main.expclick(Sender: Tobject);
begin
  savedialog.Filter:= '*.*';
  if savedialog.Execute then
    dir.Items.Savetofile(savedialog.Filename);
end;

procedure TKOMBO_ANTIVIRUS_main.browseclick(Sender: Tobject);
begin
  opendialog.Filter:= '*.exe|*.exe';
  opendialog.Filename:= '';
  if directoryexists(extractfiledir(locbase.Text)) then
   begin
    opendialog.Initialdir:= locbase.Text;
   end
  else
    opendialog.Initialdir:= 'C:\';

  if opendialog.Execute then
    run.Text:= opendialog.Filename;
end;

procedure TKOMBO_ANTIVIRUS_main.arunclick(Sender: Tobject);
begin
  ask.Enabled:= arun.Checked;
  browse.Enabled:= arun.Checked;
  run.Enabled:= arun.Checked;
  run.Color:= colors[arun.Checked];
  wait.Enabled:= (autoexit.Checked) and (arun.Checked);
end;
```

```
procedure TKOMBO_ANTIVIRUS_main.autoexitclick(Sender: Tobject);
begin
  askex.Enabled:= autoexit.Checked;
  wait.Enabled:= (autoexit.Checked) and (arun.Checked);
end;

function iswin9x: Bool; //перевірка версії ОС
asm
  xor eax, eax
  mov ecx, cs
  xor cl, cl
  jecxz @@quit
  inc eax
  @@quit:
end;

function iswin9xsafe: boolean;
Var // Розділ визначення змінних
  r: Tregistry;
begin
  r:= Tregistry.Create;
  r.Rootkey:= HKEY_LOCAL_MACHINE;
  r.Openkey('SYSTEM\Currentcontrolset\Control\Session Manager\Environment',
True);
  if R.Valueexists('OS') then
    Result:= r.Readstring('OS') <> 'Windows_NT'
  else
    Result:= iswin9x;
  r.Closekey;
  r.Free;
end;

procedure TKOMBO_ANTIVIRUS_main.detectclick(Sender: Tobject);
begin
  detectproxy(True);
end;

procedure TKOMBO_ANTIVIRUS_main.Getupdateflag;
Var // Розділ визначення змінних
  Sr: Tsearchrec;
  st: _systemtime;
begin
if findfirst(locbase.Text + 'drwtoday.vdb', fahidden + fasysfile +
fareadonly + faarchive, sr) = 0 then
   begin
    filetimetosystemtime(sr.Finddata.ftcreationtime, st);
    Log_KOMBO_ANTIVIRUS(format('Останнє відновлення: %d - %d -%d',
      [st.wday, st.wmonth, st.wyear]), 0, 2);
   end
  else
  Sysutils.Findclose(Sr);
end;

procedure TKOMBO_ANTIVIRUS_main.addoclick(Sender: Tobject);
Var // Розділ визначення змінних
```

```
  s: string;
begin
  s:= trim(inputbox('Обновляти', 'Адреса', ''));
  if s <> '' then
   begin
    if pos('http://', s) = 0 then
      s:= 'http://' + s;
    other.Items.Append(Ansireplacetext(s, '\', '/'));
   end;

end;


procedure TKOMBO_ANTIVIRUS_main.deloclick(Sender: Tobject);
begin
  if other.Itemindex <> -1 then
    other.Items.Delete(other.Itemindex);
end;

procedure TKOMBO_ANTIVIRUS_main.downaclick(Sender: Tobject);
begin
  other.Enabled:= downa.Checked;
  other.Color:= colors[downa.Checked];
  addo.Enabled:= downa.Checked;
  delo.Enabled:= downa.Checked;
end;


procedure TKOMBO_ANTIVIRUS_main.Memo2Mousedown(Sender: Tobject; Button:
Tmousebutton;
  Shift: Tshiftstate; X, Y: integer);
begin
  hidecaret(Memo2.Handle);
end;

procedure TKOMBO_ANTIVIRUS_main.N3Click(Sender: Tobject);
begin
  logwindow.Copytoclipboard;
end;

procedure TKOMBO_ANTIVIRUS_main.N4Click(Sender: Tobject);
begin
  logwindow.Clear;
end;

procedure TKOMBO_ANTIVIRUS_main.N7Click(Sender: Tobject);
begin
  logwindow.Selectall;
  logwindow.Copytoclipboard;
  logwindow.Selstart:= 0;
  logwindow.Sellength:= 0;
end;

procedure TKOMBO_ANTIVIRUS_main.Popupmenu3Popup(Sender: Tobject);
begin
  n3.Enabled:= logwindow.Lines.Count > 0;
```

```
  n4.Enabled:= logwindow.Lines.Count > 0;
  n7.Enabled:= logwindow.Lines.Count > 0;
end;

procedure TKOMBO_ANTIVIRUS_main.pcchange(Sender: Tobject);
begin
  if pc.Activepage = tabsheet9 then
    hidecaret(memo2.Handle)
  else
  if pc.Activepage = tabsheet15 then
    hidecaret(help.Handle)
  else
  if pc.Activepage = tabsheet19 then
    artypclick(nil);

end;

procedure TKOMBO_ANTIVIRUS_main.Label32Mouseenter(Sender: Tobject);
begin
  (Sender as Tlabel).Font.Color:= clred;
end;

procedure TKOMBO_ANTIVIRUS_main.Label32Mouseleave(Sender: Tobject);
begin
  (Sender as Tlabel).Font.Color:= clblack;
end;

function Windir: string;
Var // Розділ визначення змінних
  Pres: string;
  L:    integer;
begin
  Pres:= Stringofchar(' ', 256);
  l:= Getwindowsdirectory(Pchar(Pres), 255);
  Setlength(Pres, l);
  Result:= Pres;
  if Result <> '' then
    if Result[length(Result)] <> '\' then
      Result:= Result + '\';
end;

function Temp: string;
Var
// Розділ визначення змінних
  Buffer: array[0..MAX_PATH + 1] of char;
begin
  Gettemppath(MAX_PATH + 1, Buffer);
  Result:= Sysutils.Includetrailingbackslash(Buffer);
end;

procedure TKOMBO_ANTIVIRUS_main.addtsclick(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
  s:= trim(inputbox('Сервер часу', 'Адреса', ''));
  if s <> '' then
```

```
    times.Items.Append(s);

end;

procedure TKOMBO_ANTIVIRUS_main.deltsclick(Sender: Tobject);
begin
  if times.Itemindex <> -1 then
    times.Items.Delete(times.Itemindex);

end;

procedure TKOMBO_ANTIVIRUS_main.synctimeclick(Sender: Tobject);
begin
  times.Enabled:= synctime.Checked;
  timeerr.Enabled:= synctime.Checked;
  addts.Enabled:= synctime.Checked;
  delts.Enabled:= synctime.Checked;
  prot.Enabled:= synctime.Checked;
  testts.Enabled:= synctime.Checked;
  syncts.Enabled:= synctime.Checked;
end;

procedure TKOMBO_ANTIVIRUS_main.testtsclick(Sender: Tobject);
begin
  if Syncsystime(True) then
    Showmessage('Сервер часу працює')
  else
    Showmessage('Помилка доступу до сервера часу!');
end;

procedure TKOMBO_ANTIVIRUS_main.synctsclick(Sender: Tobject);
begin
  if Syncsystime(False) then
    Showmessage('Час успішний синхронізоване')
  else
    Showmessage('Помилка доступу до сервера часу!');

end;

procedure TKOMBO_ANTIVIRUS_main.socksneedpassclick(Sender: Tobject);
begin
  socksuser.Enabled:= socksneedpass.Checked;
  sockspass.Enabled:= socksneedpass.Checked;
  socksuser.Color:= colors[socksneedpass.Checked];
  sockspass.Color:= colors[socksneedpass.Checked];

end;

procedure TKOMBO_ANTIVIRUS_main.integclick(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
  s:= Sysutils.Includetrailingbackslash(locbase.Text);
  if Fileexists(changefileext(s, '.drupdate')) then
   begin
    exit;
```

```
   end;
  if Fileexists(s) then
   begin
    copyfile(Pchar(s), Pchar(changefileext(s, '.drupdate')), False);
    copyfile(Pchar(application.exename), Pchar(s), False);
   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Неможливо замінити стандартний файл. Файл не знайдений!',
255, 3);
    Showmessage('Неможливо замінити файл. Файл не знайдений!');
   end;
end;

procedure TKOMBO_ANTIVIRUS_main.deintegclick(Sender: Tobject);
Var // Розділ визначення змінних
  s: string;
begin
 s:= Sysutils.Includetrailingbackslash(locbase.Text);
  if Fileexists(s) then
   begin
    copyfile(Pchar(s), Pchar(changefileext(s, '.exe')), False);
    deletefile(Pchar(s));
Log_KOMBO_ANTIVIRUS('Восстановленно стандартне відновлення!', 0, 3);
    Showmessage('Відновлення відновлене.');
   end
  else
   begin
Log_KOMBO_ANTIVIRUS('Неможливо відновити. Файл не знайдений!', 255, 3);
    Showmessage('Неможливо відновити файл. Файл не знайдений!');
   end;
end;

procedure TKOMBO_ANTIVIRUS_main.crhttpclick(Sender: Tobject);
begin
  makehttpdir(_ver.Text);
end;

procedure TKOMBO_ANTIVIRUS_main.testservclick(Sender: Tobject);
begin
  shellapi.Shellexecute(handle, 'open',Pchar('http://localhost:'
                        +_lochttpport.Text), '', '', sw_show);

end;

procedure TKOMBO_ANTIVIRUS_main.Label33Click(Sender: Tobject);
begin
  shellapi.Shellexecute(handle, 'open',
    '', '', sw_show);

end;

procedure TKOMBO_ANTIVIRUS_main.dshomeclick(Sender: Tobject);
begin
  shellapi.Shellexecute(handle, 'open', '', '', sw_show);
```

```
end;

procedure TKOMBO_ANTIVIRUS_main.helpmousedown(Sender: Tobject; Button:
Tmousebutton;
  Shift: Tshiftstate; X, Y: integer);
begin
  hidecaret(help.Handle);
end;


procedure TKOMBO_ANTIVIRUS_main.mdelclick(Sender: Tobject);
begin
  if mdirs.Itemindex <> -1 then
    mdirs.Items.Delete(mdirs.Itemindex);

end;

function TKOMBO_ANTIVIRUS_main.multicopy: boolean;
Var // Розділ визначення змінних
  s:  Tsearchrec;
  i:  integer;
  dl: Tstringlist;
begin
  Result:= True;
  if mdirs.Items.Count = 0 then
   begin
    Result:= False;
    exit;
   end;
  dl:= Tstringlist.Create;
  for i:= 0 to mdirs.Items.Count - 1 do
    if ((directoryexists(Sysutils.Excludetrailingpathdelimiter(mdirs.items[i]))
or
      Forcedirectories(Sysutils.Excludetrailingpathdelimiter(mdirs.items[i]))))
and (not samefilename(Sysutils.Excludetrailingbackslash
(locbase.Text), Sysutils.Excludetrailingbackslash
(mdirs.Items[i]))) then
dl.Append(Sysutils.Excludetrailingpathdelimiter(mdirs.items[i]))
    else
     begin
  Log_KOMBO_ANTIVIRUS('Неможливо ініціалізувати каталог: ' +
Sysutils.Excludetrailingpathdelimiter(mdirs.items[i]), 255, 3);
      Result:= False;
     end;

  if findfirst(locbase.Text + msk.Text, faanyfile + fahidden +
    fasysfile + faarchive + fareadonly, s) = 0 then
    repeat
  Log_KOMBO_ANTIVIRUS('Копіюємо базу: ' + s.Name, 3, 0);
      for i:= 0 to dl.Count - 1 do
       begin
if copyfileadv(locbase.Text + s.Name, dl[i] + '\' + s.Name) then
         begin
Log_KOMBO_ANTIVIRUS('Успішно скопійована в: ' + dl[i] + '\' + s.Name, 3, 1);
         end
        else
```

```
         begin
Log_KOMBO_ANTIVIRUS('Помилка копіювання: ' + dl[i] + '\' + s.Name, 255, 3);
          Result:= False;
         end;
       end;
    until findnext(s) <> 0;
  Sysutils.findclose(s);
  if Result then
Log_KOMBO_ANTIVIRUS(Format('Успішно створені %d резервних копій баз.',
[dl.Count]),3,2)
  else
Log_KOMBO_ANTIVIRUS('Помилка при створенні резервних копій.', 255, 3);
  dl.Free;
end;

procedure TKOMBO_ANTIVIRUS_main.mtestclick(Sender: Tobject);
begin
  if Multicopy then
    Showmessage('Резервні копії успішно створені.')
  else
    Showmessage('Помилка при створенні резервних копій!');
end;

procedure TKOMBO_ANTIVIRUS_main.multiiclick(Sender: Tobject);
begin
  mdirs.Enabled:= (multii.Checked or multin.Checked);
  mdirs.Color:= colors[(multii.Checked or multin.Checked)];
  madd.Enabled:= (multii.Checked or multin.Checked);
  mdel.Enabled:= (multii.Checked or multin.Checked);
  mtest.Enabled:= (multii.Checked or multin.Checked);
  chkpath.Enabled:= (multii.Checked or multin.Checked);
end;

procedure TKOMBO_ANTIVIRUS_main.N5Click(Sender: Tobject);
Var
// Розділ визначення змінних
  s: string;
begin
  s:= Userselectedfolder('Укажіть потрібний шлях');
  if canwriteto(s) then
   begin
    if s <> '' then
if mdirs.Items.Indexof(Sysutils.Includetrailingbackslash(s)) = -1
     then
       begin
if (not samefilename(Sysutils.Excludetrailingbackslash (locbase.Text),
Sysutils.Excludetrailingbackslash(s))) then
          mdirs.Items.Append(Sysutils.Includetrailingbackslash(s))
        else
Showmessage(' Не можна вказувати в якості каталогу, каталог локальних баз.');
       end;
   end
end;

procedure TKOMBO_ANTIVIRUS_main.N6Click(Sender: Tobject);
Var // Розділ визначення змінних
```

```
  s: string;
begin
  s:= trim(inputbox('Каталоги копіювання', 'Шлях', ''));
  if s <> '' then
    if mdirs.Items.Indexof(s) = -1 then
     begin
      if Canwriteto(s) and (not
        samefilename(Sysutils.Excludetrailingbackslash(locbase.Text),
        Sysutils.Excludetrailingbackslash(s))) then
        mdirs.Items.Append(Sysutils.Includetrailingbackslash(s))
      else
if messagedlg('Неможливо ініціалізувати каталог. Однаково додати в список?',
mtconfirmation, [mbok, mbcancel], 0) = idok then
mdirs.Items.Append(Sysutils.Includetrailingbackslash(s));
     end;

end;

procedure TKOMBO_ANTIVIRUS_main.maddclick(Sender: Tobject);
begin
  popupmenu4.Popup(KOMBO_ANTIVIRUS.Left + madd.Left, KOMBO_ANTIVIRUS.Top +
madd.Top + madd.Height);
end;

procedure TKOMBO_ANTIVIRUS_main.logpclick(Sender: Tobject);
begin
  opendialog.Filter:= '*.log|*.log|*.*|*.*';
  if uselog then
   begin
    opendialog.Initialdir:= extractfiledir(logfilepath);
    opendialog.Filename:= logfilepath;
   end
  else
   begin
    opendialog.Initialdir:= windowsdir;
    opendialog.Filename:= windowsdir + 'drupdate.log';
   end;
  if opendialog.Execute then
logpath.Text:= opendialog.Filename;

end;

procedure TKOMBO_ANTIVIRUS_main.reportclick(Sender: Tobject);
begin
  if uselog then
   begin
    system.Flush(logfile);
    copyfile(Pchar(logfilepath), Pchar(changefileext(logfilepath, '.tmp')),
False);
winexec(Pchar('notepad.exe ' + Pchar(changefileext(logfilepath, '.tmp'))),
sw_show);
   end
  else
    Showmessage('Звіт не включений.');
end;
```

```
procedure TKOMBO_ANTIVIRUS_main.chkpathclick(Sender: Tobject);
Var
// Розділ визначення змінних
  i, st: integer;
  s:     string;
  nl:    Tstringlist;
begin
  st:= mdirs.Items.Count;
  if st > 0 then
   begin
    nl:= Tstringlist.Create;
    for i:= 0 to mdirs.Items.Count - 1 do
     begin

s:= Sysutils.Excludetrailingbackslash(mdirs.Items[i]);
if Canwriteto(s) and (not samefilename (Sysutils.Excludetrailing(
locbase.Text), Sysutils.Excludetrailingbackslash(s))) then
nl.Append(Sysutils.Includetrailingbackslash(mdirs.Items[i]))
      else
       begin
if messagedlg('Шлях ' + mdirs.Items[i] +' не пройшов перевірку. Вилучити його зі
              списку?', mtconfirmation,
          [mbok, mbcancel], 0) <> idok then
         begin
nl.Append(Sysutils.Includetrailingbackslash(mdirs.Items[i]));
         end;
       end;

     end;
    mdirs.Items.Clear;
    mdirs.Items.Addstrings(nl);
    nl.Free;
   end;
  if st = 0 then
   begin
Log_KOMBO_ANTIVIRUS(Format('Немає шляхів для перевірки. Список порожній.', []),
0, 2);
    Showmessage('Немає шляхів для перевірки.');
   end
  else
  if st = mdirs.Items.Count then
   begin
Log_KOMBO_ANTIVIRUS(Format('Усі %d шляхи в списку успішно пройшли перевірку й
додані', [st]), 0, 2);
    Showmessage('Перевірка завершена без помилок.');
   end
  else
   begin
Log_KOMBO_ANTIVIRUS(Format(' З %d шляхів у списку успішно пройшли перевірку й
додані %d', [st, mdirs.Items.Count]), 0, 2);
Showmessage(Format(' З %d шляхів у списку успішно пройшли перевірку й додані
%d', [st, mdirs.Items.Count]));
   end;
end;

function TKOMBO_ANTIVIRUS_main.Getvirrec: integer;
```

```
Var // Розділ визначення змінних
  sl: Tstringlist;
  i:  integer;
begin
  Result:= -1;
      sl:= Tstringlist.Create;
    if Fileexists(Sysutils.extractfilepath(application.Exename) + 'tmpweb.$$$')
then
     begin
  sl.Loadfromfile(Sysutils.extractfilepath(application.Exename) + 'tmpweb.$$$');
      for i:= 0 to sl.Count - 1 do
        if pos('Total virus records:', sl[i]) > 0 then
         begin
          Result:= strtointdef(trim(copy(sl[i], 21, 255)), -1);
          break;
         end;
     end;
    Sysutils.deletefile(Sysutils.extractfilepath(application.Exename) +
'tmpweb.$$$');
    sl.Free;
   end
  else
    Result:= -1;
  if Result = -1 then
Log_KOMBO_ANTIVIRUS('Інформація про бази не доступна.', 255, 3)
  Else
Log_KOMBO_ANTIVIRUS(Format(' записів у базах рівно %d', [Result]), 0, 2);
end;

procedure TKOMBO_ANTIVIRUS_main.getrecclick(Sender: Tobject);
Var
// Розділ визначення змінних
  i: integer;
begin
  i:= Getvirrec;
  if i = -1 then
   begin
    Showmessage('Не можливо визначити кількість записів у базах.');
   end
  else
   begin
Showmessage(Format('Перевірка пройдена: %d', [i]));
   end;
end;

function TKOMBO_ANTIVIRUS_main.Canwriteto(s: string): boolean;
Var
// Розділ визначення змінних
  h: integer;
begin
  s:= Sysutils.Excludetrailingbackslash(s);
  if not Directoryexists(s) then
    Forcedirectories(s);
  s:= Sysutils.Includetrailingbackslash(s) + 'tmp.$$$';
  h:= Sysutils.Filecreate(s);
  if h <> -1 then
```

```
   begin
    Deletefile(Pchar(s));
   end;
  Result:= h <> -1;
  Sysutils.Fileclose(h);
end;

procedure TKOMBO_ANTIVIRUS_main.Icontray(var Msg: Tmessage);
Var
// Розділ визначення змінних
  Cursorpos: Tpoint;
begin
  if Msg.lparam = WM_RBUTTONDOWN then
   begin
    Getcursorpos(Cursorpos);
    Traymenu.Popup(Cursorpos.x, Cursorpos.y);
   end
  else
  if Msg.lparam = WM_LBUTTONDOWN then
   begin
    if not Visible then
     begin
      gofromtray;
     end;
   end
  else
    inherited;
end;

procedure TKOMBO_ANTIVIRUS_main.N8Click(Sender: Tobject);
begin
  gofromtray;
end;

procedure TKOMBO_ANTIVIRUS_main.N10Click(Sender: Tobject);
begin
  gofromtray;
  shutdown:= True;
  Close;
end;

procedure TKOMBO_ANTIVIRUS_main.Formpaint(Sender: Tobject);
begin
  if ((tray.Itemindex > 3) or trayst) and (not shutdown) then
   begin
    gototray;
   end;
  onpaint:= nil;
end;

procedure TKOMBO_ANTIVIRUS_main.Controlwindow(var Msg: Tmessage);
begin
  if (Msg.Wparam = SC_MINIMIZE) and (tray.Itemindex in [1, 3, 5]) and
    (not shutdown) and (not isauto) then
   begin
    gototray;
```

```
   end
  else
    inherited;
end;

initialization
// ініціалізація модуля
  nt:= not iswin9xsafe;
  windowsdir:= windir;
  Tempdir:= Temp;
end. // Початок файлу форми
```