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 оцінки стійкості комп’ютерних мереж до загроз”

Виконав здобувач вищої освіти
IV курсу, групи КБ-21-3СК
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Бойко О.А.
«____» _____ 2024 р.

Керівник проекту
доктор технічних наук, професор
_____ Коваленко О.В.
«____» _____ 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

Бойко Олександру Андрійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз*

2. Керівник роботи *Коваленко Олександр Володимирович, докт. техн. наук, професор*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 136-02 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 оцінки стійкості комп’ютерних мереж до загроз*

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
_____ Коваленко О.В.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2024 р.

Підпис здобувача
_____ Бойко О.А.
(прізвище та ініціали)

АНОТАЦІЯ

Бойко О.А. Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних мереж до загроз. 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 оцінки стійкості комп'ютерних мереж до загроз.

Метою розробки є програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних мереж до загроз.

Результат роботи – програмна реалізація системи кібербезпеки оцінки стійкості комп'ютерних мереж до загроз.

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, оцінки стійкості

ABSTRACT

Boiko O.A. Cyber security system software for assessing the resistance of computer networks to threats. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this final qualification work for the first (bachelor) level of higher education, software was developed, which is intended for the cyber security system for assessing the resistance of computer networks to threats.

The purpose of the development is the software of the cyber security system for assessing the resistance of computer networks to threats.

The result of the work is the software implementation of the cyber security system for assessing the resistance of computer networks to threats.

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, resilience assessments

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ІТ	–	інформаційні технології
ПЗ	–	програмне забезпечення
ТСКМ	–	телекомунікаційні системи і комп'ютерні мережі
QoS	–	механізми контролю й керування якістю

ВСТУП

Актуальність теми. Політика інформаційної безпеки для оцінки стійкості комп'ютерних мереж до загроз – це формальний, задокументований набір правил і вказівок, які організація встановлює для захисту своїх інформаційних активів і забезпечення конфіденційності, цілісності та доступності своїх даних у комп'ютерній мережі.

Ця політика служить основою для управління ризиками, визначення прийнятної поведінки та встановлення очікувань безпеки для працівників, підрядників, партнерів та інших зацікавлених сторін. Це також допомагає організаціям дотримуватися правових, нормативних і галузевих вимог.

Ефективна політика інформаційної безпеки зазвичай включає такі компоненти:

– Мета: чітке формулювання цілей політики та зобов'язань організації щодо інформаційної безпеки.

– Сфера застосування: опис систем, даних і персоналу, на які поширюється політика, включаючи будь-яких сторонніх постачальників або партнерів.

– Ролі та обов'язки: визначення ролей та обов'язків різних зацікавлених сторін, таких як керівництво, IT-персонал і співробітники, у впровадженні, підтримці та забезпеченні політики.

– Управління активами: вказівки щодо ідентифікації, класифікації та управління інформаційними активами організації для забезпечення відповідних рівнів захисту.

– Контроль доступу: правила надання та скасування доступу до систем і даних, включаючи автентифікацію користувачів, авторизацію та керування паролями.

– Реагування на інциденти: процедури виявлення, звітування та реагування на інциденти безпеки, включаючи протоколи зв’язку та шляхи ескалації.

– Фізична безпека: заходи для захисту об’єктів, обладнання та інформаційних активів організації від несанкціонованого доступу, крадіжки або пошкодження.

– Навчання та підвищення обізнаності: Вимоги до регулярних програм навчання та підвищення обізнаності працівників, щоб сприяти розвитку культури безпеки та гарантувати, що персонал розуміє свої обов’язки.

– Моніторинг і аудит: процеси моніторингу відповідності політиці, включаючи регулярні аудити, оцінки та перегляди для виявлення прогалин і областей для покращення.

– Перегляд і оновлення політики: графік періодичного перегляду та оновлення політики, щоб переконатися, що вона залишається актуальною, ефективною та відповідає мінливим потребам організації та мінливому ландшафту загроз.

Політика інформаційної безпеки є критично важливим компонентом загальної стратегії безпеки організації, оскільки вона забезпечує основу для впровадження технічних заходів, адміністративного контролю та найкращих практик для захисту її інформаційних активів.

Аналіз мережі є важливим у сфері комп’ютерів, оскільки він дозволяє аналізувати дані в соціальних мережах і мережевих структурах, які переважають у різних наборах даних. Він також відіграє важливу роль у забезпеченні високої продуктивності та безпеки комп'ютерних систем і мереж. Аналіз мережі особливо цінний у цифровій криміналістиці, оскільки він може надати цінні дані для розслідування подій у мережі та аналізу шкідливих двійкових файлів і мережевого трафіку. Крім того, інтеграція та розвиток систем зв’язку та комп’ютерних мереж створили взаємодоповнюючі відносини, що робить аналіз мережі вирішальним для розуміння поточного стану та майбутнього розвитку

комп’ютерних мереж і систем зв’язку. Нарешті, в епоху великих даних аналіз безпеки мережевої інформації має важливе значення для розробки та використання комп’ютерних мереж, оскільки він допомагає ідентифікувати та зменшувати ризики безпеки.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз.

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

– Огляд існуючих систем оцінки стійкості комп’ютерних мереж до загроз.

– Дослідження системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз.

– Програмна реалізація системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі оцінки стійкості комп’ютерних мереж до загроз.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз, 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 призначена для реалізації програмного забезпечення оцінки стійкості комп'ютерних мереж. З появою комп'ютерних мереж (КМ) важливою проблемою стала проблема стійкого забезпечення сервісів, які надаються цими системами. Негативний антропогенний вплив може бути результатом відсутності необхідних знань та вмінь, результатом халатності або злого наміру. Враховуючи, що теоретичні результати забезпечення стійкості технічних систем від негативних фізичних впливів лише частково ефективні для протидії негативним антропогенним впливам, виникла необхідність в формуванні нового терміну для позначення стійких сервісів КМ.

Проблематика стійкості таких систем, маючи споріднену природу із проблематикою гарантоздатності, потребує врахування не лише аспектів стійкості та інформаційної безпеки, але й більш повного врахування когнітивних аспектів їх функціонування. Тому в вітчизняних наукових роботах запропоновано термін «інформаційна стійкість» [3,6,7] як здатність КМ або інформаційної технології досягати поставленої довгострокової мети. Врахування таких особливостей потребує певного переосмислення – розширення та доповнення – складових поняття гарантоздатності для можливості їх подальшого ефективного використання. Якщо в розрізі гарантоздатності аналізуються негативні впливи (фізичні, антропогенні) на КМ та інформаційні ресурси, то в розрізі інформаційної стійкості додатково аналізуються негативні явища. Під негативними явищами слід розуміти використання інформаційних технологій, яке саме по собі не впливає на стійке (гарантоздатне) функціонуванню КМ та мереж, але не відповідає законодавчим або моральним нормам, прийнятим в відповідних галузях людської діяльності або в суспільстві в цілому. Можна

вказати на наступні приклади негативних явищ у використанні інформаційних технологій, пов'язаних з антропогенним фактором – створення та розповсюдження в комп'ютерній мережі сцен насильства та порнографії, порушення прав інтелектуальної власності, плагіат в творчій діяльності та освіті, невідповідність структури електронних документів (креслень, бухгалтерської звітності, кваліфікаційних студентських робіт та багатьох інших) існуючим нормативним положенням і т.ін.

1.2 Область застосування

Областю застосування системи є оцінка стійкості комп'ютерних мереж. Із наведених визначень інформаційної стійкості випливає, що суттєві проблеми виникають навіть на рівні автоматизованого визначення наявності негативних антропогенних впливів та особливо негативних явищ, не кажучи про складність створення методів та засобів протидії таким впливам та явищам. Можливим напрямком вирішення таких проблем є дослідження методів та засобів автоматизації оцінювання когнітивних властивостей інформації.

Що таке управління ризиками інформаційної безпеки?

Управління ризиками інформаційної безпеки – це процес виявлення, оцінки, встановлення пріоритетів і пом'якшення ризиків, пов'язаних з інформаційними активами організації та IT-інфраструктурою. Метою управління ризиками інформаційної безпеки є захист конфіденційності, цілісності та доступності інформаційних активів при мінімізації впливу інцидентів безпеки на діяльність організації, репутацію та юридичні зобов'язання.

Процес зазвичай включає такі кроки:

1. Визначте активи: створіть перелік інформаційних активів, таких як обладнання, програмне забезпечення, дані та мережеві компоненти, і визначте їх цінність для організації.

2. Визначте загрози та вразливі місця: проаналізуйте потенційні загрози та вразливі місця, які можуть поставити під загрозу безпеку інформаційних активів. Загрози можуть надходити з різних джерел, як-от хакери, зловмисне програмне забезпечення, стихійні лиха або помилка людини. Уразливості стосуються слабких місць у системах, процесах або політиках, які можуть бути використані загрозами.

3. Оцінка ризиків: оцінка ймовірності та потенційного впливу виявлених загроз і вразливостей на інформаційні активи. Це можна зробити за допомогою якісних або кількісних методів, таких як матриці ризику, системи оцінки або ймовірнісні моделі.

4. Пріоритезація ризиків: ранжування ідентифікованих ризиків на основі їх потенційного впливу та ймовірності, зосереджуючись на найбільш значущих ризиках, які потребують негайної уваги та ресурсів.

5. Розробка стратегій пом'якшення: розробка та впровадження заходів безпеки, політики та процедур для пом'якшення пріоритетних ризиків. Вони можуть включати превентивні заходи (наприклад, брандмауери, шифрування), заходи виявлення (наприклад, системи виявлення вторгнень, моніторинг журналів) і коригувальні заходи (наприклад, плани реагування на інциденти, резервне копіювання).

6. Моніторинг і перегляд: постійно відстежуйте ефективність запроваджених засобів контролю безпеки та переглядайте процес управління ризиками, щоб переконатися, що він залишається актуальним і оновленим. Це означає бути в курсі нових загроз і вразливостей, а також про зміни в активах організації, її діяльності та правових вимогах.

7. Звітуйте та повідомляйте: регулярно повідомляйте зацікавленим сторонам, таким як вище керівництво, члени правління та аудитори, про стан ризиків інформаційної безпеки та заходи щодо їх зменшення. Чітка комунікація допомагає гарантувати, що кожен в організації розуміє свої ролі та обов'язки щодо підтримки інформаційної безпеки.

Передовий досвід інформаційної безпеки:

– Розробіть план реагування на інциденти. План реагування на інциденти готує організацію до ефективного управління інцидентами безпеки та реагування на них, мінімізуючи потенційний вплив і забезпечуючи швидке повернення до нормальної роботи. Встановлюючи чіткі ролі та обов'язки, описуючи процедури реагування та сприяючи постійному вдосконаленню, план реагування на інциденти допомагає організаціям підтримувати надійну безпеку та захищати свої важливі активи.

– Використовуйте DevSecOps. DevSecOps, що розшифровується як Development, Security, and Operations, інтегрує методи безпеки протягом життєвого циклу розробки програмного забезпечення. Включаючи безпеку як невід'ємну частину процесу розробки, DevSecOps прагне зменшити вразливість, забезпечити швидшу реакцію на інциденти безпеки та сприяти культурі спільної відповідальності за безпеку в усій організації.

– Створіть червону команду та синю команду. У вправах «Червона команда – синя команда» дві групи працюють разом, щоб посилити безпеку організації. Червона команда імітує атаки в реальному світі, тоді як синя команда захищає від цих атак, виявляє вторгнення та пом'якшує загрози. Беручи участь у цих навчаннях, організації можуть зміцнити свою безпеку, покращити можливості реагування на інциденти та виховувати культуру спільної відповідальності за безпеку.

– Проведіть тестування на проникнення. Тестування на проникнення передбачає імітацію реальних кібератак на системи, мережі або програми організації для виявлення вразливостей і оцінки захисту їх безпеки. Проводячи регулярні тести на проникнення, організації можуть виявити слабкі місця в своїх засобах безпеки, оцінити свою стійкість до атак і усунути проблеми, перш ніж вони будуть використані.

– Автоматизуйте керування вразливими місцями. Впровадження автоматизованих інструментів керування вразливістю, таких як сканери вразливостей і системи керування виправленнями, допомагає організаціям регулярно виявляти, оцінювати, визначати пріоритети та усувати вразливості

системи безпеки. Цей безперервний процес зменшує вікно можливостей для зловмисників використовувати відомі слабкі місця та покращує загальну безпеку організації.

– **Впровадити шифрування даних.** Шифрування даних захищає конфіденційні дані від несанкціонованого доступу та забезпечує конфіденційність і цілісність цих даних як під час передачі, так і під час зберігання. Застосовуючи надійні засоби шифрування, організації можуть мінімізувати ризики витоку даних, зміцнити довіру серед зацікавлених сторін і підтримувати надійну безпеку.

– **Використовуйте надійну автентифікацію.** Впровадження надійних механізмів автентифікації, таких як багатофакторна автентифікація (MFA), допомагає гарантувати, що лише авторизовані користувачі можуть отримати доступ до конфіденційних даних і систем. MFA поєднує кілька методів перевірки, наприклад те, що користувач знає (пароль), те, що користувач має (токен безпеки чи смартфон), або те, що користувач є (біометрія). Цей багаторівневий підхід значно знижує ризик неавторизованого доступу через скомпрометовані облікові дані.

– **Навчайте та навчайте користувачів.** Людська помилка часто є суттєвим фактором порушення інформаційної безпеки. Проведення регулярних тренінгів з питань безпеки для співробітників, підрядників і партнерів допомагає створити культуру безпеки та гарантує, що користувачі розуміють свої обов’язки щодо захисту даних організації. Теми такого навчання можуть включати знання про фішинг, безпечні введення пароля та те, як повідомляти про ймовірні інциденти безпеки.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз, 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

EMYCIN

Клас програм, які ми називаємо оболонкою експертної системи, створювався з метою дозволити непрограмістам скористатися результатами роботи програмістів, що вирішували аналогічні проблеми. Так, програма EMYCIN дозволяє використовувати архітектуру системи MYCIN у додатку до інших областей медицини (нагадаємо, що програма MYCIN була орієнтована тільки на захворювання крові). На базі EMYCIN були розроблені експертні системи як для медицини (наприклад, система PUFF для діагностики легеневих захворювань), так і для інших областей знань, наприклад програма структурного аналізу SACON.

Зовсім очевидно, що оболонки є програмами, орієнтованими на досить вузький клас завдань, хоча й більше широкий, чим та програма, на основі якої була створена та або інша оболонка. Автор системи EMYCIN Ван Мелле одним з перших підкреслив, що оболонки аж ніяк не є універсальною архітектурою для рішення проблем. Розроблена ним система EMYCIN орієнтована на ті проблеми діагностування з більшими обсягами даних, які піддаються рішенню за допомогою дедуктивного підходу в припущенні, що простір діагностичних категорій стаціонарно. Кленсі назвав клас подібних проблем "проблемами евристичної класифікації". Однак цей підхід значно менше підходить для рішення проблем конструювання, тобто об'єднання окремих елементів у єдиний комплекс із урахуванням заданих обмежень. На жаль, не можна занадто довіряти рекомендаціям про можливість використання оболонки для рішення конкретних

проблем. Справа в тому, що ми ще не маємо настільки чіткого подання про класифікацію завдань, розв'язуваних експертними системами, щоб можна було точно представити, до якого саме класу варто віднести конкретну систему. Класифікації завдань, придатних для рішення експертними системами, присвячено досить багато робіт. Може створитися враження, що відрізнити завдання класифікації від завдання конструювання можна й "неозброєним оком", але це враження неправдиве. Безліч проблем допускає рішення різними способами. Наприклад, у тому підході до завдання діагностування, що використаний у системі INTRN1ST, застосовуються методи, властиві рішенню завдань конструювання. Складні проблеми найчастіше вимагають застосування комбінованих методів, у яких проглядаються риси, властивим обом підходам.

Ми ще зупинимося на загальному підході до вибору інструментальних засобів для побудови конкретних експертних систем. Але якщо мова йде конкретно про оболонки, те вже зараз потрібно відзначити, що більшість комерційних продуктів цього типу підходить тільки для тих проблем, у яких простір пошуку невеликий. Як правило, у них застосовується метод вичерпного пошуку з побудовою зворотного ланцюжка виводу й обмежених можливостей керування процесом. Але деякі сучасні оболонки, наприклад M.4, як затверджують їхні творці, можуть застосовуватися для рішення широкого кола завдань, оскільки вони підтримують безліч функцій подання знань і керування, включаючи й моделювання прямого ланцюжка логічного виводу, процедури, передачу повідомлень і т.п.

Простота мов подання знань, застосовуваних у більшості оболонок, є, з одного боку, достоїнством, а з іншого боку недоліком такого роду систем. На це звернула увагу Ейкінс у критичному зауваженні із приводу реалізації експертної системи PUFF на базі оболонки EMYCIN.

Використаний в EMYCIN формалізм правил, що породжують, утрудняє поділ різних видів знань – евристичних, керуючих, знань про очікувані значення параметрів. Ми вже відзначали що здатність системи диференціювати види знань

розглядається багатьма дослідниками як одне з головних умов забезпечення її "прозорості" для користувача

Недостатня структурованість набору правил, що породжують, в EMYCIN також утрудняє й сприйняття нових знань, оскільки додавання в базу знань нового правила вимагає внесення змін у різні компоненти системи. Наприклад, потрібно вносити зміни в таблиці знань, що містять інформацію про медичні параметри. Це одна із проблем, рішенням якої пишаються творці системи TEIRESIAS.

Основним методом формування суджень в EMYCIN є побудова зворотного ланцюжка виводу. При цьому використовується безліч правил мета- і об'єктного рівня. У результаті дуже складно формувати вичерпне й зрозуміле для користувача пояснення. Як відзначав Кленсі ті рішення, які приймаються на етапі програмування правил, зокрема дотичного порядку й кількості виражень в антецедентної частини, можуть разючим образом вплинути на шлях пошуку в просторі рішень у процесі функціонування системи.

Інше критичне зауваження Ейкінс стосується не стільки конкретної системи PUFF або EMYCIN, скільки функціональних можливостей систем, що базуються на правилах, загалом, а отже, і всіх оболонок, у яких правила, що породжують, використовуються як основну мову подання знань. Значна частина експертності – це знання про типові випадки, тобто досить що часто зустрічаються в предметній області. Експерти легко розпізнають відомі типові випадки й здатні без особливої праці класифікувати їх у термінах ідеальних прототипів навіть при наявності певних "перешкод" або неповних даних. Вони інтуїтивно розрізняють підходящому випадку або незвичайні значення вихідних даних і приймають адекватне рішення про те, як надійти надалі при рішенні проблеми.

Останнє зауваження із приводу використання оболонок стосується механізму обробки невизначеності. Такі оболонки, як M.1, уже містять у собі певний формальний механізм роботи з невизначеністю, наприклад заснований на

використанні коефіцієнтів упевненості. Однак більшість, якщо не всі використані в оболонках механізми такого роду, не погодяться з виводами теорії ймовірностей і мають властивості, які із працею піддаються аналізу. Звичайно, конкретному методу обробки невизначеності при рішенні конкретного завдання в даній предметній області можна дати прагматичне обґрунтування, як надійшов, наприклад, Шортлифф стосовно схеми обробки коефіцієнтів упевненості в системі MYCIN. Але навряд чи виправдано поширювати цей апарат на інші області застосування, вмонтувавши його в оболонку.

У порівнянні з першими розробками сучасні оболонки більше гнучкі, принаймні, у тому, що без особливої праці можуть бути інтегровані в більшість операційних середовищ, доступних на ринку програмного забезпечення, і оснащені досить розвиненими засобами користувальницького інтерфейсу.

G2

Для створення експертних систем реального часу використовуються спеціалізовані інтегровані середовища, подібні G2.

Класи завдань, для яких призначена G2:

– моніторинг у реальному масштабі часу;

– системи керування верхнього рівня;

– системи виявлення несправностей;

– діагностика;

– складання розкладів;

– планування;

– оптимізація;

– системи – порадники оператора;

– системи проектування.

На основі аналізу характеристики перерахованих вище інструментальних засобів нами буде реалізована програмна оболонка для проектування й реалізації експертних систем з урахуванням всіх достоїнств і недоліків, властивим розглянутим програмним засобам.

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 оцінки стійкості комп'ютерних мереж до загроз.

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

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

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

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

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

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

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

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

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

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Розглянемо існуючі методи підвищення вірогідності експертної оцінки основних технічних параметрів, режимів роботи й характеристик вхідних у поняття стійкості комп'ютерних мереж (КМ) до загроз на поточному етапі життєвого циклу, а також технічні засоби аналізу ризиків провідних виробників, як прототипи розроблювальних засобів підвищення вірогідності експертної оцінки стійкості, їхнього достоїнства й недоліки, а також аналізується проблематика використання в питаннях експертизи стійкості комп'ютерних мереж до загроз на поточному етапі життєвого циклу.

Розглядається стійкість КМ, що:

– поєднує в собі такі категорії, як стійкість, функціональність і інформаційна безпека;

– виробляється аналіз принципової схеми КМ із погляду побудови живучого програмно-апаратного комплексу;

– здійснюється вимір основних параметрів функціонування КМ за допомогою Microsoft Windows System Resource Manager;

– у складі окремого проекту визначаються значення стійкості КМ і зіставляються отримані значення із заданими в ТЗ;

– у складі окремого проекту виробляється побудова моделі загроз і порушників, виробляється аналіз ризиків і їхнє зіставлення з заданим у ТЗ рівнем інформаційної безпеки;

– виробляється розбивка життєвого циклу проекту на послідовність ітерацій у вигляді спіральної (spiral) моделі Боема, що являє собою систему координат, що рухається спіралі по, яка розвертається, що поєднує чотири фази, що повторюються на кожному новому витку часу; здійснюється реалізація моделі

на етапі впровадження в промислову експлуатацію шляхом розбивки етапу на п'ять стадій:

– стадія розробки концепції впровадження системи;

– стадія проекту впровадження КМ;

– стадія макетування;

– стадія першої версії робочої КМ;

– стадія готової КМ.

З переходом від стадії до стадії в експертної групи відбувається нагромадження інформації, досвіду й історичних даних; вірогідність експертизи починає зростати, але разом з її зростанням змінюється й стан стійкості комп'ютерних мереж до загроз. Взаємний вплив перерахованих факторів приводить до потреби корекції результатів експертизи.

Для забезпечення експертизи існує множину технічних засобів, таких, як:

– комплекс Digital Security Office 2006 російського виробництва;

– COBRA, розробки компанії C & A Systems Security Ltd.;

– RA Software Tool, засоби розробки компанії RiskWatch.

В основі розглянутих програмних комплексів лежать статистичні й імовірнісні методи, що вимагають великої кількості історичних даних про роботу КМ. В умовах неповноти інформації розглянуті засоби виявляються неефективними.

Узагальнюючи вищесказане, при розгляді завдань оцінки стійкості телекомунікаційних систем і комп'ютерних мереж (КМ) в умовах неповноти інформації з урахуванням поточного етапу життєвого циклу необхідно виділити фактори, що роблять на вірогідність експертної оцінки безпосередній вплив:

– неповнота історичних даних вимірів основних параметрів функціонування КМ на різних режимах роботи (видах трафиков);

– неповнота історичних даних про стійкість апаратних компонентів і функціональності програмних засобів КМ;

– ризики інформаційної безпеки КМ;

– особливості поточного етапу життєвого циклу і його стадій.

Для успішного рішення перерахованих вище проблем, представляється доцільним розгляд ряду питань, серед яких можна виділити наступні:

– розгляд існуючих методів оцінки стійкості комп’ютерних мереж до загроз;

– експлікація (роз'яснення) експертних оцінок стійкості комп’ютерних мереж до загроз в умовах неповноти історичних даних на поточному етапі життєвого циклу комп’ютерних мереж до загроз;

– розробка моделі життєвого циклу для умов неповноти інформації;

– розробка моделі забезпечення вірогідності експертизи стійкості комп’ютерних мереж до загроз на поточному етапі життєвого циклу (етапі впровадження в промислову експлуатацію) у вигляді знаходження погрішності експертних оцінок;

– імітаційне моделювання проведення експертизи стійкості комп’ютерних мереж до загроз на поточному етапі життєвого циклу за рахунок застосування моделі, створеної в середовищі, здатної забезпечити процеси збору й аналізу історичних даних, у тому числі про граничні режими роботи КМ.

Розробимо метод експлікації експертних оцінок стійкості комп’ютерних мереж до загроз, що реалізує можливість роз'яснення точки зору експертів на об'єкт дослідження й враховуючий розкид їхніх думок. У його основі лежить визначення комплексних показників стійкості телекомунікаційних систем і комп'ютерних мереж (КМ) на основі логіко-лінгвістичних методів оцінки, нечітких поліхроматичних множин, застосовуваних в умовах неповноти інформації, що робить вплив на вірогідність експертної оцінки стійкості комп’ютерних мереж до загроз на поточному етапі життєвого циклу.

Застосування для оперування з невизначеними величинами апарата теорії імовірності приводить до того, що фактично невизначеність, незалежно від її природи, ототожнюється з випадковістю, тим часом як основним джерелом невизначеності в багатьох процесах прийняття рішень є нечіткість (неточність), або розпливчастість (fuzzines).

У питаннях забезпечення стійкості комп’ютерних мереж до загроз виявляються неефективними й іншими кількісними методами прийняття рішень, такі, як максимізація очікуваної корисності, мінімаксна теорія, методи максимальної правдоподібності, теорія ігор, аналіз "витрати – ефективність" і інші. Розглянуті методи допомагають вибирати найкращі з множини можливих рішень лише в умовах одного конкретного виду невизначеності або в умовах повної визначеності.

У зв'язку з вищесказаним пропонується реалізувати метод експлікації експертних оцінок стійкості телекомунікаційних систем і комп'ютерних мереж (КМ) на основі методів нечіткої логіки.

На першому етапі на підставі IDEF0 моделі загроз і порушників здійснюється аналіз можливих загроз стійкості й аналізується їхній вплив один на одного.

На другому етапі на підставі кожної сформульованої загрози й внутрішніх факторів формуються вихідні дані, що служать компонентами вектора запиту.

Своєрідною бальною шкалою для формування єдиних думок експертів є таблиця визначення ризику залежно від трьох факторів, що складена на основі статистичних розробок компанії Gartner Group.

Такі таблиці використовуються як в "паперових", так і в програмних методиках оцінки ризиків. В останньому випадку матриця оцінки ризиків поставляється разом із системою оцінки КМ і не підлягає зміні.

На третьому етапі виконується ранжирування загроз (вихідних даних) через визначення коефіцієнтів важливості. Експерти вказують свої суб'єктивні оцінки у вигляді чисельних значень від 0 (максимально необхідна, оптимальна, стійкість – точка "обрію") до 1 (повний крах системи).

Результатом ранжирування нечіткої множини коефіцієнтів важливості на множини вихідних даних, може стати функція приналежності (ФП – відображення множини вихідних даних в одиничний відрізок [0, 1].

Для визначення базової (еталонної) функції приналежності (ФП) нечіткої множини коефіцієнтів важливості в бакалаврській роботі пропонується

використовувати метод побудови експонентної ФП (Гаусса – gaussmf, в інтерпретації). З аналізу різних джерел, присвячених методам побудови функцій приналежності, розглянута метод доцільніше всього використовувати для рішення завдань виробітку й оцінки альтернатив, а також подання нормально розподілених випадкових величин, таких, як загрози (збурювання) інформаційної безпеки. Відповідно, реакція експертів на розглянуті загрози, також являє собою нормально розподілену випадкову величину, що спрощує подальший перехід до стохастичних методів оцінки.

У бакалаврській роботі пропонується поняття колір нечіткої множини розглядати як експлікацію (роз'яснення, пояснення) понять:

– властивість;

– атрибут;

– характеристика.

У нечіткій поліхроматичній множини, що моделює складний об'єкт, який володіє різними властивостями, кожному елементу поставимо у відповідність множину персональних кольорів або розфарбувань, що роз'ясняють ці властивості, а множини в цілому – множину унітарних кольорів або розфарбувань самої множини.

На четвертому етапі пропонується, щоб уникнути некоректності нечітких моделей, в описі множини вказувати умови існування персональних кольорів елементів і унітарних кольорів самої множини, а також ураховувати їхній вплив один на одного.

На підставі аналізу значимості лінгвістичних термов мнемонічне подання бінарної матриці буде показувати наявність фізичного впливу загроз на стан стійкості комп’ютерних мереж до загроз у цілому.

Персональні розфарбування всіх елементів множини описуються матрицею бінарних відносин між елементами й квітами, що представляється у вигляді підмножини декартова добутку.

Ідеальним є випадок, коли всі загрози рівноцінні й не залежать друг від друга, однак для реальних систем існують більше й менш небезпечні й залежні загрози, які, у свою чергу, можуть бути домінуючими й доминуємі.

При позитивному (домінуючому) впливі персональні кольори розглянутого елемента розширюють унітарне розфарбування множини за рахунок появи нових тіл унітарних кольорів, у які входить розглянутий елемент.

При негативному (доминуємому) впливі кольору розглянутого елемента негативно впливають на відповідні персональні кольори інших елементів і на унітарні кольори множини, що приводить до припинення їхнього існування.

При нейтральному впливі умови існування кольорів інших елементів і унітарних кольорів множини залишаються незмінними.

На п'ятому етапі пропонується, на підставі умовиводів, уважати ступенем впливу в унітарному розфарбуванні її персональних складову міру близькості між розфарбуваннями сусідніх елементів поліхроматичної множини. Способи визначення міри близькості засновані на понятті відстані Хеммінгу.

У роботі експертів зручно використовувати відносну відстань Хеммінгу.

Очевидно: чим більше відносна відстань Хеммінгу, тим менше одне розфарбування впливає на сусіднє розфарбування в розглянутому универсумі.

Механізм, або алгоритм, виводу є наступною важливою частиною базової архітектури систем нечіткого виводу. Одним з найбільше широко розповсюджених алгоритмів нечіткого виводу є алгоритм Мамдані (Mamdani). Зазначений алгоритм, у порівнянні з іншими методами нечіткого виводу (Цукамото, Ларсена, Сугено), має економічність алгоритмічної реалізації, щонайкраще застосуємо для нечітких множин, що відповідають термам висновків, що ставляться до тим самим вихідним лінгвістичним змінних.

Подальші експериментальні дослідження застосування методів нечіткого виводу Цукамото, Ларсена, Сугено для рішення завдань експлікації експертних оцінок показали, що різниця отриманих значень не перевищує 15%. З огляду на ту обставину, що механізм нечіткого виводу має фізичний сенс "мір і ваг", у

роботі пропонується використовувати саме алгоритм Мамдані з умовою, що зазначений метод необхідно буде застосовувати у всіх подібних випадках.

3.2 Розробка структурної схеми

Розробимо моделі впливу поточного етапу життєвого циклу комп'ютерних мереж до загроз на вірогідність експертизи стійкості в умовах неповноти інформації на основі моделі життєвого циклу Боема.

У питаннях підвищення вірогідності експертизи на ранніх етапах життєвого циклу комп'ютерних мереж до загроз в умовах неповноти інформації необхідно розглядати два процеси:

– На початкових стадіях експертизи в експертної групи накопичена неповна кількість знань. У міру нагромадження знань зростає вірогідність експертизи.

– Із часом відбувається розвиток (приробіток) КМ, а відповідно змінюється (як правило, убік зростання) її стан стійкості.

Невизначеність (нечіткість) оцінки експертами кожної з загроз і невизначеність експертизи стійкості комп'ютерних мереж до загроз у цілому на етапі впровадження в промислову експлуатацію буде становити.

Розробимо метод підвищення вірогідності експертизи стійкості комп'ютерних мереж до загроз на основі систем дуального керування.

Для систем керування з невизначеністю в застосуванні розглянутої теорії до засобів оцінки стійкості комп'ютерних мереж до загроз основна мета складається у відтворенні впливів, що задають, і фільтрації збурювань, викликаних умовами невизначеності загроз.

В умовах неповноти апріорної інформації стратегія експертизи, що забезпечує гарантоване досягнення мети (необхідного рівня вірогідності) тільки шляхом створення надмірності (уточнення інформації, одержуваної експертом)

неефективна, а в багатьох випадках (для досить масивних класів невизначеності) нереалізована.

Природно очікувати, що ціль синтезу засобів експертизи стійкості комп'ютерних мереж до загроз досягається використанням стратегій, що забезпечують бажану вірогідність керування зі зменшенням невизначеності у вихідному описі об'єкта й загроз.

Для спіральної моделі Боема в умовах невизначеності функція кінцевого стану системи (розвитку експертної групи в часі), без обліку стану невизначеності, визначається передатною функцією запізнілої ланки.

Гранично досяжна швидкість збіжності нечітких оцінок для досить масивних класів розподілів загроз обмежена невизначеністю. Ці умови забезпечуються параметричним регулятором, структурна схема якого представлена нижче.

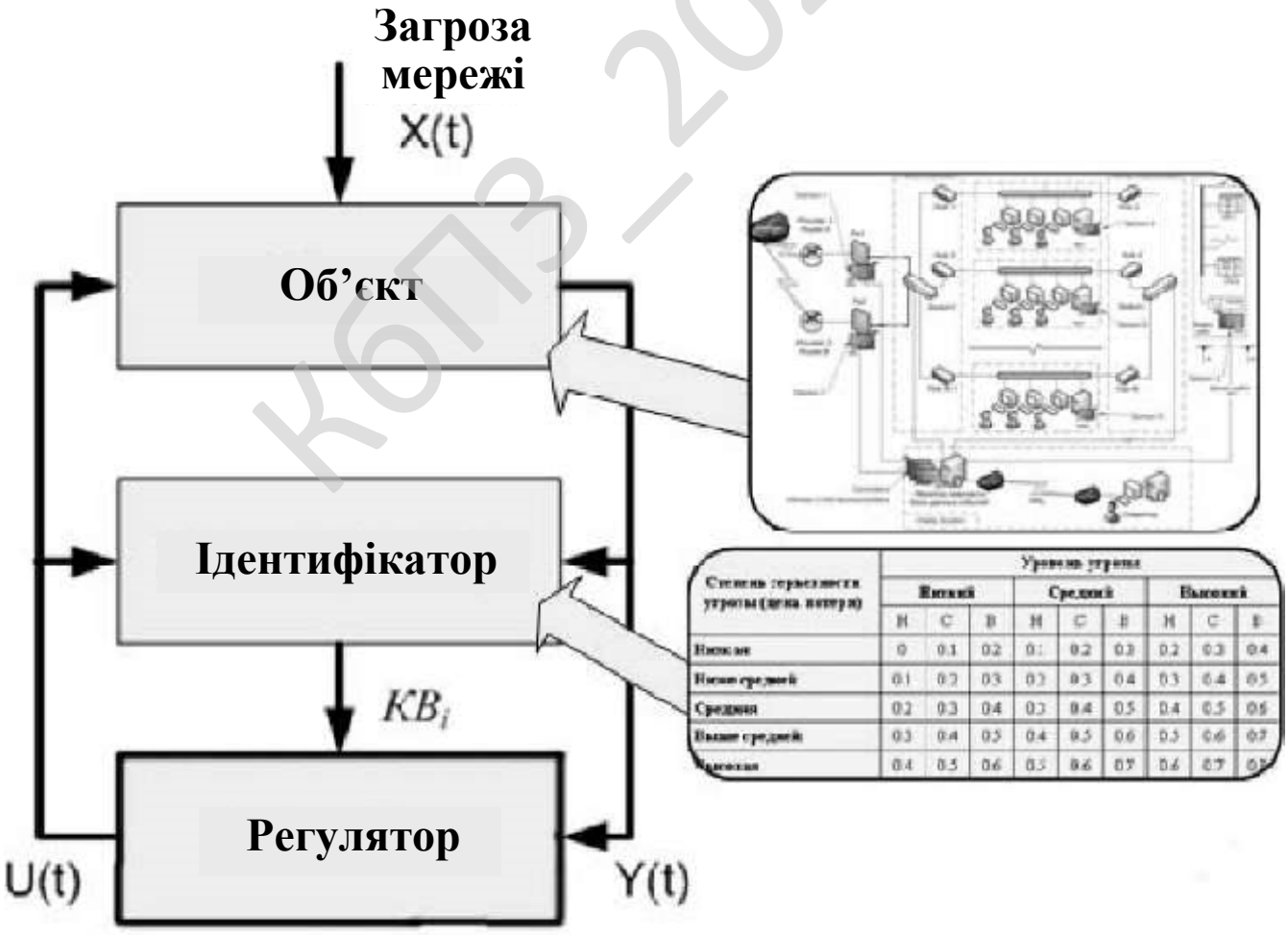

Рисунок 3.1 – Структурна схема системи

На рисунку приведені наступні позначення:

– Y(t) – показання датчика стану об'єкта є реакцією цього датчика на загрозу X(t);

– U(t) – сигнал керування;

– w0(t) – сигнал корекції.

Для розглянутої схеми приймається допущення, що результат оцінки стійкості комп'ютерних мереж до загроз одержують один раз наприкінці кожної стадії. Отже, базові характеристики стійкості, отримані в попередній главі для кожної загрози залежно від етапу життєвого циклу (ЖЦ) будуть змінюватися, становлячи траєкторії експертних оцінок.

Чисельно вірогідність експертизи зручно представити у вигляді виправлення функціонала середніх витрат:

$$\Delta J_T[U_0^T(*)] = \{\Delta J_T[U_1^T(*)], \Delta J_T[U_2^T(*)], \ldots\ldots, \Delta J_T[U_s^T(*)]\} = \frac{J_T[U_0^T(*)]}{J_n[U_0^{\ni}(*)]} \cdot 100\% .$$

Виправлення (допуски) до оцінок стійкості перебуває як відношення поточного значення вірогідності керування, до її максимального значення й виражаються у відсотках для кожної стадії поточного етапу життєвого циклу

3.3 Розробка функціональної схеми

Розглянемо функціональну схему системи у вигляді імітаційної моделі засобів підвищення вірогідності експертної оцінки стійкості телекомунікаційних систем і комп'ютерних мереж. Моделювання дозволяє відслідковувати будь-які зміни зовнішніх деструктивних впливів на стан стійкості телекомунікаційних систем і комп'ютерних мереж і оперативно реагувати на виявлені деструктивні впливи. Отримана імітаційна модель може служити електронним шаблоном стійкості телекомунікаційних систем і комп'ютерних мереж. У четвертому розділі пропонується програмно-апаратний засіб інтерпретації рівня стійкості

телекомунікаційних систем і комп'ютерних мереж, що моделює використання системи дуального керування, реалізоване в середовищі.

Вихідними даними для формування функціональної схеми яка представлена на рисунку 3.2, є експертні оцінки, представлені в зручному для уведення виді.

Далі, виконується уведення експертних оцінок у графічному режимі програми:

– у графічному інтерфейсі користувача редактори програми встановлюються позначення змінних, кожне з яких окремо утворить елементи нечіткої поліхроматичної множини загроз;·

– викликається редактор функцій приналежності, у якому задаються для кожної із шести вхідних загроз значення параметрів, що відповідають настроювання виконуються й для всіх значень оцінки;

– викликається редактор правил нечіткого виводу, у якому задаються для кожного із шести вхідних параметрів правила нечіткого виводу;

– для одержання результатів розподілу нечіткого виводу викликається програма перегляду правил нечіткого виводу;

– розглянута програма дає можливість переглянути функції приналежності в графічному режимі, але не змінювати їх;

– результати експертизи наочніше всього переглядати у вигляді поверхні нечіткого виводу, так, наприклад, графічно залежності еталонної функцій приналежності загрози відмови доступу від загрози виходу з ладу активного мережного встаткування представлена на рисунку 3.2 у вигляді поверхні нечіткого виводу;

– для відображення був викликаний графічний інтерпретатор нечіткого виводу командою;

– як видно з рисунка, загроза відмови доступу при виході з ладу активного мережного встаткування, по оцінках експертів, найбільшу небезпеку для стійкості

телекомунікаційних систем і комп'ютерних мереж може представляти в епіцентрі отриманої поверхні.

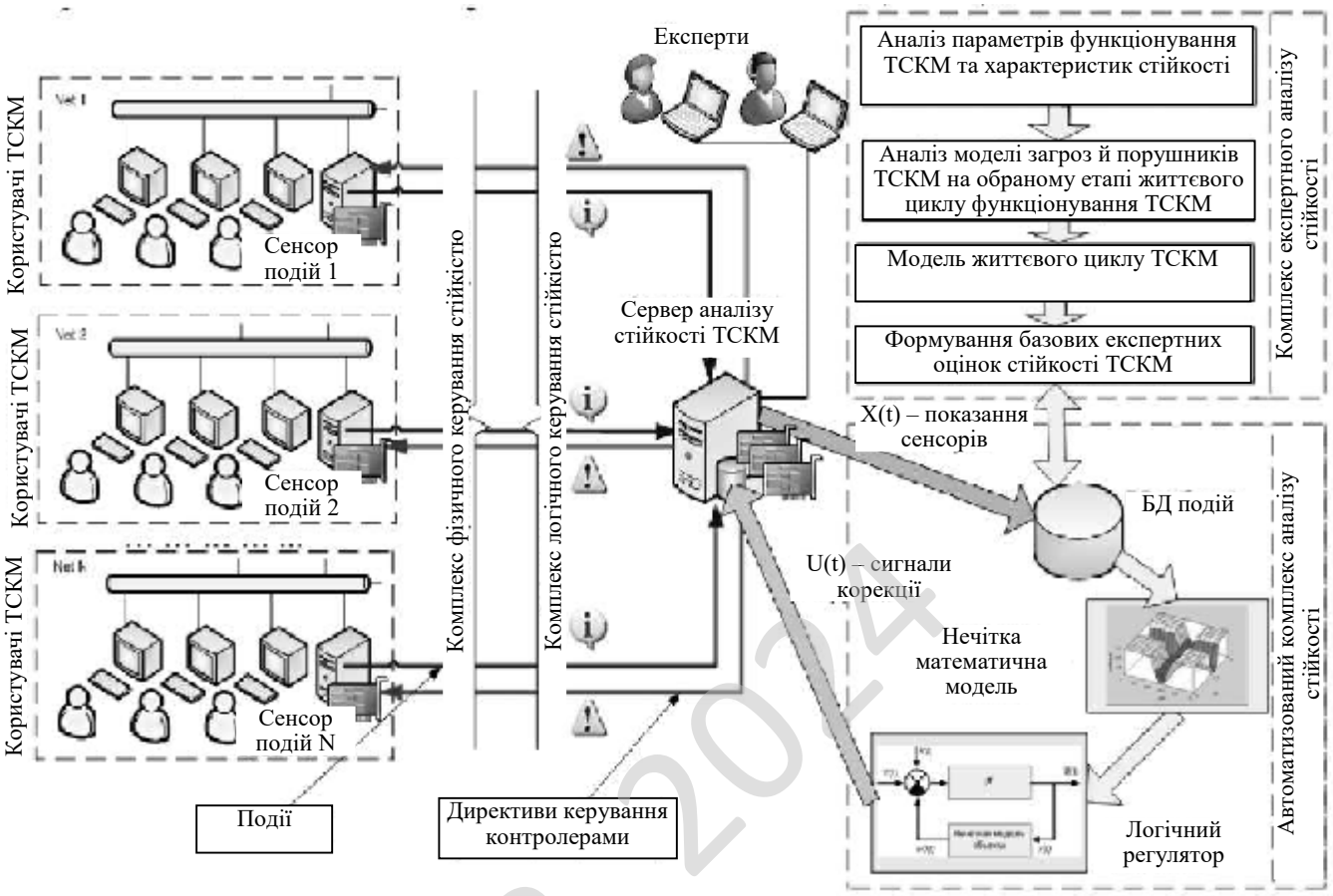

Рисунок 3.2 – Функціональна схема системи

Найменш критичними є загрози виходу з ладу пасивного мережного устаткування й загрози порушення електроживлення. Це обумовлено дублюванням пасивного мережного встаткування й застосуванням джерел безперебійного живлення.

Найбільш критичними, за результатами експлікації експертних оцінок, протягом усього етапу впровадження в промислову експлуатацію залишаються загрози відмови доступу в результаті виходу з ладу активного мережного встаткування. Це пов'язане з високим ступенем ризиків збоїв програмного забезпечення ТСКМ у результаті помилок передачі даних.

З аналізу отриманих результатів видно, що вірогідність експертних оцінок зростає від стадії до стадії. Темпи росту вірогідності щодо кожної з загроз приблизно однакові.

Однак, вірогідність оцінок кожної з загроз не однакова, тому що різним загрозам експерти із самого початку експертизи приділяли не однакову увагу, а виділяли, на свій розсуд, більш-менш значимі.

Наприклад, ризику порушення конфіденційності, що має високе значення для оцінки стійкості телекомунікаційних систем і комп'ютерних мереж, експерти споконвічно приділили меншу увагу, через велику кількість необхідної для аналізу документації.

Найменшу увагу експерти приділяли аналізу ризиків загрози порушення електроживлення й загрози виходу з ладу пасивного мережного встаткування, через високий ступінь резервування зазначених компонентів ТСКМ.

Допуски виміру основних параметрів функціонування ТСКМ для зазначених стадій етапу впровадження ТСКМ, отримані в результаті імітаційного моделювання представлені для двох стадій етапу впровадження ТСКМ.

Як показало практичне застосування розроблених засобів підвищення вірогідності експертизи для редактора програми, час на уведення вихідних даних, обробку інформації й аналіз отриманих результатів не перевищує 0,5 значення, відведеного для ухвалення рішення), що скорочує час нагромадження історичних даних на 50%.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. Після початку роботи розробленого ПЗ ми потрапляємо до інтерфейсу користувача ПЗ, де через обробник помилок ПЗ можна потрапити до статистики роботи мережі з відображенням статистики роботи TCP, IP-протоколу, UDP, ICMP-протоколу та проведенням формування звіту роботи мережі. Крім того через обробник помилок

можна потрапити до моніторингу трафіку з відображенням вихідного трафіку та вхідного трафіку. Далі надсилаючи X(t) показання сенсорів до БД подій, через логічний регулятор ми отримумо U(t) сигнали корекції.

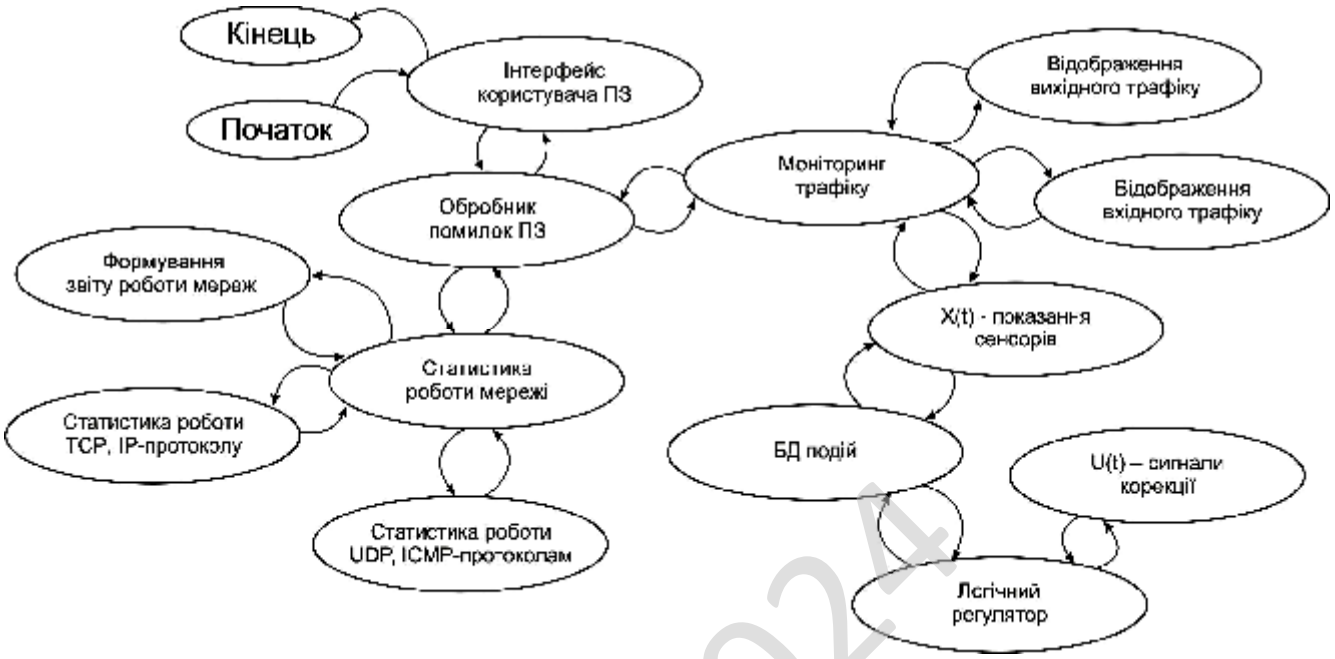

Рисунок 3.3 – Діаграма взаємодії процесів

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 ініціалізація ресурсів ПЗ. Потім здійснюється:

– Виділення пам`яті ПЗ.

– Виведення головного вікна ПЗ.

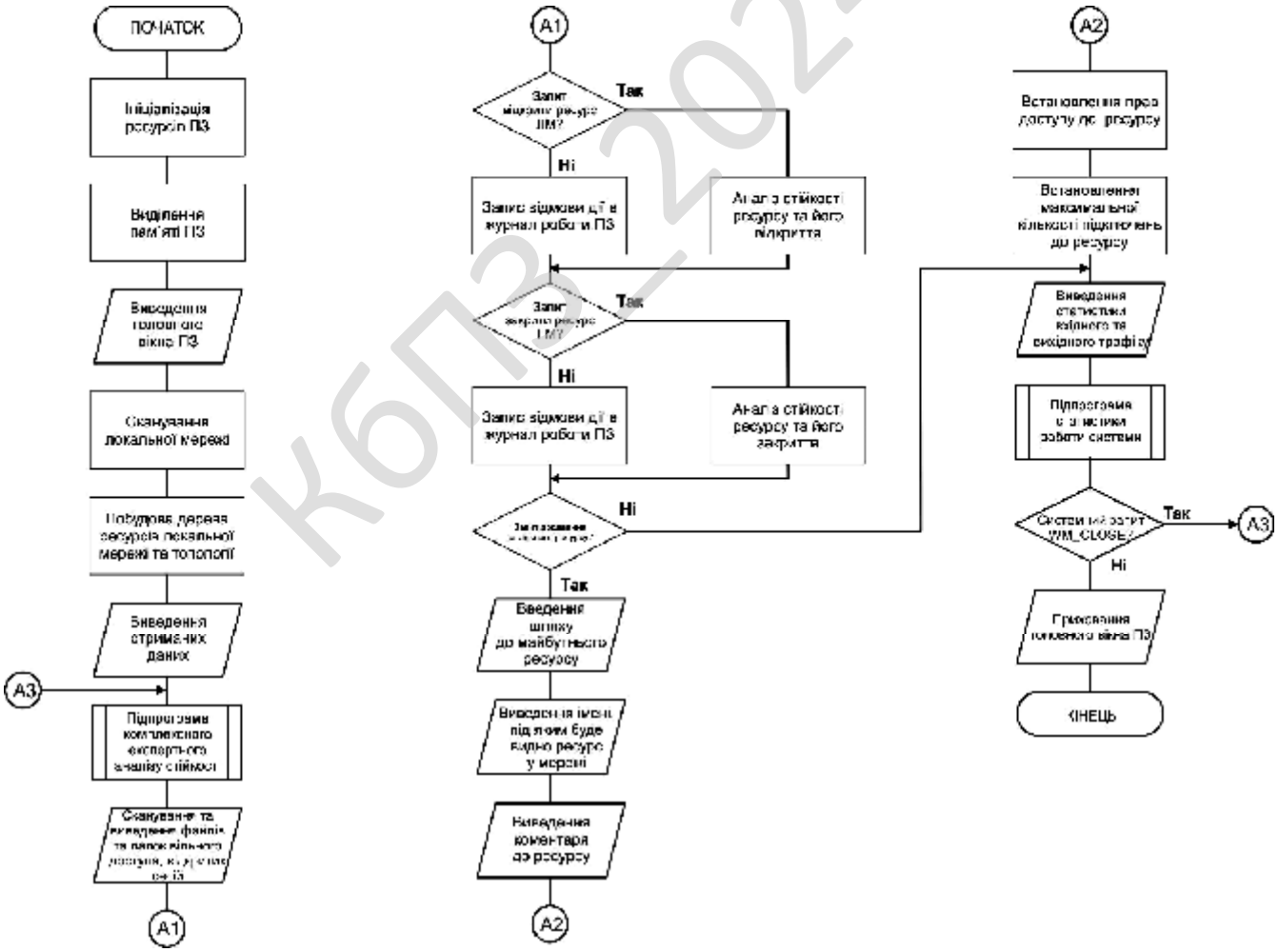

Рисунок 4.1 – Блок-схема основної програми

– Сканування локальної мережі.

– Побудова дерева ресурсів локальної мережі та топології.

– Виведення отриманих даних.

– Підпрограма комплексного експертного аналізу стійкості.

– Сканування та виведення файлів та папок вільного доступу, відкритих сесій.

– Запит відкрити ресурс ЛМ?

– Запис відмови дії в журнал роботи ПЗ.

– Запит закрити ресурс ЛМ?

– Запис відмови дії в журнал роботи ПЗ.

– Запит додавання загального ресурсу?

– Введення шляху до майбутнього ресурсу.

– Виведення імені, під яким буде видно ресурс у мережі.

– Виведення коментаря до ресурсу .

– Встановлення прав доступу до ресурсу.

– Встановлення максимальної кількості підключень до ресурсу.

– Виведення статистики вхідного та вихідного трафіку.

– Підпрограма статистики роботи системи.

– Системний запит WM_CLOSE (запит).

– Приховання головного вікна ПЗ.

На рисунку 4.2 зображено виконання підпрограми комплексного експертного аналізу стійкості:

– Відправлення X(t) – показання сенсорів.

– Аналіз параметрів функціонування ТСКМ.

– Аналіз характеристик стійкості ТСКМ.

– Аналіз моделі загроз й порушників ТСКМ на обраному етапі життєвого циклу функціонування ТСКМ.

– Побудова моделі життєвого циклу ТСКМ.

– Формування базових експертних оцінок стійкості ТСКМ.

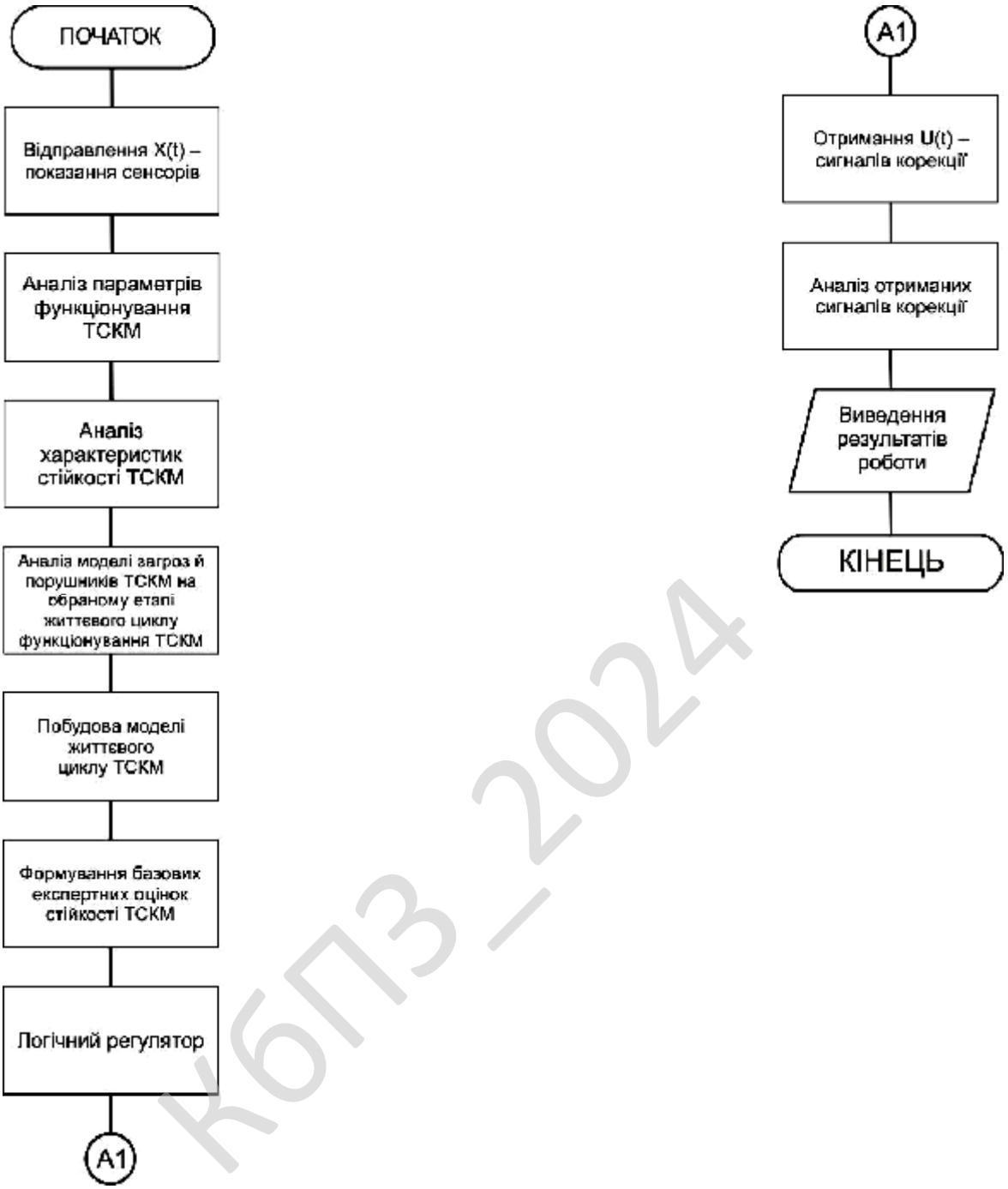

Рисунок 4.2 – Блок-схема підпрограми комплексного експертного аналізу стійкості

– Логічний регулятор.

– Отримання U(t) – сигналів корекції.

– Аналіз отриманих сигналів корекції.

– Виведення результатів роботи .

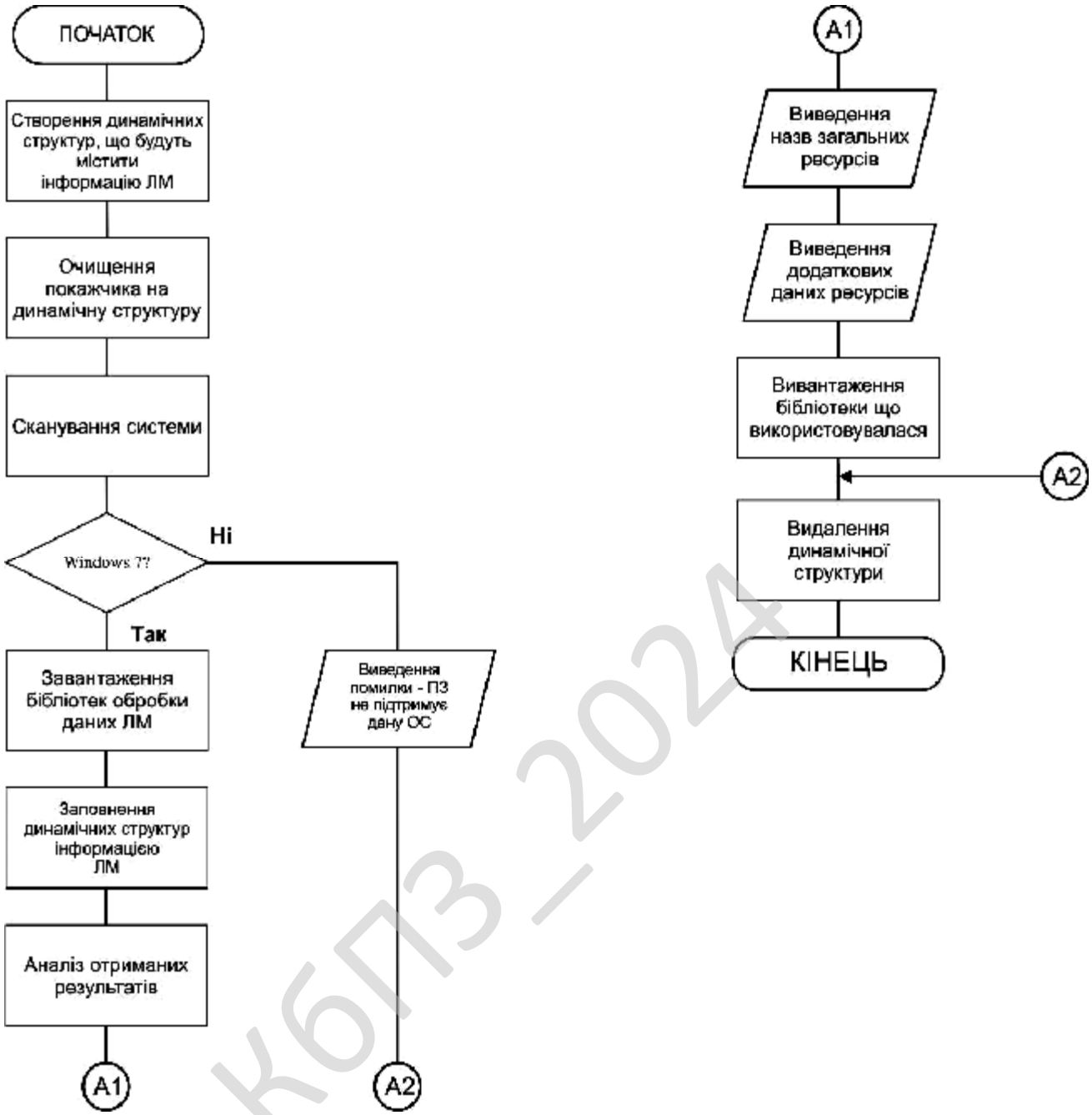

Рисунок 4.3 – Блок-схема підпрограми статистики роботи системи

На рисунку 4.3 зображено виконання підпрограми статистики роботи системи:

– Створення динамічних структур, що будуть містити інформацію ЛМ.

– Очищення покажчика на динамічну структуру.

– Сканування системи.

– Windows 7 (запит).

– Завантаження бібліотек обробки даних ЛМ.

– Заповнення динамічних структур інформацією ЛМ.

– Аналіз отриманих результатів.

– Виведення назв загальних ресурсів.

– Виведення додаткових даних ресурсів.

– Вивантаження бібліотеки що використовувалася.

– Видалення динамічної структури.

Опис алгоритмів функціонування системи

Розглянемо розроблені процедури та функції, що надають інформацію про локальні ресурси й можливість їхнього контролю та інший функціонал мого ПЗ.

Приховання й показ ресурсів. Сховати ресурс можна простим додаванням до його імені значка долара. Активувати, оберненою операцією. Це не зовсім правильно, але працює. Другий варіант сховати ресурс, це видалити його й створити його копію але із правами SHI50F_SYSTEM або вказівкою shi502_security_descriptor (для цього потрібно використовувати структуру SHARE_INFO_502).

Не можна проводити маніпуляції над наступними системними ресурсами: IPC$, ADMIN$, PRINT$, WWWROOT$, BIOSINFO$, A$, B$, C$, D$, E$, F$, G$, H$, I$, J$, K$, L$, M$, N$, O$, P$, Q$, R$,S$, T$, U$, V$, W$, X$, Y$, Z$. Відкриття доступу до цих ресурсів робить беззахисною операційну систему.

Одержання списку поточних сесій. Для визначення користувачів підключених до нашого комп'ютера скористаємося функцією Net_SessionEnum.

Оголошення функції для Windows 8:

```
var
   Net_SessionEnum:function(  pszServer      : PChar;
                              sLevel         : DWORD;
                              pbBuffer       : Pointer;
                              cbBuffer       : DWORD;
                              pcEntriesRead,
                              pcTotalAvial   : Pointer):integer; stdcall;
```

Параметри:

– pszServer – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо дивимося в себе, то даному параметру потрібно привласнити NIL.

– sLevel – повинен містити ідентифікатор структури.

– pbBuffer – повинен містити покажчик на масив структур.

– cbBuffer – повинен містити розмір масиву структур.

– pcEntriesRead – покажчик на змінну утримуючу загальну кількість структур.

– pcTotalAvial – не використовується.

Оголошення функції для Windows NT:

```
var
    Net_SessionEnum:function(   ServerName,
                                UncClientName,
                                Username        : PWChar;
                                Level           : DWORD;
                                bufptr          : Pointer;
                                prefmaxlen      : DWORD;
                                entriesread,
                                totalentries,
                                resume_handle   : LPDWORD):DWORD; stdcall;
```

Параметри:

– ServerName – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо дивимося в себе, то даному параметру потрібно привласнити NIL.

– UncClientName – містить покажчик на рядок утримуючий ім'я сесії, про яку ми хочемо одержати інформацію, якщо потрібно переглянути всі сесії параметру потрібно привласнити NIL.

– UserName – містить покажчик на рядок утримуючий ім'я користувача, про яке ми хочемо одержати інформацію, якщо потрібно переглянути всі, параметру потрібно привласнити NIL.

– Level – повинен містити ідентифікатор структури.

– Bufptr – повинен містити адресу покажчика на масив структур.

– Prefmaxlen – повинен містити максимальну довжину повернутих даних у байтах, якщо не ставити обмеження, то даному параметру потрібно привласнити DWORD(-1).

– Entriesread – повинен містити покажчик на змінну, в яку запишеться кількість загальних ресурсів доступних на даний момент.

– Totalentries – не використовується.

– Resume_handle – не використовується, повинен бути NIL.

Існує також 6 типів структур, що передаються функції Net_SessionEnum:

– SESSION_INFO_0 – тільки Windows NT.

– SESSION_INFO_1 – тільки Windows NT.

– SESSION_INFO_2 – тільки Windows NT.

– SESSION_INFO_10 – тільки Windows NT.

– SESSION_INFO_502 – тільки Windows NT.

– session_info_50 – тільки Windows 8.

Розглянемо дві з них як найбільш повні по своїй суті.

Завершення сесій. Для завершення відкритих сесій будемо використовувати функцію Net_SessionDel.

Оголошення функції для Windows 8:

```
var
   Net_SessionDel:function(   pszServer      : PChar;
                              PszClientName  : PChar;
                              SReserved      : SmallInt):DWORD; stdcall;
```

Параметри:

– pszServer – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо завершується сесія в себе, то даному параметру потрібно привласнити NIL.

– pszClientName – повинен містити ім'я клієнта, чия сесія завершується.

– sReserved – повинен містити унікальний ключ для завершення сесії (той який ми одержали попередньою функцією).

Оголошення функції для Windows NT:

```
var
    Net_SessionDel:function(    ServerName,
                                UncClientName,
                                Username       :PWChar):DWORD; stdcall;
```

Параметри:

– ServerName – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо завершується сесія в себе, то даному параметру потрібно привласнити NIL.

– uncClientName – повинен містити ім'я клієнта, чия сесія завершується, якщо параметр NIL, завершаться всі сесії зазначені в параметрі username.

– username – повинен містити ім'я користувача, чия сесія завершується, якщо параметр NIL, завершаться всі сесії зазначені в параметрі uncClientName.

Як можна помітити, ніяких структур дані функції не використовують. Напишемо процедуру яка буде завершувати обрану нами в lvSessions сесію по ім'ю клієнта. От як цей код виглядає:

```
procedure TMainForm.btnCloseSessionClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  CNameNT: PWideChar;
  CName9x: PAnsiChar;
  Key:SmallInt;
  i: Integer;
begin
  if not IsNT(OS) then Close;
//З'ясовуємо тип системи
  if not Assigned(lvSessions.Selected) then Exit;
  i:= lvSessions.Selected.Index;
//Визначаємо номер обраної сесії
  if OS then begin
    FLibHandle := LoadLibrary('NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @Net_SessionDelNT := GetProcAddress(FLibHandle,
                         'Net_SessionDel');
    if not Assigned(Net_SessionDelNT) then
    begin
```

```
        FreeLibrary(FLibHandle);
        Exit;
      end;
      //Перетворимо дані в необхідний вид
      CNameNT:=PWChar(WideString('\\'+lvSessions.Items.
              Item[i].Caption));
      Net_SessionDelNT(nil,CNameNT,nil);
    end else begin
      FLibHandle := LoadLibrary('SVRAPI.DLL');
      if FLibHandle = 0 then Exit;
      @Net_SessionDel := GetProcAddress(FLibHandle, 'Net_SessionDel');
      if not Assigned(Net_SessionDel) then
      begin
        FreeLibrary(FLibHandle);
        Exit;
      end;
// Перетворимо дані в необхідний вид
      CName9x := PAnsiChar(lvSessions.Items.Item[i].Caption);
      key := SessionCloseKey[i];
//Беремо ключ із масиву
      Net_SessionDel(nil,CName9x,Key);
    end;
    FreeLibrary(FLibHandle);
end;
```

Net_ShareEnum – за допомогою цієї функції ми одержимо дані про всі свої й чужі загальні ресурси.

Оголошення функції для Windows 8:

```
var
    Net_ShareEnum :function (pszServer       : PChar;
                             sLevel     : Cardinal;
                             pbBuffer        : PChar;
                             cbBuffer        : Cardinal;
                             pcEntriesRead,
                             pcTotalAvail    : Pointer):DWORD; stdcall;
```

Параметри:

– pszServer – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо виконуємо в себе то даному параметру можна привласнити NIL.

– sLevel – повинен містити ідентифікатор структури.

– pbBuffer – повинен містити покажчик на масив структур.

– cbBuffer – повинен містити розмір масиву структур.

– pcEntriesRead – повинен містити покажчик на змінну, в яку запишеться кількість спільних ресурсів доступних на даний момент.

– pcTotalAvail – не використовується.

Оголошення функції для Windows NT:

```
var
    Net_ShareEnum :function (ServerName        :PWChar;
                             Level             :DWORD;
                             Bufptr            :Pointer;
                             Prefmaxlen        :DWORD;
                             EntriesRead,
                             TotalEntries,
                             resume_handle:LPDWORD): DWORD; stdcall;
```

Параметри:

– ServerName – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо виконуємо в себе то даному параметру можна привласнити NIL.

– Level – повинен містити ідентифікатор структури.

– Bufptr – повинен містити адресу покажчика на масив структур.

– Prefmaxlen – повинен містити максимальну довжину повернутих даних у байтах, якщо не ставити обмеження, то даному параметру потрібно привласнити DWORD(-1).

– EntriesRead – повинен містити покажчик на змінну, в яку запишеться кількість спільних ресурсів доступних на даний момент.

– TotalEntries – не використовується.

– Resume_handle – не використовується, повинен бути NIL.

У випадку успішного виконання результат обох функцій дорівнює нулю. Слід звернути увагу на те, що функція, яка використовується в Windows 8 одержує саме покажчик на масив структур, у той час як інша функція одержує адресу покажчика, це критично.

Результати виконання будуть збережені в масиві структур передані функції при її виклику.

Існує 6 типів структур функції Net_ShareEnum які передаються: SHARE_INFO_0 – тільки Windows NT; SHARE_INFO_1 – тільки Windows NT; share_info_1 – тільки Windows 8; SHARE_INFO_2 – тільки Windows NT; share_info_50 – тільки Windows 9 XP; SHARE_INFO_502 – тільки Windows NT.

Зупинимося на двох з них.

Структура share_info_50. Оголошення структури:

```
type
 TShareInfo50 = packed record
  shi50_netname         : array [0..12] of Char;
  shi50_type            : Byte;
  shi50_flags           : Word;
  shi50_remark          : PChar;
  shi50_path            : PChar;
  shi50_rw_password     : array [0..8] of Char;
  shi50_ro_password     : array [0..8] of Char;
 end;
```

Поля:

– shi50_netname – містить рядок, що утримує мережне ім'я ресурсу;

– shi50_type – визначає тип ресурсу (докладніше в MSDN);

– shi50_flags – містить інформацію про права доступу до ресурсу;

– shi50_remark – покажчик на рядок, що містить необов'язковий коментар до ресурсу;

– shi50_path – містить локальне розташування ресурсу;

– shi50_rw_password – містить пароль на запис/читання;

– shi50_ro_password – містить пароль на читання.

Реально одержати значення двох останніх полів можна тільки при одержанні інформації про свій комп'ютер, в інших випадках вони залишаються порожніми.

Структура SHARE_INFO_2. Оголошення структури:

```
type
 TShareInfo2 = packed record
```

```
  shi2_netname                : PWChar;
  shi2_type                   : DWORD;
  shi2_remark                 : PWChar;
  shi2_permissions            : DWORD;
  shi2_max_uses               : DWORD;
  shi2_current_uses           : DWORD;
  shi2_path                   : PWChar;
  shi2_passwd                 : PWChar;
 end;
 PShareInfo2 = ^TShareInfo2;
 TShareInfo2Array = array [0..512] of TShareInfo2;
 PShareInfo2Array = ^TShareInfo2Array;
```

Поля:

– shi2_netname – містить покажчик на рядок, що утримує ім'я ресурсу;

– shi2_type – визначає тип ресурсу (докладніше в MSDN);

– shi2_remark – покажчик на рядок, що містить необов'язковий коментар до ресурсу;

– shi2_permissions – містить інформацію про права доступу до ресурсу;

– shi2_max_uses – визначає максимальну кількість підключень до ресурсу;

– shi2_current_uses – визначає кількість поточних підключень;

– shi2_path – містить покажчик на рядок утримуючий локальне розташування ресурсу;

– shi2_passwd – містить покажчик на рядок утримуючий пароль.

Визначення вхідного та вихідного трафіку. Для цього досить використовувати всього лише одну функцію бібліотеки IPHLPAPI.DLL, що поставляється з усіма версіями Windows. Оголошення функції (всі версії Windows):

```
var
GetIfTable:function(    pIfTable: PMibIfTable;
                        pdwSize : PULONG;
                        bOrder  : Boolean ): DWORD; stdcall;
```

Параметри:

– pIfTable – повинен містити покажчик на структуру.

– pdwSize – повинен містити розмір структури.

– bOrder – указує, чи потрібне сортування в масиві, що повертається.

Як перший параметр функція використовує покажчик на структуру. Опис структури:

```
type
 TMibIfTable = packed record
   dwNumEntries : DWORD;
   Table        : TMibIfArray;
 end;
 PMibIfTable = ^ TMibIfTable;
```

Поля:

– dwNumEntries – визначає розмірність масиву представленого другим параметром.

– Table – є масивом структур.

Структура сама по собі вкрай неінформативна, нас цікавить друге її поле, що також представляє собою структуру. Опис структури:

```
type
  TMibIfRow = packed record
    wszName             : array[0..255] of WideChar;
    dwIndex             : DWORD;
    dwType              : DWORD;
    dwMtu               : DWORD;
    dwSpeed             : DWORD;
    dwPhysAddrLen       : DWORD;
    bPhysAddr           : array[0..7] of Byte;
    dwAdminStatus       : DWORD;
    dwOperStatus        : DWORD;
    dwLastChange        : DWORD;
    dwInOctets          : DWORD;
    dwInUcastPkts       : DWORD;
    dwInNUCastPkts      : DWORD;
    dwInDiscards        : DWORD;
  end;
  TMibIfArray = array [0..512] of TMibIfRow;
  PMibIfRow = ^TMibIfRow;
  PmibIfArray = ^TmibIfArray;
```

Поля:

– wszName – покажчик на рядок утримуючий ім'я інтерфейсу.

– dwIndex – визначає індекс інтерфейсу.

– dwType – визначає тип інтерфейсу.

– dwMtu – визначає максимальну швидкість передачі.

– dwSpeed – визначає поточну швидкість передачі в бітах у секунду.

– dwPhysAddrLen – визначає довжину адреси, що втримується в bPhysAddr.

– bPhysAddr – містить фізичну адресу інтерфейсу (його трохи видозмінений MAC-адрес).

– dwAdminStatus – Визначає активність інтерфейсу.

– dwOperStatus – містить поточний статус інтерфейсу.

– dwLastChange – містить останній змінений статус.

– dwInOctets – містить кількість байтів прийнятих через інтерфейс.

– dwInUcastPkts – містить кількість спрямованих пакетів прийнятих інтерфейсом.

– dwInNUCastPkts – містить кількість ненаправлених пакетів прийнятих інтерфейсом.

– dwInDiscards – містить кількість забракованих вхідних пакетів (навіть якщо вони не містили помилки).

В цій структурі утримується безліч інформації, що ми й будемо використовувати. Код визначення поточного вхідного/вихідного трафіку виглядає наступним чином:

```
procedure TMainForm.tmrTrafficTimer(Sender: TObject);
// Допоміжна функція, що перетворить MAC-адресу до "нормального" виду
//Визначаємо спеціальний тип, щоб можна було передати у функцію масив
  type TMAC = array [0..7] of Byte;
//Як перше значення масив, друге значення, розмір даних у масиві
  function GetMAC(Value: TMAC; Length: DWORD): String;
  var
    i: Integer;
```

```
  begin
    if Length = 0 then Result := ' 00-00-00' else
    begin
      Result := '';
      for i:= 0 to Length -2 do
        Result := Result + IntToHex(Value[i],2)+'-';
      Result := Result + IntToHex(Value[ Length-1],2);
    end;
  end;
//Сама процедура
var
  FLibHandle    : THandle;
  Table         : TMibIfTable;
  i             : Integer;
  Size          : Integer;
begin
  tmrTraffic.Enabled := False;
// Зупиняємо таймер
  lvTraffic.Items.BeginUpdate;
  lvTraffic.Items.Clear;
// Очищаємо список
  FLibHandle := LoadLibrary('IPHLPAPI.DLL');
// Завантажуємо бібліотеку
  if FLibHandle = 0 then Exit;
  @GetIfTable := GetProcAddress(FLibHandle, 'GetIfTable');
  if not Assigned(GetIfTable) then
  begin
    FreeLibrary(FLibHandle);
    Close;
  end;
  Size := SizeOf(Table);
  if GetIfTable(@Table, @Size, False ) = 0 then
// Виконуємо функцію
    for i:= 0 to Table.dwNumEntries-1 do begin
      with lvTraffic.Items.Add do begin
// Виводимо результати
        Caption := String(Table.Table[i].bDescr);
// Найменування інтерфейсу
        SubItems.Add(GetMAC(TMAC(Table.Table[i].bPhysAddr),
          Table.Table[i].dwPhysAddrLen));
// MAC адреса
        SubItems.Add(IntToStr(Table.Table[i].dwInOctets));
```

```
// Усього прийнято байтів
        SubItems.Add(IntToStr(Table.Table[i].dwOutOctets));
// Усього відправлено байтів
      end;
    end;
  lvTraffic.Items.EndUpdate;
  FreeLibrary(FLibHandle);
  tmrTraffic.Enabled := True;
// Активуємо таймер
end;
```

Тип даних TMAC було створено для передачі масиву, у якому втримується сам MAC адресу у функцію для перетворення його в більш звичний вигляд.

Код TMAC(Table.Table[i].bPhysAddr) – це передача масиву, обов'язково потрібно вказати, що масив передається як тип TMAC, у протилежному випадку компілятор видасть помилку несумісності типів.

Одержання списку відкритих файлів. Розглянемо функцію NetFileEnum. Оголошення функції для Windows 8:

```
var
NetFileEnum:function(  pszServer,
                       pszBasePath    : PChar;
                       sLevel         : DWORD;
                       pbBuffer       : Pointer;
                       cbBuffer       : DWORD;
                       pcEntriesRead,
                       pcTotalAvail   : Pointer):Integer; stdcall;
```

Параметри:

– pszServer – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо дивимося в себе, то даному параметру потрібно привласнити NIL.

– pszBasePath – повинен містити каталог, відкриті файли з якого ми хочемо подивитися, якщо ми хочемо переглянути всі відкриті файли параметру потрібно привласнити NIL.

– sLevel – повинен містити ідентифікатор структури.

– pbBuffer – повинен містити покажчик на масив структур.

– cbBuffer – повинен містити розмір структури.

– pcEntriesRead – покажчик на змінну утримуючу загальну кількість структур.

– pcTotalAvial – не використовується.

Тепер напишемо код який покаже нам список відкритих файлів на нашому комп'ютері:

```
procedure TMainForm.btnGetFilesClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  FileInfoNT: PFileInfo3Array;
  FileInfo9x: array [0..512] of TFileInfo50;
  TotalEntries,EntriesReadNT: DWORD;
  EntriesRead,TotalAvial: Word;
  i:integer;
begin
  lvfiles.Items.Clear;
  if not IsNT(OS) then Close;
// З'ясовуємо тип системи
  if OS then begin
//Код для NT
    FLibHandle := LoadLibrary('NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileEnumNT := GetProcAddress(FLibHandle, 'NetFileEnum');
    if not Assigned(NetFileEnumNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    FileInfoNT := nil;
    if NetFileEnumNT(nil,nil,nil,3,
                    @FileInfoNT,DWORD(-1),@entriesreadNT,
                     @totalentries, nil)=0 then
    for i:=0 to EntriesReadNT-1 do
    begin
      with lvFiles.Items.Add do
// Заповнення даними зі структури
      begin
        Caption := string(IntToStr(FileInfoNT^[i].fi3_id));
// Ідентифікатор
        SubItems.Add(FileInfoNT^[i].fi3_pathname);
```

```
// Шлях до файлу
        SubItems.Add(FileInfoNT^[i].fi3_username);
// Ім'я користувача
      end;
    end;
  end else begin
// Код для Windows 8
   FLibHandle := LoadLibrary('SVRAPI.DLL');
   if FLibHandle = 0 then Exit;
   @NetFileEnum := GetProcAddress(FLibHandle, 'NetFileEnum');
    if not Assigned(NetFileEnum) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
   if NetFileEnum (nil, nil,50,@FileInfo9x, SizeOf(FileInfo9x),
                   @EntriesRead,@TotalAvial) = 0 then
   for i:=0 to EntriesRead-1 do
   begin
      with lvFiles.Items.Add do
// Заповнення даними зі структури
      begin
        Caption := string(IntToStr(FileInfo9x[i].fi50_id));
// Ідентифікатор
        SubItems.Add(FileInfo9x[i].fi50_pathname);
// Шлях до файлу
        SubItems.Add(FileInfo9x[i].fi50_username);
// Ім'я користувача
      end;
   end;
 end;
 FreeLibrary(FLibHandle);
end;
```

Закриття відкритого файлу. Тепер розглянемо функції NetFileClose і NetFileClose2, за допомогою яких будемо закривати відкриті по мережі файли.

Функція NetFileClose2 – використовується тільки в Windows 8:

```
var
 NetFileClose2:function(       pszServer      :PChar;
                               UlFileId       :LongWord):DWORD; stdcall;
```

Поля:

– pszServer – повинен містити ім'я віддаленого комп'ютера, якщо закриваємо в себе, то даному параметру потрібно привласнити NIL.

– UlFileId – повинен містити унікальний ідентифікатор відкритого файлу.

Функція NetFileClose – використовується тільки в Windows NT:

```
var
   NetFileClose:function(      ServerName      :PWideChar;
                               FileId  :DWORD):DWORD; stdcall;
```

Поля:

– ServerName – повинен містити ім'я віддаленого комп'ютера, якщо закриваємо в себе, то даному параметру потрібно привласнити **NIL**.

– FileId – повинен містити унікальний ідентифікатор відкритого файлу.

Як перший параметр передамо NIL (закриваємо в себе), у якості другого, ідентифікатор файлу, що відображається у полі ID. От приклад коду:

```
procedure TMainForm.btnCloseFileClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  i: Integer;
begin
  if not IsNT(OS) then Close;
// З'ясовуємо тип системи
  if not Assigned(lvFiles.Selected) then Exit;
  i:= lvFiles.Selected.Index;
// Визначаємо номер обраного файлу
  if OS then begin
// Код для NT
    FLibHandle := LoadLibrary('NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileClose := GetProcAddress(FLibHandle, 'NetFileClose');
    if not Assigned(NetFileClose) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    NetFileClose(nil,StrToInt(lvFiles.Items.Item[i].Caption));
// Закриваємо
```

```
  end else begin
// Код для Windows 8
    FLibHandle := LoadLibrary('SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileClose2 := GetProcAddress(FLibHandle, 'NetFileClose2');

    if not Assigned(NetFileClose2) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    NetFileClose2(nil,StrToInt(lvFiles.Items.Item[i].Caption));
// Закриваємо
  end;
  FreeLibrary(FLibHandle);
end;
```

Закриття локального ресурсу. Тепер розглянемо функцію Net_ShareDel яка дозволяє закрити обраний загальний ресурс.

Оголошення функції для Windows 8:

```
var
   Net_ShareDel:function  (pszServer,
                           pszNetName :PChar;
                           usReserved :Word ): DWORD; stdcall;
```

Параметри:

– ServerName – повинен містити ім'я віддаленого комп'ютера, якщо закриваємо свої ресурси, то даному параметру потрібно привласнити NIL.

– NetName – покажчик на рядок, який містить ім'я ресурсу, що закривається.

– Reserved – не використовується, повинен бути рівним нулю.

Обидві функції не використовують ніяких структур. Нас цікавить тільки другий параметр, що містить ім'я ресурсу, що закривається.

Як ім'я передається не шлях до ресурсу, а саме ім'я ресурсу яке ми визначили за допомогою коду даного вище.

У випадку успішного виконання функцій, їхній результат буде рівний нулю.

Відкриття локального ресурсу. Під відкриттям локального ресурсу мається на увазі дозвіл загального доступу до нього.

Для вікриття локального ресурсу скористаємося функцією Net_ShareAdd**.**

Оголошення функції для Windows 8:

```
var
   Net_ShareAdd: function (    pszServer       :Pchar;
                               SLevel          :Cardinal;
                               PbBuffer        :PChar;
                               CbBuffer        :Word):DWORD; stdcall;
```

Параметри:

– pszServer – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо відкриваємо локальний ресурс, то даному параметру потрібно привласнити NIL.

– sLevel – повинен містити ідентифікатор структури.

– pbBuffer – повинен містити покажчик на структуру.

– cbBuffer – повинен містити розмір структури.

Оголошення функції для Windows NT:

```
var
   Net_ShareAdd: function (    servername      : PWideChar;
                               level           : DWORD;
                               buf             : Pointer;
                               parm_err        : LPDWORD): DWORD; stdcall;
```

Параметри:

– servername – повинен містити ім'я віддаленого комп'ютера, на якому повинна виконається функція, якщо відкриваємо локальний ресурс то даному параметру потрібно привласнити NIL.

– level – повинен містити ідентифікатор структури.

– buf – повинен містити покажчик на структуру.

– parm_err – містить покажчик на член структури, що викликає помилку.

У функції для NT також використовується покажчик а не адреса покажчика. Обидві функції використовують структури описані вище.

Для демонстрації їхніх можливостей скористаємося структурами

share_info_50 і SHARE_INFO_502. Код, що дозволяє вибрати папку, загальний доступ до якої ми хочемо відкрити:

```
procedure TMainForm.btnAddSharesClick(Sender: TObject);
const
  STYPE_DISKTREE = 0;
  ACCESS_ALL = 258;
  SHI50F_FULL = 258;
var
  FLibHandle : THandle;
  Share9x : TShareInfo50;
  ShareNT : TShareInfo2;
  TmpDir, TmpName: String;
  TmpDirNT, TmpNameNT: PWChar;
  OS: Boolean;
  TmpLength: Integer;
begin
  TmpDir := SelectDirectory;
//Визначаємо шлях до майбутнього ресурсу
  TmpName := InputBox('Share name','Enter name','Test');
//Визначаємо ім'я під яким його буде видно у мережі
  if TmpDir = '' then Exit;
  if not IsNT(OS) then Close;
//З'ясовуємо тип системи
  if OS then begin //Код для NT
    FLibHandle := LoadLibrary('NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @Net_ShareAddNT := GetProcAddress(FLibHandle,'Net_ShareAdd');
    if not Assigned(Net_ShareAddNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    TmpLength := SizeOF(WideChar)*256;
// Визначаємо необхідний розмір
    GetMem(TmpNameNT, TmpLength);
// Конвертуємо в PWChar
    StringToWideChar(TmpName, TmpNameNT, TmpLength);
    ShareNT.shi2_netname := TmpNameNT;
// Ім'я
    ShareNT.shi2_type := STYPE_DISKTREE;
// Тип ресурсу
```

```
    ShareNT.shi2_remark := '';
// Коментар
    ShareNT.shi2_permissions := ACCESS_READ;
// Доступ
    ShareNT.shi2_max_uses := DWORD(-1);
 // Кількість макс. підключ.
    ShareNT.shi2_current_uses := 0;
// Кількість тік подкл.
    GetMem(TmpDirNT, TmpLength);
    StringToWideChar(TmpDir, TmpDirNT, TmpLength);
    ShareNT.shi2_path := TmpDirNT;
//Шлях до ресурсу
    ShareNT.shi2_passwd := nil;
//Пароль
    Net_ShareAddNT(nil,2,@ShareNT,nil);
//Додаємо ресурс
    FreeMem (TmpNameNT);
//звільняємо пам'ять
    FreeMem (TmpDirNT);
  end else begin
//Код для 9x
    FLibHandle := LoadLibrary('SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @Net_ShareAdd := GetProcAddress(FLibHandle,'Net_ShareAdd');
    if not Assigned(Net_ShareAdd) then
    begin
      FreeLibrary(FLibHandle);
      Close;
    end;
    FillChar(Share9x.shi50_netname, SizeOf(Share9x.shi50_netname), #0);
    move(TmpName[1],Share9x.shi50_netname[0],Length(TmpName));
//Ім'я
    Share9x.shi50_type := STYPE_DISKTREE; //
// Тип ресурсу
    Share9x.shi50_flags := SHI50F_RDONLY; //
// Доступ
    FillChar(Share9x.shi50_remark,
      SizeOf(Share9x.shi50_remark), #0);
// Коментар
    FillChar(Share9x.shi50_path,
      SizeOf(Share9x.shi50_path), #0);
    Share9x.shi50_path := PAnsiChar(TmpDir);
```

```
// Шлях до ресурсу
    FillChar(Share9x.shi50_rw_password,
      SizeOf(Share9x.shi50_rw_password), #0);
// Пароль повного доступу
    FillChar(Share9x.shi50_ro_password,
      SizeOf(Share9x.shi50_ro_password), #0);
// Пароль для читання
    Net_ShareAdd(nil,50,@Share9x,SizeOf(Share9x));
  end;
  FreeLibrary(FLibHandle);
end;
```

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм SEED – у криптографії симетричний блоковий криптоалгоритм на основі Мережі Фейстеля, розроблений Корейським агентством інформаційної безпеки (Korean Information Security Agency, KISA) в 1998 році. В алгоритмі використовується 128-бітний блок і ключ довжиною 128 біт. Алгоритм одержав широке поширення й використовується фінансовими й банківськими структурами, виробничими підприємствами й бюджетними установами Південної Кореї, оскільки 40-бітний SSL не забезпечує на даний момент мінімально необхідного рівня безпеки. Агентством по захисту інформації специфіковане використання шифру SEED у протоколах TLS і S/MIME. У той же час, алгоритм SEED не реалізований у більшості сучасних браузерів і інтернет-додатків, що утрудняє його використання в даній сфері поза межами Південної Кореї.

SEED являє собою мережу Фейстеля з 16 раундами, 128-бітовими блоками й 128-бітовим ключем. Алгоритм використовує дві 8×8 таблиці підстановки, які, як такі з Safer, виведені з дискретного зведення в ступінь (у цьому випадку, x^{247} і x^{251} – плюс деякі «несумісні операції»). Це є деякою подібністю с MISTY1 у рекурсивності його структури: 128-бітовий повний шифр – мережа Фейстеля з F-функцією, що впливає на 64-бітові половини, у той час як сама F-функція –

Мережа Фейстеля, складена з G-функції, що впливає на 32-розрядні половини. Однак рекурсія не простягнеться далі, тому що G-функція – не Мережа Фейстеля. В G-функції 32-розрядне слово розглядають як чотири 8-бітових байта, кожний з яких проходить через одну або іншу таблицю підстановки, потім поєднується в помірковано комплексному наборі булевих функцій таким чином, що кожний біт виводу залежить від 3 з 4 вхідних байтів.

SEED має складний ключовий розклад, генеруючи тридцять два 32-розрядних додаткових символу, використовуючи G-функції на серіях обертань вихідного неопрацьованого ключа, комбінованого зі спеціальними раундовими константами (як в TEA) від «Золотого співвідношення» (англ. Golden ratio).

Згідно з дослідженнями KISA, алгоритм SEED «надійно протистоїть відомим атакам».

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програма має простий та інтуїтивно зрозумілий інтерфейс, який зображений на рисунку 5.1

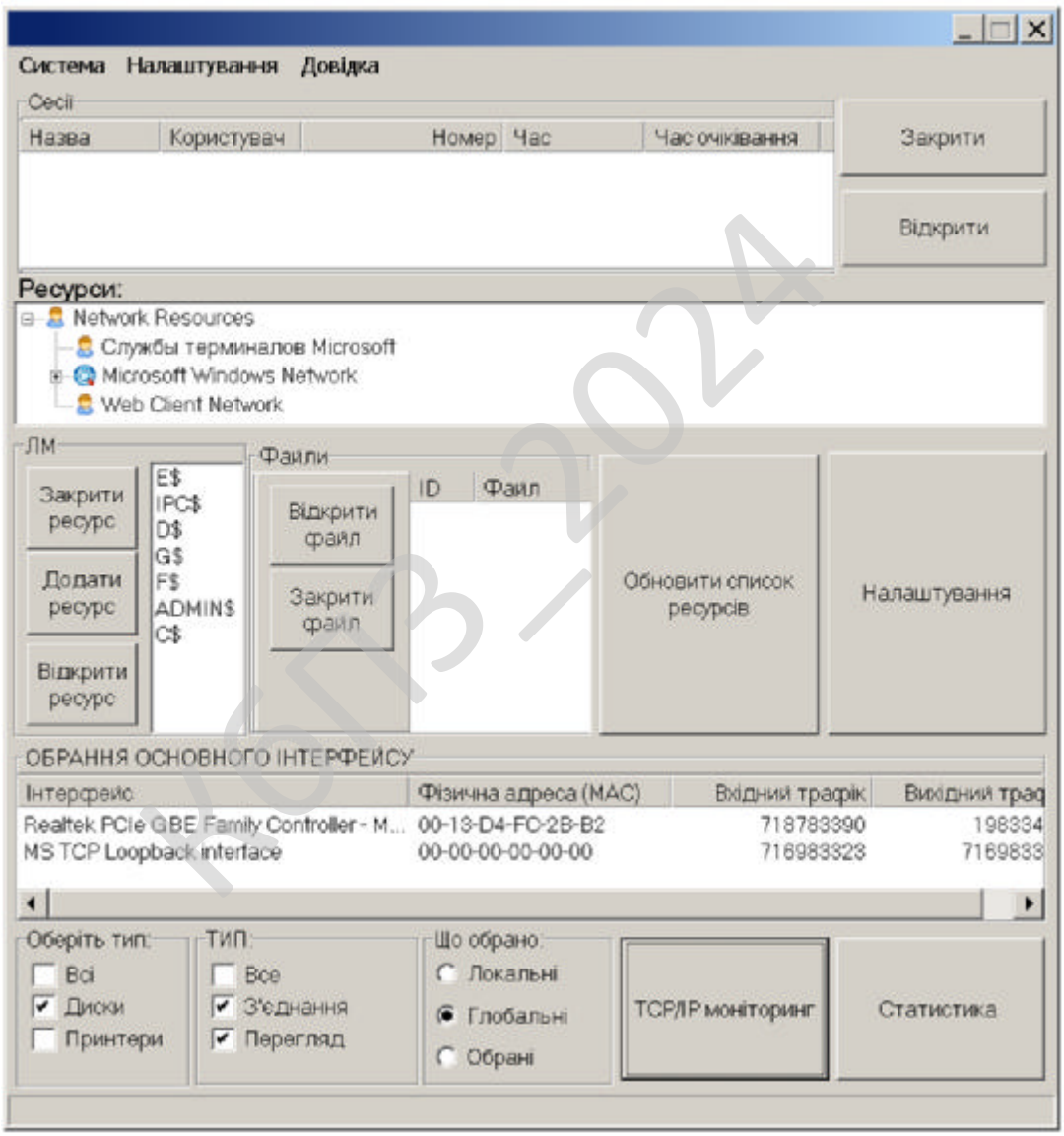

Рисунок 5.1 – Головне вікно програми

З нього видно, що інтерфейс користувача програми складається з таких логічних блоків:

– Меню.

– Сесії.

– Ресурси.

– ЛМ (локальна мережа) .

– Обрання основного інтерфейсу.

– Шаблонів моніторингу.

– Функціональних кнопок.

На рисунку 5.2 зображено форму авторського права. Було обрано Shareware умову розповсюдження. Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (неповнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми.

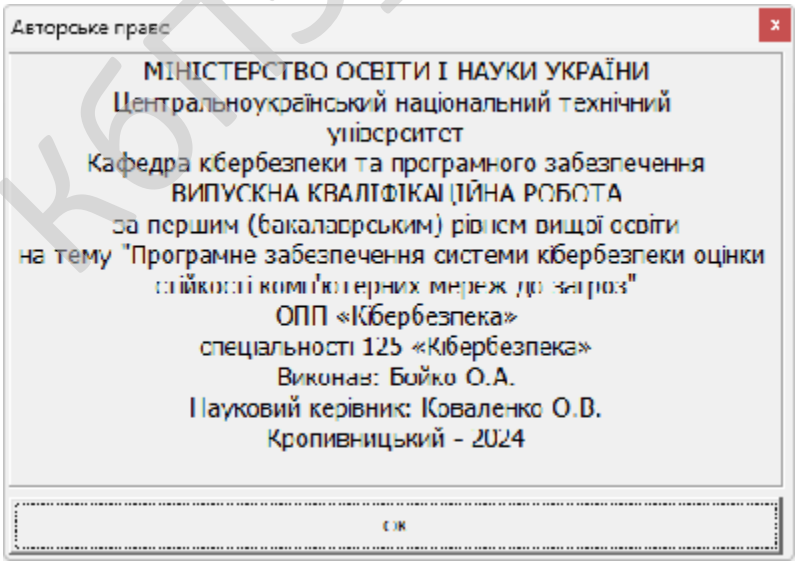

Рисунок 5.2 – Довідка

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки оцінки стійкості комп'ютерних мереж до загроз.

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем оцінки стійкості комп'ютерних мереж до загроз.

– Досліджена система оцінки стійкості комп'ютерних мереж до загроз.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки оцінки стійкості комп'ютерних мереж до загроз.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання оцінки стійкості комп'ютерних мереж до загроз.

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 оцінки стійкості комп'ютерних мереж до загроз. 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 SEED.

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

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

43 Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Кібербезпека: освіта, наука, техніка.* 2024. №3(23), С. 111-131.

44 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.

45 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.

46 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.

47 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.

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

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

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

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз.

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. № 136-02 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 оцінки стійкості комп’ютерних мереж до загроз.

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 оцінки стійкості комп’ютерних мереж до загроз;

– 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 – 70 аркушів.

8 Етапи розробки

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

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

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

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

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

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 7.06.2024 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

_____ Коваленко О.В.

Програмне забезпечення системи кібербезпеки оцінки стійкості комп’ютерних мереж до загроз

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 51

Літера: РП

Кропивницький – 2024 року

ФАЙЛ MONITOR_DEF_LM.DPR

ПРОЕКТ ОСНОВНОЇ ПРОГРАМИ

```
program MONITOR_DEF_LM;
{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
Uses // підключення бібліотек VCL та форм
  Forms,
  MAIN in 'MAIN.pas'{MainForm},
  About in ' About.pas'{Form1},
  TCP_IP in ' TCP_IP.pas'{Form2},
  STATISTIK in 'STATISTIK.pas'{Form3};

{$R *.res} // ресурси

Begin
// початок основного потоку
  Application.Initialize;
// ініціалізація
  Application.CreateForm(TMain, MainForm);
// створення форми
  Application.CreateForm(TForm1, Form1);
// створення форми
  Application.CreateForm(TForm2, Form2);
// створення форми
  Application.CreateForm(TForm3, Form3);
// створення форми
  Application.Run;
// Запуск на виконання
// кінець основного потоку
end.
```

ФАЙЛ STATISTIK.PAS
СТАТИСТИКА МЕРЕЖІ

```
unit STATISTIK; // опис модулю

interface

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, IP, M_API, Buttons;

type
  TForm3 = class(TForm)
    StaticText7: TStaticText;
    TCPStatMemo: TMemo;
    StaticText5: TStaticText;
    IPStatsMemo: TMemo;
    StaticText12: TStaticText;
    ICMPInMemo: TMemo;
    ICMPOutMemo: TMemo;
    StaticText4: TStaticText;
    UDPStatsMemo: TMemo;
    Timer1: TTimer;
    cbTimer: TCheckBox;
    btRTTI: TSpeedButton;
    edtRTTI: TEdit;
    procedure Timer1Timer(Sender: TObject);
    procedure btRTTIClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
     procedure DOIpStuff;
  public
    { Public declarations }
  end;

var
  Form3: TForm3;

implementation

{$R *.dfm}

procedure TForm3.DOIpStuff;
begin

  Get_TCPStatistics( TCPStatMemo.Lines );
  Get_IPStatistics( IPStatsMemo.Lines );
  Get_UDPStatistics( UDPStatsMemo.Lines );
  Get_ICMPStats( ICMPInMemo.Lines, ICMPOutMemo.Lines );
```

```
end;

procedure TForm3.Timer1Timer(Sender: TObject);
begin
  if cbTimer.State = cbCHECKED then
 begin
   Timer1.Enabled := false;
   DoIPStuff;
   Timer1.Enabled := true;
 end;
end;

procedure TForm3.btRTTIClick(Sender: TObject);
var
  IPadr         : dword;
  Rtt, HopCount : longint;
  Res           : integer;
begin
  btRTTI.Enabled := false;
  Screen.Cursor := crHOURGLASS;
  IPadr := Str2IPAddr( edtRTTI.Text );
  Res := Get_RTTAndHopCount( IPadr, 128, RTT, HopCount );
  if Res = NO_ERROR then
    ShowMessage( ' Час запиту '
      + inttostr( rtt ) + ' ms, '
      + inttostr( HopCount )
      + ' hops to : ' + edtRTTI.Text)
  else
    ShowMessage( ' Помилка:'+ #13
                 + ICMPErr2Str( Res ) ) ;
  btRTTI.Enabled := true;
  Screen.Cursor := crDEFAULT;

end;

procedure TForm3.FormCreate(Sender: TObject);
begin

  if LoadIpHlp then
  begin
      DOIpStuff;
      Timer1.Enabled := true;
  end
  else
  ShowMessage( 'Інтернет помічник DLL не є доступним, або не підтримується') ;
end;

end.
```

ФАЙЛ ABOUT.PAS
ДОВІДКА

```
unit About; // опис модулю

interface

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Button1: TButton;
    Image2: TImage;
    Image1: TImage;
    Image3: TImage;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Close;
end;

end.
```

ФАЙЛ MAIN.PAS
ОСНОВНОЇ ПРОГРАМИ

```
unit Main; // опис модулю

interface
{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
// опис бібліотек

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, Stat,
  ShellAPI, ShlObj, ImgList, TCP_IP, About;

// опис типів

type
  TMain = class(TForm)
    cbTypeDisk: TCheckBox;
    cbTypePrint: TCheckBox;
    NetTree: TTreeView;
    ImageList1: TImageList;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    gbxShares: TGroupBox;
    lbxShares: TListBox;
    gbxSessions: TGroupBox;
    btnCloseFile: TButton;
    bvlLeftFiles: TBevel;
    plFiles: TPanel;
    lvFiles: TListView;
    bvlTopFiles: TBevel;
    gbxTraffic: TGroupBox;
    lvTraffic: TListView;
    bvlTraffic: TBevel;
    tmrTraffic: TTimer;
    Button1: TButton;
    rgScope: TRadioGroup;
    lvSessions: TListView;
    bvlSessions: TBevel;
    gbxFiles: TGroupBox;
    btnGetShares: TButton;
    btnCloseShares: TButton;
    btnAddShares: TButton;
    btnCloseSession: TButton;
    btnGetSessions: TButton;
    bvlTopSessions: TBevel;
    plButtonFiles: TPanel;
    btnGetFiles: TButton;
    GroupBox1: TGroupBox;
    cbUsageAll: TCheckBox;
```

```
    cbUsageConnectable: TCheckBox;
    cbUsageContainer: TCheckBox;
    GroupBox2: TGroupBox;
    cbTypeAny: TCheckBox;
    function  IsNT(var Value: Boolean): Boolean;
    procedure btnGetSharesClick(Sender: TObject);
    procedure btnCloseSharesClick(Sender: TObject);
    function  SelectDirectory: String;
    procedure btnAddSharesClick(Sender: TObject);
    function  CardinalToTimeStr(Value:Cardinal):String;
    procedure btnGetSessionsClick(Sender: TObject);
    procedure btnCloseSessionClick(Sender: TObject);
    procedure btnGetFilesClick(Sender: TObject);
    procedure btnCloseFileClick(Sender: TObject);
    procedure tmrTrafficTimer(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure NetTreeCustomDrawItem(Sender: TCustomTreeView;
      Node: TTreeNode; State: TCustomDrawState; var DefaultDraw: Boolean);
    procedure NetTreeDblClick(Sender: TObject);
    procedure NetTreeGetImageIndex(Sender: TObject; Node: TTreeNode);
    procedure Button4Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);

// опис типів та записів

  private
    { Private declarations }
  public
    { Public declarations }
    SessionCloseKey: array [0..512] of SmallInt;
    procedure Open_Do_Close_Enum(const ParentNode: TTreeNode;
    ResScope, ResType, ResUsage: DWORD; const NetContainerToOpen: PNetResource);
    function EnumResources(const ParentNode: TTreeNode;
             ResScope, ResType, ResUsage: DWORD; hNetEnum: THandle): UINT;
  end;

type
 TShareInfo2 = packed record
  shi2_netname : PWChar;
  shi2_type: DWORD;
  shi2_remark :PWChar;
  shi2_permissions: DWORD;
  shi2_max_uses : DWORD;
  shi2_current_uses : DWORD;
  shi2_path : PWChar;
  shi2_passwd : PWChar;
 end;
 PShareInfo2 = ^ TShareInfo2;
 TShareInfo2Array = array [0..512] of TShareInfo2;
 PShareInfo2Array = ^ TShareInfo2Array;

type
 TShareInfo50 = packed record
  shi50_netname : array [0..12] of Char;
  shi50_type : Byte;
  shi50_flags : Word;
  shi50_remark : PChar;
  shi50_path : PChar;
  shi50_rw_password : array [0..8] of Char;
```

```
  shi50_ro_password : array [0..8] of Char;
 end;

type
 TSessionInfo502 = packed record
   Sesi502_cname: PWideChar;
   Sesi502_username: PWideChar;
   Sesi502_num_opens: DWORD;
   Sesi502_time: DWORD;
   Sesi502_idle_time: DWORD;
   Sesi502_user_flags: DWORD;
   Sesi502_cltype_name: PWideChar;
   Sesi502_transport: PWideChar;
 End;
 PSessionInfo502 = ^TSessionInfo502;
 TSessionInfo502Array = array[0..512] of TSessionInfo502;
 PSessionInfo502Array = ^TSessionInfo502Array;

type
 TSessionInfo50 = packed record
   Sesi50_cname       : PChar;
   Sesi50_username    : PChar;
   sesi50_key         : Cardinal;
   sesi50_num_conns   : Word;
   sesi50_num_opens   : Word;
   sesi50_time        : Cardinal;
   sesi50_idle_time   : Cardinal;
   sesi50_protocol    : Byte;
   pad1               : Byte;
 end;

type
 TFileInfo3 = packed record
   fi3_id          : DWORD;
   fi3_permissions : DWORD;
   fi3_num_locks   : DWORD;
   fi3_pathname    : PWChar;
   fi3_username    : PWChar;
 end;
 PFileInfo3 = ^TFileInfo3;
 TFileInfo3Array = array[0..512] of TFileInfo3;
 PFileInfo3Array = ^TFileInfo3Array;

type
 TFileInfo50 = packed record
   fi50_id          : Cardinal;
   fi50_permissions : WORD;
   fi50_num_locks   : WORD;
   fi50_pathname    : PChar;
   fi50_username    : PChar;
   fi50_sharename   : PChar;
 end;

type
  TMibIfRow = packed record
    wszName            : array[0..255] of WideChar;
    dwIndex            : DWORD;
    dwType             : DWORD;
    dwMtu              : DWORD;
    dwSpeed            : DWORD;
```

```
    dwPhysAddrLen      : DWORD;
    bPhysAddr          : array[0..7] of Byte;
    dwAdminStatus      : DWORD;
    dwOperStatus       : DWORD;
    dwLastChange       : DWORD;
    dwInOctets         : DWORD;
    dwInUcastPkts      : DWORD;
    dwInNUCastPkts     : DWORD;
    dwInDiscards       : DWORD;
    dwInErrors         : DWORD;
    dwInUnknownProtos  : DWORD;
    dwOutOctets        : DWORD;
    dwOutUCastPkts     : DWORD;
    dwOutNUCastPkts    : DWORD;
    dwOutDiscards      : DWORD;
    dwOutErrors        : DWORD;
    dwOutQLen          : DWORD;
    dwDescrLen         : DWORD;
    bDescr             : array[0..255] of Char;
  end;
  TMibIfArray = array [0..512] of TMibIfRow;
  PMibIfRow = ^TMibIfRow;
  PMibIfArray = ^TMibIfArray;

type
 TMibIfTable = packed record
   dwNumEntries     : DWORD;
   Table            : TMibIfArray;
 end;
 PMibIfTable = ^TMibIfTable;

var
NetShareEnumNT:function (    servername:PWChar;
                         level:DWORD;
                         bufptr:Pointer;
                         prefmaxlen:DWORD;
                         entriesread,
                         totalentries,
                         resume_handle:LPDWORD): DWORD; stdcall;

var
NetShareEnum:function ( pszServer   : PChar;
                        sLevel           : Cardinal;
                        pbBuffer    : Pchar;
                        cbBuffer    : Cardinal;
                        pcEntriesRead,
                        pcTotalAvail: Pointer):DWORD; stdcall;

var
NetShareDelNT:function (servername: PWideChar;
                        netname: PWideChar;
                        reserved: DWORD): LongInt; stdcall;

var
NetShareDel:function (  pszServer,
                        pszNetName:PChar;
                        usReserved:Word): DWORD; stdcall;

var
NetShareAddNT: function(servername: PWideChar;
```

```
                        level: DWORD;
                        buf: Pointer;
                        parm_err: LPDWORD): DWORD; stdcall;

var
NetShareAdd: function ( pszServer:Pchar;
                        sLevel:Cardinal;
                        pbBuffer:PChar;
                        cbBuffer:Word):DWORD; stdcall;

Var
NetSessionEnumNT:function(servername,
                        UncClientName,
                        username:PWChar;
                        level:DWORD;
                        bufptr:Pointer;
                        prefmaxlen:DWORD;
                        entriesread,
                        totalentries,
                        resume_handle:LPDWORD):DWORD; stdcall;

var
NetSessionEnum:function(pszServer:PChar;
                        sLevel: DWORD;
                        pbBuffer:Pointer;
                        cbBuffer:DWORD;
                        pcEntriesRead,
                        pcTotalAvial:Pointer):integer; stdcall;

var
NetSessionDelNT:function(ServerName,
                        UncClientName,
                        username:PWChar):DWORD; stdcall;

var
NetSessionDel:function( pszServer:PChar;
                        pszClientName: PChar;
                        sReserved: SmallInt):DWORD; stdcall;

var
NetFileEnumNT:function( servername,
                        basepath,
                        username:PWChar;
                        level:DWORD;
                        bufptr:Pointer;
                        prefmaxlen:DWORD;
                        entriesread,
                        totalentries,
                        resume_handle:LPDWORD):DWORD; stdcall;

var
NetFileEnum:function(   pszServer,
                        pszBasePath:PChar;
                        sLevel:DWORD;
                        pbBuffer:Pointer;
                        cbBuffer:DWORD;
                        pcEntriesRead,
                        pcTotalAvail:pointer):integer; stdcall;

var
```

```
NetFileClose:function(  ServerName:PWideChar;
                        FileId:DWORD):DWORD; stdcall;

var
NetFileClose2:function( pszServer:PChar;
                        ulFileId:LongWord):DWORD; stdcall;

var
GetIfTable:function(    pIfTable    : PMibIfTable;
                        pdwSize     : PULONG;
                        bOrder      : Boolean ): DWORD; stdcall;

var
  MainForm: TMain;

implementation

{$R *.dfm}

{ TMain }

//
//  Спочатку нам потрібно визначиться, під якою системою ми працюємо,
// щоб довідатися яку частину коду (для NT чи ні) використовувати в цей момент.
// Для цього напишемо невелику функцію, що і буде визначати тип системи.
//

function TMain.IsNT(var Value: Boolean): Boolean;
var Ver: TOSVersionInfo;
    BRes: Boolean;
begin
  Ver.dwOSVersionInfoSize := SizeOf(TOSVersionInfo);
  BRes := GetVersionEx(Ver);
  if not BRes then   // Перевірка
  begin
    Result := False; //Інформація не отримана
    Exit;            //ідемо
  end else
    Result := True;  //Інформація отримана

  case Ver.dwPlatformId of // визначаємося
    VER_PLATFORM_WIN32_NT       : Value := True;  //Windows NT
    VER_PLATFORM_WIN32_WINDOWS  : Value := False; //Windows 9 x-Me
    VER_PLATFORM_WIN32s         : Result := False //Windows 3.x
  end;
end;

//  Одержання всіх відкритих загальних ресурсів
procedure TMain.btnGetSharesClick(Sender: TObject);
var
  i:Integer;
  FLibHandle : THandle;
  ShareNT : PShareInfo2Array;     //<= Змінні
  entriesread,totalentries:DWORD; //<= для Windows NT
  Share : array [0..512] of TShareInfo50; //<= Змінні
  pcEntriesRead,pcTotalAvail:Word;        //<= для Windows 9 x-Me
  OS: Boolean;
begin
  lbxShares.Items.Clear;
  if not IsNT(OS) then Close; //Визначаємо тип системи
```

```
  if OS then begin  //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL'); //Завантажуємо бібліотеку
    if FLibHandle = 0 then Exit;
    //Зв'язуємо функцію
    @NetShareEnumNT := GetProcAddress(FLibHandle,'NetShareEnum');
    if not Assigned(NetShareEnumNT) then //Перевірка
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    ShareNT := nil; //Очищаємо покажчик на масив структур
    //Виклик функції
    if NetShareEnumNT(nil,2,@ShareNT,DWORD(-1),
                      @entriesread,@totalentries,nil) <> 0 then
    begin  //Якщо виклик невдалий вивантажуємо бібліотеку
      FreeLibrary(FLibHandle);
      Exit;
    end;
    if entriesread > 0 then //Обробка результатів
    for i:= 0 to entriesread- 1 do
      lbxShares.Items.Add(String(ShareNT^[i].shi2_netname));
  end else begin //Код для 9 x-ме
    FLibHandle := LoadLibrary(' SVRAPI.DLL'); //Завантажуємо бібліотеку
    if FLibHandle = 0 then Exit;
    //Зв'язуємо функцію
    @NetShareEnum := GetProcAddress(FLibHandle,'NetShareEnum');
    if not Assigned(NetShareEnum) then //Перевірка
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    //Виклик функції
    if NetShareEnum(nil,50,@Share,SizeOf(Share),
                @pcEntriesRead,@pcTotalAvail)<> 0 then
    begin  //Якщо виклик невдалий вивантажуємо бібліотеку
      FreeLibrary(FLibHandle);
      Exit;
    end;
    if pcEntriesRead > 0 then //Обробка результатів
    for i:= 0 to pcEntriesRead- 1 do
      lbxShares.Items.Add(String(Share[i].shi50_netname));
  end;
  FreeLibrary(FLibHandle); //Не забуваємо вивантажити бібліотеку
end;

//
//  Закриття загального ресурсу
//

procedure TMain.btnCloseSharesClick(Sender: TObject);
var
  OS:Boolean;
  FLibHandle : THandle;
  Name9x:array [0..12] of Char;
  NameNT:PWChar;
  i:Integer;
  ShareName: String;
begin
  if not IsNT(OS) then Close; //Визначаємо тип системи
```

```
  if lbxShares.Items.Count = 0 then Exit;
  for i:= 0 to lbxShares.Items.Count-1 do
    if lbxShares.Selected[i] then Break;  //Шукаємо обраний елемент
  if not lbxShares.Selected[i] then Exit; //Якщо не знайдений ідемо
  ShareName := lbxShares.Items.Strings[i];

  if OS then begin //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetShareDelNT := GetProcAddress(FLibHandle,'NetShareDel');
    if not Assigned(NetShareDelNT) then //Перевірка
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    i:= SizeOf(WideChar)*256;
    GetMem(NameNT,i);  //Виділяємо пам'ять під змінну
    StringToWideChar(ShareName,NameNT,i); //Перетворимо в PWideChar
    NetShareDelNT(nil,NameNT,0);   //Видаляємо ресурс
    FreeMem(NameNT);  //Звільняємо пам'ять
  end else begin  //Код для 9 х-ме
    FLibHandle := LoadLibrary(' SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @NetShareDel := GetProcAddress(FLibHandle,'NetShareDel');
    if not Assigned(NetShareDel) then //Перевірка
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    FillChar(Name9x, SizeOf(Name9x), #0); //Очищаємо масив
    move(ShareName[1],Name9x[0],Length(ShareName)); //Заповнюємо масив
    NetShareDel(nil,@Name9x,0); //Видаляємо ресурс
  end;
  FreeLibrary(FLibHandle);
end;

//
//  Показу діалогу вибору директорії
//

function TMain.SelectDirectory: String;
var
  lpItemID : PItemIDList;
  BrowseInfo : TBrowseInfo;
  DisplayName : array[0..MAX_PATH] of Char;
  TempPath : array[0..MAX_PATH] of Char;
begin
  FillChar(BrowseInfo, sizeof(TBrowseInfo), #0);
  BrowseInfo.hwndOwner := Handle;
  BrowseInfo.pszDisplayName := @DisplayName;
  BrowseInfo.lpszTitle := ' Specify a directory';
  BrowseInfo.ulFlags := BIF_RETURNONLYFSDIRS;
  lpItemID := SHBrowseForFolder(BrowseInfo);
  if Assigned(lpItemId) then begin
    SHGetPathFromIDList(lpItemID, TempPath);
    GlobalFreePtr(lpItemID);
  end else Result := ' ' ;
  Result := String(TempPath);
end;
```

```
//
//  Додавання загального ресурсу
//

procedure TMain.btnAddSharesClick(Sender: TObject);
const
  STYPE_DISKTREE = 0;
  ACCESS_ALL =  258;
  SHI50F_FULL = 258;
var
  FLibHandle : THandle;
  Share9x : TShareInfo50;
  ShareNT : TShareInfo2;
  TmpDir, TmpName: String;
  TmpDirNT, TmpNameNT: PWChar;
  OS: Boolean;
  TmpLength: Integer;
begin
  TmpDir := SelectDirectory; //Визначаємо шлях до наступного ресурсу
  TmpName := InputBox(' Share name','Enter name','Test'); //Визначаємо ім'я під
яким він буде видний у мережі
  if TmpDir = ' '  then Exit;

  if not IsNT(OS) then Close; //З'ясовуємо тип системи

  if OS then begin //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetShareAddNT := GetProcAddress(FLibHandle,'NetShareAdd');
    if not Assigned(NetShareAddNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    TmpLength := SizeOF(WideChar)*256; //Визначаємо необхідний розмір

    GetMem(TmpNameNT, TmpLength); //Конвертуємо в PWChar
    StringToWideChar(TmpName, TmpNameNT, TmpLength);
    ShareNT.shi2_netname := TmpNameNT; //Ім'я

    ShareNT.shi2_type := STYPE_DISKTREE; //Тип ресурсу
    ShareNT.shi2_remark := ' ' ; //Коментар
    ShareNT.shi2_permissions := ACCESS_ALL; //Доступ
    ShareNT.shi2_max_uses := DWORD(-1); // Кіл-У максим. підключ.
    ShareNT.shi2_current_uses := 0; // Кіл-У тік підкл.

    GetMem(TmpDirNT, TmpLength);
    StringToWideChar(TmpDir, TmpDirNT, TmpLength);
    ShareNT.shi2_path := TmpDirNT; //Шлях до ресурсу

    ShareNT.shi2_passwd := ' ' ; //Пароль

    NetShareAddNT(nil,2,@ShareNT, nil); //Додаємо ресурс
    FreeMem (TmpNameNT); //звільняємо пам'ять
    FreeMem (TmpDirNT);
  end else begin
    FLibHandle := LoadLibrary(' SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @NetShareAdd := GetProcAddress(FLibHandle,'NetShareAdd');
    if not Assigned(NetShareAdd) then
```

```
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    FillChar(Share9x.shi50_netname, SizeOf(Share9x.shi50_netname), #0);
    move(TmpName[1],Share9x.shi50_netname[0],Length(TmpName)); //Ім'я
    Share9x.shi50_type := STYPE_DISKTREE; //Тип ресурсу
    Share9x.shi50_flags := SHI50F_FULL; //Доступ
    FillChar(Share9x.shi50_remark,
      SizeOf(Share9x.shi50_remark), #0); //Коментар
    FillChar(Share9x.shi50_path,
      SizeOf(Share9x.shi50_path), #0);
    Share9x.shi50_path := PAnsiChar(TmpDir); //Шлях до ресурсу
    FillChar(Share9x.shi50_rw_password,
      SizeOf(Share9x.shi50_rw_password), #0); //Пароль повного доступу
    FillChar(Share9x.shi50_ro_password,
      SizeOf(Share9x.shi50_ro_password), #0); //Пароль для читання
    NetShareAdd(nil,50,@Share9x,SizeOf(Share9x));
  end;
  FreeLibrary(FLibHandle);
end;

//
//  Помітьте що активний і неактивний час сесій буде даватися нам
//  у вигляді кіл-ті секунд (тип Cardinal). Напишемо невелику
//  функцію, задача якої буде перетворювати кіл-у секунд у більше
//  звичну форму відображення.
//

function TMain.CardinalToTimeStr(Value: Cardinal): String;
var d,h,m,s: Real;
begin
    d:=0;
    h:=0;
    m:=0;
    s:=Value;
    if s > 59 then begin
        m:=int(s / 60);
        s:= s-s-(m*60);
    end;
    if m > 59 then begin
        h:=int(m/60);
        m:= m-m-(h*60);
    end;
    if h > 23 then begin
        d:=int(h/24);
        h:= h-h-(d*24);
    end;
    Result:=' ' ;
    if (d>0) then Result:=Result+floattostr(d)+' d. ' ;
    if (h<9) then Result:=Result+'0'+floattostr(h)+':' else
Result:=Result+floattostr(h)+':';
    if (m<9) then Result:=Result+'0'+floattostr(m)+':' else
Result:=Result+floattostr(m)+':';
    if (s<9) then Result:=Result+'0'+floattostr(s)     else
Result:=Result+floattostr(s);
end;

//
//  Одержання списку сесій
```

```
procedure TMain.btnGetSessionsClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  SessionInfo50: array [0..512] of TSessionInfo50;
  SessionInfo502 : PSessionInfo502Array;
  TotalEntries,EntriesReadNT: DWORD;
  EntriesRead,TotalAvial: Word;
  i:integer;
begin
  lvSessions.Items.Clear;

  if not IsNT(OS) then Close; //З'ясовуємо тип системи

  if OS then begin  //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetSessionEnumNT := GetProcAddress(FLibHandle, ' NetSessionEnum');
    if not Assigned(NetSessionEnumNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    SessionInfo502 := nil;
    if NetSessionEnumNT(nil,nil,nil,502,@SessionInfo502,DWORD(-
1),@entriesreadNT, @totalentries, nil)=0 then
    for i:=0 to EntriesReadNT-1 do
    begin
      with lvSessions.Items.Add do  //Заповнення даними зі структури
      begin
        Caption := string(SessionInfo502^[i].sesi502_cname); //Ім'я комп'ютера
        SubItems.Add(SessionInfo502^[i].sesi502_username);   //Ім'я користувача
        SubItems.Add(IntToStr(SessionInfo502^[i].sesi502_num_opens));
//Відкритих ресурсів
        SubItems.Add(CardinalToTimeStr(SessionInfo502^[i].Sesi502_Time));
//Час активний
        SubItems.Add(CardinalToTimeStr(SessionInfo502^[i].sesi502_idle_time));
//Час не активний
      end;
    end;
  end else begin //Код для Windows 9 x-Me
   FLibHandle := LoadLibrary(' SVRAPI.DLL');
   if FLibHandle = 0 then Exit;
   @NetSessionEnum := GetProcAddress(FLibHandle, ' NetSessionEnum');
    if not Assigned(NetSessionEnum) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
   if NetSessionEnum
(nil,50,@SessionInfo50,SizeOf(SessionInfo50),@EntriesRead,@TotalAvial) = 0 then
   for i:=0 to EntriesRead-1 do
   begin
      with lvSessions.Items.Add do  //Заповнення даними зі структури
      begin
        Caption := string(SessionInfo50[i].Sesi50_cname); //Ім'я комп'ютера
        SubItems.Add(SessionInfo50[i].Sesi50_username);   //Ім'я користувача
        SubItems.Add(IntToStr(SessionInfo50[i].sesi50_num_opens));
//Відкритих ресурсів
        SubItems.Add(CardinalToTimeStr(SessionInfo50[i].Sesi50_Time));
```

```
//Час активний
        SubItems.Add(CardinalToTimeStr(SessionInfo50[i].sesi50_idle_time));
//Час не активний
        SessionCloseKey[i]:= SessionInfo50[i].sesi50_key;
// Унікальний ідентифікатор для закриття
      end;
   end;
 end;
 FreeLibrary(FLibHandle);
end;

//
//  Завершення обраної сесії
//

procedure TMain.btnCloseSessionClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  CNameNT: PWideChar;
  CName9x: PAnsiChar;
  Key:SmallInt;
  i: Integer;
begin
  if not IsNT(OS) then Close; //З'ясовуємо тип системи

  if not Assigned(lvSessions.Selected) then Exit;
  i:= lvSessions.Selected.Index; //Визначаємо номер обраної сесії

  if OS then begin
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetSessionDelNT := GetProcAddress(FLibHandle, ' NetSessionDel');
    if not Assigned(NetSessionDelNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
// Перетворимо дані в необхідний вид
    CNameNT := PWChar(WideString(' \\'+lvSessions.Items.Item[i].Caption));
    NetSessionDelNT(nil,CNameNT,nil);
  end else begin
    FLibHandle := LoadLibrary(' SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @NetSessionDel := GetProcAddress(FLibHandle, ' NetSessionDel');
    if not Assigned(NetSessionDel) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
// Перетворимо дані в необхідний вид
    CName9x := PAnsiChar(lvSessions.Items.Item[i].Caption);
    key := SessionCloseKey[i];
// Беремо ключ із масиву
    NetSessionDel(nil,CName9x,Key);
  end;
  FreeLibrary(FLibHandle);
end;

//
```

```
//  Одержання списку відкритих файлів

procedure TMain.btnGetFilesClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  FileInfoNT: PFileInfo3Array;
  FileInfo9x: array [0..512] of TFileInfo50;
  TotalEntries,EntriesReadNT: DWORD;
  EntriesRead,TotalAvial: Word;
  i:integer;
begin
  lvfiles.Items.Clear;

  if not IsNT(OS) then Close; //З'ясовуємо тип системи

  if OS then begin  //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileEnumNT := GetProcAddress(FLibHandle, ' NetFileEnum');
    if not Assigned(NetFileEnumNT) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
    FileInfoNT := nil;
    if NetFileEnumNT(nil,nil,nil,3,@FileInfoNT,DWORD(-1),@entriesreadNT,
@totalentries, nil)=0 then
    for i:=0 to EntriesReadNT-1 do
    begin
      with lvFiles.Items.Add do
// Заповнення даними зі структури
      begin
        Caption := string(IntToStr(FileInfoNT^[i].fi3_id));
// Ідентифікатор
        SubItems.Add(FileInfoNT^[i].fi3_pathname);   //Шлях до файлу
        SubItems.Add(FileInfoNT^[i].fi3_username);   //Ім'я користувача
      end;
    end;
  end else begin //Код для Windows 9 x-Me
   FLibHandle := LoadLibrary(' SVRAPI.DLL');
   if FLibHandle = 0 then Exit;
   @NetFileEnum := GetProcAddress(FLibHandle, ' NetFileEnum');
    if not Assigned(NetFileEnum) then
    begin
      FreeLibrary(FLibHandle);
      Exit;
    end;
   if NetFileEnum (nil,
nil,50,@FileInfo9x,SizeOf(FileInfo9x),@EntriesRead,@TotalAvial)= 0 then
   for i:=0 to EntriesRead-1 do
   begin
      with lvFiles.Items.Add do
// Заповнення даними зі структури
      begin
        Caption := string(IntToStr(FileInfo9x[i].fi50_id)); //Ідентифікатор
        SubItems.Add(FileInfo9x[i].fi50_pathname); //Шлях до файлу
        SubItems.Add(FileInfo9x[i].fi50_username); //Ім'я користувача
      end;
   end;
```

```
 end;
 FreeLibrary(FLibHandle);
end;
//  Закриття файлу
//

procedure TMain.btnCloseFileClick(Sender: TObject);
var
  OS: Boolean;
  FLibHandle : THandle;
  i: Integer;
begin
  if not IsNT(OS) then Close; //З'ясовуємо тип системи

  if not Assigned(lvFiles.Selected) then Exit;
  i:= lvFiles.Selected.Index; //Визначаємо номер обраного файлу

  if OS then begin //Код для NT
    FLibHandle := LoadLibrary(' NETAPI32.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileClose := GetProcAddress(FLibHandle, ' NetFileClose');
    if not Assigned(NetFileClose) then
    begin
      FreeLibrary(FLibHandle);
      Close;
    end;
    NetFileClose(nil,StrToInt(lvFiles.Items.Item[i].Caption));//Закриваємо файл
  end else begin //Код для Windows 9 x-Me
    FLibHandle := LoadLibrary(' SVRAPI.DLL');
    if FLibHandle = 0 then Exit;
    @NetFileClose2 := GetProcAddress(FLibHandle, ' NetFileClose2');
    if not Assigned(NetFileClose2) then
    begin
      FreeLibrary(FLibHandle);
      Close;
    end;
    NetFileClose2(nil,StrToInt(lvFiles.Items.Item[i].Caption));
  end;
  FreeLibrary(FLibHandle);
end;

//  Визначаємо вхідний- вихідний трафік

procedure TMain.tmrTrafficTimer(Sender: TObject);
  // Допоміжна функція, що перетворить MAC адресу до "нормального" виду
  //Визначаємо спеціальний тип, щоб можна було передати у функцію масив
  type TMAC = array [0..7] of Byte;
  //Як перше значення масив, друге значення, розмір даних у масиві
  function GetMAC(Value: TMAC; Length: DWORD): String;
  var
    i: Integer;
  begin
    if Length = 0 then Result := '  00-00-00' else
    begin
      Result := ' ' ;
      for i:= 0 to Length-2 do
        Result := Result + IntToHex(Value[i],2)+'-';
      Result := Result + IntToHex(Value[ Length-1],2);
    end;
  end;
```

```
// Сама процедура
var
  FLibHandle : THandle;
  Table: TMibIfTable;
  i : integer;
  Size   : integer;
begin
  tmrTraffic.Enabled := false; //Припиняємо про всякий випадок таймер
  lvTraffic.Items.BeginUpdate;
  lvTraffic.Items.Clear;  //Очищаємо список
  FLibHandle := LoadLibrary(' M_API.DLL'); //Завантажуємо бібліотеку
  if FLibHandle = 0 then Exit;
  @GetIfTable := GetProcAddress(FLibHandle, ' GetIfTable');
  if not Assigned(GetIfTable) then
  begin
    FreeLibrary(FLibHandle);
    Close;
  end;

  Size := SizeOf(Table);
  if GetIfTable(@Table, @Size, false ) = 0 then //Виконуємо функцію
    for i:= 0 to Table.dwNumEntries-1 do begin
      with lvTraffic.Items.Add do begin //Виводимо результати
        Caption := String(Table.Table[i].bDescr); //Найменування інтерфейсу
        SubItems.Add(GetMAC(TMAC(Table.Table[i].bPhysAddr),
          Table.Table[i].dwPhysAddrLen)); //MAC адреса
        SubItems.Add(IntToStr(Table.Table[i].dwInOctets));
// Усього прийнято байт
        SubItems.Add(IntToStr(Table.Table[i].dwOutOctets));
// Усього відправлено байт
      end;
    end;
  lvTraffic.Items.EndUpdate;
  FreeLibrary(FLibHandle);
  tmrTraffic.Enabled := true;
// Не забуваємо активувати таймер
end;

function OpenEnum(const NetContainerToOpen: PNetResource; ResScope,
                  ResType, ResUsage: DWORD): THandle;
var
 hNetEnum: THandle;
begin
 Result:=0;
if (NO_ERROR<>WNetOpenEnum(ResScope, ResType, ResUsage,
                           NetContainerToOpen, hNetEnum))
 then ShowMessage(' Помилка!')
 else Result:=hNetEnum;
end;

function EnumResources(const ParentNode: TTreeNode;
ResScope, ResType, ResUsage: DWORD; hNetEnum: THandle): UINT;

function ShowResource(const ParentNode: TTreeNode; Res: TNetResource):
TTreeNode;
  begin
   Result:=MainForm.NetTree.Items.AddChild(ParentNode,
string(Res.lpRemoteName));
  end;
```

```
const
 RESOURCE_BUF_ENTRIES = 2000;

var
 ResourceBuffer: array[1..RESOURCE_BUF_ENTRIES] of TNetResource;
 i, ResourceBuf, EntriesToGet: dword;
 NewNode: TTreeNode;
begin
 Result:=0;
 while true do
  begin
   ResourceBuf:=sizeof(ResourceBuffer);
   EntriesToGet:=RESOURCE_BUF_ENTRIES;
   if (NO_ERROR<>WNetEnumResource(hNetEnum, EntriesToGet,
                                  @ResourceBuffer, ResourceBuf))
   then
    begin
     case GetLastError() of
      NO_ERROR:
// проход буферу без перемикання
       Break;
      ERROR_NO_MORE_ITEMS:
// Повертає о у тому випадку, коли останов
// RESOURCE_BUF_ENTRIES данні на попередньому виклику, щоб
// WNetEnumResource, та були точно
// RESOURCE_BUF_ENTRIES данні в запису на момент
// попереднього виклику
       Exit;
      else ShowMessage(Помилка!');
       Result:=1;
       Exit;
     end;
    end;
   for i:=1 to EntriesToGet do
    begin
     NewNode:=ShowResource(ParentNode, ResourceBuffer[i]);
     if (ResourceBuffer[i].dwUsage and RESOURCEUSAGE_CONTAINER)<>0
     then MainForm.Open_Do_Close_Enum(NewNode, ResScope, ResType, ResUsage,
@ResourceBuffer[i]);
     Application.ProcessMessages;
    end;
  end;
end;

procedure TMain.Open_Do_Close_Enum(const ParentNode: TTreeNode; ResScope,
ResType, ResUsage: DWORD; const NetContainerToOpen: PNetResource);
var
 hNetEnum: THandle;
begin
 hNetEnum:=OpenEnum(NetContainerToOpen, ResScope, ResType, ResUsage);
 if (hNetEnum=0)
 then Exit;
 EnumResources(ParentNode, ResScope, ResType, ResUsage, hNetEnum);
 if (NO_ERROR<>WNetCloseEnum(hNetEnum))
 then ShowMessage(' WNetCloseEnum Помилка');
end;

procedure TMain.Button1Click(Sender: TObject);
var
 ResScope, ResType, ResUsage: dword;
```

```
begin
 Button1.Caption:=‘ Пошук мережних ресурсів. Чекайте...’;
 Button1.Enabled:=false;
 //
 NetTree.Items.Clear;
 case rgScope.ItemIndex of
  1: ResScope:=RESOURCE_GLOBALNET;
  2: ResScope:=RESOURCE_REMEMBERED;
  else ResScope:=RESOURCE_CONNECTED;
 end;
 ResType:=0;
 if cbTypeAny.Checked
 then ResType:=ResType or RESOURCETYPE_ANY;
 if cbTypeDisk.Checked
 then ResType:=ResType or RESOURCETYPE_DISK;
 if cbTypePrint.Checked
 then ResType:=ResType or RESOURCETYPE_PRINT;
 ResUsage:=0;
 if cbUsageConnectable.Checked
 then ResUsage:=ResUsage or RESOURCEUSAGE_CONNECTABLE;
 if cbUsageContainer.Checked
 then ResUsage:=ResUsage or RESOURCEUSAGE_CONTAINER;
 Open_Do_Close_Enum(NetTree.Items.Add(nil, ‘ Network Resources’),
                              ResScope, ResType, ResUsage, nil);
Button1.Caption:=‘ Обновити список ресурсів’;
 Button1.Enabled:=true;
end;
procedure TMain.NetTreeCustomDrawItem(Sender: TCustomTreeView;
  Node: TTreeNode; State: TCustomDrawState; var DefaultDraw: Boolean);
begin
if cdsSelected in State
 then Sender.Canvas.Font.Style:=Sender.Canvas.Font.Style+[fsUnderline];
end;

procedure TMain.NetTreeDblClick(Sender: TObject);
begin
ShellExecute(0,’open’,PChar(NetTree.Selected.Text),’‘ ,’‘ ,SW_SHOW);
end;
procedure TMain.NetTreeGetImageIndex(Sender: TObject; Node: TTreeNode);
begin
 if Node.HasChildren
 then Node.ImageIndex:=1
 else Node.ImageIndex:=0;
end;

procedure TMain.Button4Click(Sender: TObject);
begin
   Form1.Show;
end;
procedure TMain.Button2Click(Sender: TObject);
begin
    Form2.Show;
end;
procedure TMain.Button3Click(Sender: TObject);
begin
Form3.Show;
end;
end.
```

ФАЙЛ M_API.PAS - ОБРОБКА API ФУНКЦІЙ

```
unit M_API; // опис модулю

interface

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
uses
  Windows, winsock;

const
  VERSION ='3.1'; // поточна версія

// Заголовок з Microsoft IPTYPES.H

const
  ANY_SIZE      = 1;
  MAX_ADAPTER_DESCRIPTION_LENGTH = 128; // arb.
  MAX_ADAPTER_NAME_LENGTH = 256; // змінна
  MAX_ADAPTER_ADDRESS_LENGTH = 8; // змінна
  DEFAULT_MINIMUM_ENTITIES = 32; // змінна
  MAX_HOSTNAME_LEN = 128; // змінна
  MAX_DOMAIN_NAME_LEN = 128; // змінна
  MAX_SCOPE_ID_LEN = 256; // змінна

 // Вузлові типи ( NETBIOS)
  BROADCAST_NODETYPE = 1;
  PEER_TO_PEER_NODETYPE = 2;
  MIXED_NODETYPE = 4;
  HYBRID_NODETYPE = 8;

NETBIOSTypes  : array[0..8] of string[20]=(' Невизначений',
'Передача','Рівень до рівня',' ',' Змішаний','' ,'', '','Гібрід');

 // Типи адаптеру
{ v1.4-> 1.5
  IF_OTHER_ADAPTERTYPE = 0;
  IF_ETHERNET_ADAPTERTYPE = 1;
  IF_TOKEN_RING_ADAPTERTYPE = 2;
  IF_FDDI_ADAPTERTYPE = 3;
  IF_PPP_ADAPTERTYPE = 4;
  IF_LOOPBACK_ADAPTERTYPE = 5;
  IF_SLIP_ADAPTERTYPE = 6;

 found in ipifcons.h :
#define MIB_IF_TYPE_OTHER               1
#define MIB_IF_TYPE_ETHERNET            6
#define MIB_IF_TYPE_TOKENRING           9
#define MIB_IF_TYPE_FDDI                15
#define MIB_IF_TYPE_PPP                 23
#define MIB_IF_TYPE_LOOPBACK            24
#define MIB_IF_TYPE_SLIP                28
```

```
}
  IF_OTHER_ADAPTERTYPE = 1;
  IF_ETHERNET_ADAPTERTYPE = 6;
  IF_TOKEN_RING_ADAPTERTYPE = 9;
  IF_FDDI_ADAPTERTYPE = 15;
  IF_PPP_ADAPTERTYPE = 23;
  IF_LOOPBACK_ADAPTERTYPE = 24;
  IF_SLIP_ADAPTERTYPE = 28;

AdaptTypes    : array[0..6] of string[10] = ( ' інший', ' ethernet', '
tokenring', ' FDDI', ' PPP', ' loopback', ' SLIP' );

AdaptTypes:array[1..28] of string[10] = ( ' інший', ' ' , ' ' , ' ' , ' ' , '
ethernet', ' ' , ' ' , ' tokenring',     ' ' , ' ' , ' ' , ' ' , ' ' , ' FDDI',
' ' , ' ' , ' ' , ' ' , ' ' , ' ' , ' ' , ' PPP',     ' loopback', ' ' , ' ' , '
' , ' SLIP' );
// Кінець змін в типі адаптерів
//-------для інших MS заготовочних файлів-------------------
  MAX_INTERFACE_NAME_LEN = 256; { mrapi.h }
  MAXLEN_PHYSADDR = 8; { iprtrmib.h }
  MAXLEN_IFDESCR = 256; {-"--     }
//-----------------------------------
type
  TMacAddress = array[1..MAX_ADAPTER_ADDRESS_LENGTH] of byte;
//---IP адресні структури

  PTIP_ADDRESS_STRING = ^TIP_ADDRESS_STRING;
  TIP_ADDRESS_STRING = array[0..15] of char;
//  IP рядок
  PTIP_ADDR_STRING = ^TIP_ADDR_STRING;
  TIP_ADDR_STRING = packed record
// для використання у зв'язних списках
    Next: PTIP_ADDR_STRING;
    IpAddress: TIP_ADDRESS_STRING;
    IpMask: TIP_ADDRESS_STRING;
    Context: DWORD;
  end;

//-----Fixed Info структура

  PTFixedInfo = ^TFixedInfo;
  TFixedInfo = packed record
    HostName: array[1..MAX_HOSTNAME_LEN + 4] of char;
// данні
    DomainName: array[1..MAX_DOMAIN_NAME_LEN + 4] of char;
// данні
    CurrentDNSServer: PTIP_ADDR_STRING;
    DNSServerList: TIP_ADDR_STRING;
    NodeType: UINT;
    ScopeID: array[1..MAX_SCOPE_ID_LEN + 4] of char;
// данні
    EnableRouting: UINT;
    EnableProxy: UINT;
    EnableDNS: UINT;
  end;

// структура мережного інтерфейсу
{
Наступне є діючими станами для WAN да LAN інтерфейсів. Порядок станів створений
для визначення. Для стану >= CONNECTED можливо передавати данні зразу. Стан >=
```

```
DISCONNECTED може передавати деякі данні. Стан < DISCONNECTED може не передавати
дані. карта з поміткою UNREACHABLE якщо DIM викликає InterfaceUnreachable для
причин. Крім невдачі з'єднання. NON_OPERATIONAL- Перевірка для LAN інтерфейсу.
Позначає карту що не працює або не з'єднується з картою. UNREACHABLE- Перевірка
WAN інтерфейсів.
}
const
// данні додані до ipifcons.h
  IF_OPER_STATUS_NON_OPERATIONAL = 0 ;
  IF_OPER_STATUS_UNREACHABLE = 1 ;
  IF_OPER_STATUS_DISCONNECTED = 2 ;
  IF_OPER_STATUS_CONNECTING = 3 ;
  IF_OPER_STATUS_CONNECTED = 4 ;
  IF_OPER_STATUS_OPERATIONAL = 5 ;
  MIB_IF_TYPE_OTHER = 1 ;
  MIB_IF_TYPE_ETHERNET = 6 ;
  MIB_IF_TYPE_TOKENRING = 9 ;
  MIB_IF_TYPE_FDDI = 15 ;
  MIB_IF_TYPE_PPP = 23 ;
  MIB_IF_TYPE_LOOPBACK = 24 ;
  MIB_IF_TYPE_SLIP = 28 ;
  MIB_IF_ADMIN_STATUS_UP = 1 ;
  MIB_IF_ADMIN_STATUS_DOWN = 2 ;
  MIB_IF_ADMIN_STATUS_TESTING = 3 ;
  MIB_IF_OPER_STATUS_NON_OPERATIONAL = 0 ;
  MIB_IF_OPER_STATUS_UNREACHABLE = 1 ;
  MIB_IF_OPER_STATUS_DISCONNECTED = 2 ;
  MIB_IF_OPER_STATUS_CONNECTING = 3 ;
  MIB_IF_OPER_STATUS_CONNECTED = 4 ;
  MIB_IF_OPER_STATUS_OPERATIONAL = 5 ;

type
  PTMibIfRow = ^TMibIfRow;
  TMibIfRow = packed record
    wszName: array[1..MAX_INTERFACE_NAME_LEN] of WCHAR;
    dwIndex: DWORD;
    dwType: DWORD;      // дивись MIB_IF_TYPE
    dwMTU: DWORD;
    dwSpeed: DWORD;
    dwPhysAddrLen: DWORD;
    bPhysAddr: array[1..MAXLEN_PHYSADDR] of byte;
    dwAdminStatus: DWORD;    // дивись MIB_IF_ADMIN_STATUS
    dwOperStatus: DWORD;     // дивись MIB_IF_OPER_STATUS
    dwLastChange: DWORD;
    dwInOctets: DWORD;
    dwInUcastPkts: DWORD;
    dwInNUCastPkts: DWORD;
    dwInDiscards: DWORD;
    dwInErrors: DWORD;
    dwInUnknownProtos: DWORD;
    dwOutOctets: DWORD;
    dwOutUCastPkts: DWORD;
    dwOutNUCastPkts: DWORD;
    dwOutDiscards: DWORD;
    dwOutErrors: DWORD;
    dwOutQLen: DWORD;
    dwDescrLen: DWORD;
    bDescr: array[1..MAXLEN_IFDESCR] of char; //byte;
  end;
```

```
 //
  PTMibIfTable = ^TMIBIfTable;
  TMibIfTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..ANY_SIZE- 1] of TMibIfRow;
  end;

//---ADAPTER INFO структура

  PTIP_ADAPTER_INFO = ^TIP_ADAPTER_INFO;
  TIP_ADAPTER_INFO = packed record
    Next: PTIP_ADAPTER_INFO;
    ComboIndex: DWORD;
    AdapterName: array[1..MAX_ADAPTER_NAME_LENGTH + 4] of char;
// данні
    Description: array[1..MAX_ADAPTER_DESCRIPTION_LENGTH + 4] of char;
// данні
    AddressLength: UINT;
    Address: array[1..MAX_ADAPTER_ADDRESS_LENGTH] of byte;
// данні
    Index: DWORD;
    aType: UINT;
    DHCPEnabled: UINT;
    CurrentIPAddress: PTIP_ADDR_STRING;
    IPAddressList: TIP_ADDR_STRING;
    GatewayList: TIP_ADDR_STRING;
    DHCPServer: TIP_ADDR_STRING;
    HaveWINS: BOOL;
    PrimaryWINSServer: TIP_ADDR_STRING;
    SecondaryWINSServer: TIP_ADDR_STRING;
    LeaseObtained: LongInt;
// UNIX час, секунди з 1970
    LeaseExpires: LongInt;
// UNIX час, секунди з 1970
    SpareStuff: array [1..200] of char ;
// данні- простір для списку IP адрес
  end;

//--------TCP структура---------------------

  PTMibTCPRow = ^TMibTCPRow;
  TMibTCPRow = packed record
    dwState: DWORD;
    dwLocalAddr: DWORD;
    dwLocalPort: DWORD;
    dwRemoteAddr: DWORD;
    dwRemotePort: DWORD;
  end;
  //
  PTMibTCPTable = ^TMibTCPTable;
  TMibTCPTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..0] of TMibTCPRow;
  end;
  //
  PTMibTCPStats = ^TMibTCPStats;
  TMibTCPStats = packed record
    dwRTOAlgorithm: DWORD;
    dwRTOMin: DWORD;
    dwRTOMax: DWORD;
```

```
    dwMaxConn: DWORD;
    dwActiveOpens: DWORD;
    dwPassiveOpens: DWORD;
    dwAttemptFails: DWORD;
    dwEstabResets: DWORD;
    dwCurrEstab: DWORD;
    dwInSegs: DWORD;
    dwOutSegs: DWORD;
    dwRetransSegs: DWORD;
    dwInErrs: DWORD;
    dwOutRsts: DWORD;
    dwNumConns: DWORD;
  end;

//-----UDP СТРУКТУРА -------------------------

  PTMibUDPRow = ^TMibUDPRow;
  TMibUDPRow = packed record
    dwLocalAddr: DWORD;
    dwLocalPort: DWORD;
  end;
 //
  PTMibUDPTable = ^TMIBUDPTable;
  TMIBUDPTable = packed record
    dwNumEntries: DWORD;
    UDPTable: array[0..ANY_SIZE- 1] of TMibUDPRow;
  end;
 //
  PTMibUdpStats = ^TMIBUdpStats;
  TMIBUdpStats = packed record
    dwInDatagrams: DWORD;
    dwNoPorts: DWORD;
    dwInErrors: DWORD;
    dwOutDatagrams: DWORD;
    dwNumAddrs: DWORD;
  end;

//------IP СТРУКТУРА ------------------------

 //
  PTMibIPNetRow = ^TMibIPNetRow;
  TMibIPNetRow = packed record
    dwIndex: DWord;
    dwPhysAddrLen: DWord;
    bPhysAddr: TMACAddress;
    dwAddr: DWord;
    dwType: DWord;
  end;
  //
  PTMibIPNetTable = ^TMibIPNetTable;
  TMibIPNetTable = packed record
    dwNumEntries: DWORD;
    Table: array[0..ANY_SIZE- 1] of TMibIPNetRow;
  end;
  //
  PTMibIPStats = ^TMibIPStats;
  TMibIPStats = packed record
    dwForwarding: DWORD;
    dwDefaultTTL: DWORD;
    dwInReceives: DWORD;
```

```
  dwInHdrErrors: DWORD;
  dwInAddrErrors: DWORD;
  dwForwDatagrams: DWORD;
  dwInUnknownProtos: DWORD;
  dwInDiscards: DWORD;
  dwInDelivers: DWORD;
  dwOutRequests: DWORD;
  dwRoutingDiscards: DWORD;
  dwOutDiscards: DWORD;
  dwOutNoRoutes: DWORD;
  dwReasmTimeOut: DWORD;
  dwReasmReqds: DWORD;
  dwReasmOKs: DWORD;
  dwReasmFails: DWORD;
  dwFragOKs: DWORD;
  dwFragFails: DWORD;
  dwFragCreates: DWORD;
  dwNumIf: DWORD;
  dwNumAddr: DWORD;
  dwNumRoutes: DWORD;
end;
//
PTMibIPAddrRow = ^TMibIPAddrRow;
TMibIPAddrRow = packed record
  dwAddr: DWORD;
  dwIndex: DWORD;
  dwMask: DWORD;
  dwBCastAddr: DWORD;
  dwReasmSize: DWORD;
  Unused1,
  Unused2: WORD;
end;
//
PTMibIPAddrTable = ^TMibIPAddrTable;
TMibIPAddrTable = packed record
  dwNumEntries: DWORD;
  Table: array[0..ANY_SIZE- 1] of TMibIPAddrRow;
end;

//
PTMibIPForwardRow = ^TMibIPForwardRow;
TMibIPForwardRow = packed record
  dwForwardDest: DWORD;
  dwForwardMask: DWORD;
  dwForwardPolicy: DWORD;
  dwForwardNextHop: DWORD;
  dwForwardIFIndex: DWORD;
  dwForwardType: DWORD;
  dwForwardProto: DWORD;
  dwForwardAge: DWORD;
  dwForwardNextHopAS: DWORD;
  dwForwardMetric1: DWORD;
  dwForwardMetric2: DWORD;
  dwForwardMetric3: DWORD;
  dwForwardMetric4: DWORD;
  dwForwardMetric5: DWORD;
end;
//
PTMibIPForwardTable = ^TMibIPForwardTable;
TMibIPForwardTable = packed record
```

```
    dwNumEntries: DWORD;
    Table: array[0..ANY_SIZE- 1] of TMibIPForwardRow;
  end;

//---- ICMP-СТРУКТУРА ------------------------

  PTMibICMPStats = ^TMibICMPStats;
  TMibICMPStats = packed record
    dwMsgs: DWORD;
    dwErrors: DWORD;
    dwDestUnreachs: DWORD;
    dwTimeEcxcds: DWORD;
    dwParmProbs: DWORD;
    dwSrcQuenchs: DWORD;
    dwRedirects: DWORD;
    dwEchos: DWORD;
    dwEchoReps: DWORD;
    dwTimeStamps: DWORD;
    dwTimeStampReps: DWORD;
    dwAddrMasks: DWORD;
    dwAddrReps: DWORD;
  end;

  PTMibICMPInfo = ^TMibICMPInfo;
  TMibICMPInfo = packed record
    InStats: TMibICMPStats;
    OutStats: TMibICMPStats;
  end;
//---------імпорт до M_API.DLL------------------
var

GetAdaptersInfo: function ( pAdapterInfo: PTIP_ADAPTER_INFO;
  pOutBufLen: PULONG ): DWORD; stdcall;
GetNetworkParams: function ( FixedInfo: PTFixedInfo; pOutPutLen: PULONG ):
        DWORD; stdcall;
GetTcpTable: function ( pTCPTable: PTMibTCPTable; pDWSize: PDWORD;
  bOrder: BOOL ): DWORD; stdcall;
GetTcpStatistics: function ( pStats: PTMibTCPStats ): DWORD; stdcall;
GetUdpTable: function ( pUdpTable: PTMibUDPTable; pDWSize: PDWORD;
 bOrder: BOOL ): DWORD; stdcall;
GetUdpStatistics: function ( pStats: PTMibUdpStats ): DWORD; stdcall;
GetIpStatistics: function ( pStats: PTMibIPStats ): DWORD; stdcall;
GetIpNetTable: function ( pIpNetTable: PTMibIPNetTable;
  pdwSize: PULONG;  bOrder: BOOL ): DWORD; stdcall;
GetIpAddrTable: function ( pIpAddrTable: PTMibIPAddrTable;
  pdwSize: PULONG; bOrder: BOOL ): DWORD; stdcall;
GetIpForwardTable: function ( pIPForwardTable: PTMibIPForwardTable;
  pdwSize: PULONG; bOrder: BOOL ): DWORD; stdCall;
GetIcmpStatistics: function ( pStats: PTMibICMPInfo ): DWORD; stdCall;
GetRTTAndHopCount: function ( DestIPAddress: DWORD; HopCount: PULONG;
  MaxHops: ULONG; RTT: PULONG ): BOOL; stdCall;
GetIfTable: function ( pIfTable: PTMibIfTable; pdwSize: PULONG;
  bOrder: boolean ): DWORD; stdCall;
GetIfEntry: function ( pIfRow: PTMibIfRow ): DWORD; stdCall;

// попередження – недокументована функція, можливі баги при використанні
GetFriendlyIfIndex: function (var IfIndex: DWORD): DWORD; stdcall;
const
    IpHlpDLL = ‘ M_API.DLL’;
var
```

```
    IpHlpModule: THandle;

    function LoadIpHlp: Boolean;

implementation

function LoadIpHlp: Boolean;
begin
    Result := True;
    if IpHlpModule <> 0 then Exit;

// відкрити DLL
    IpHlpModule := LoadLibrary (IpHlpDLL);
    if IpHlpModule = 0 then
    begin
        Result := false;
        exit ;
    end ;
    GetAdaptersInfo := GetProcAddress (IpHlpModule, ' GetAdaptersInfo') ;
    GetNetworkParams := GetProcAddress (IpHlpModule, ' GetNetworkParams') ;
    GetTcpTable := GetProcAddress (IpHlpModule, ' GetTcpTable') ;
    GetTcpStatistics := GetProcAddress (IpHlpModule, ' GetTcpStatistics') ;
    GetUdpTable := GetProcAddress (IpHlpModule, ' GetUdpTable') ;
    GetUdpStatistics := GetProcAddress (IpHlpModule, ' GetUdpStatistics') ;
    GetIpStatistics := GetProcAddress (IpHlpModule, ' GetIpStatistics') ;
    GetIpNetTable := GetProcAddress (IpHlpModule, ' GetIpNetTable') ;
    GetIpAddrTable := GetProcAddress (IpHlpModule, ' GetIpAddrTable') ;
    GetIpForwardTable := GetProcAddress (IpHlpModule, ' GetIpForwardTable') ;
    GetIcmpStatistics := GetProcAddress (IpHlpModule, ' GetIcmpStatistics') ;
    GetRTTAndHopCount := GetProcAddress (IpHlpModule, ' GetRTTAndHopCount') ;
    GetIfTable := GetProcAddress (IpHlpModule, ' GetIfTable') ;
    GetIfEntry := GetProcAddress (IpHlpModule, ' GetIfEntry') ;
    GetFriendlyIfIndex := GetProcAddress (IpHlpModule, ' GetFriendlyIfIndex') ;
end;

initialization
    IpHlpModule := 0 ;
finalization
    if IpHlpModule <> 0 then
    begin
        FreeLibrary (IpHlpModule) ;
        IpHlpModule := 0 ;
    end ;

end.
```

ФАЙЛ MYIP.PAS - ФУНКЦІЇ РОБОТИ З IP-ПРОТОКОЛОМ

```
unit MyIP; // опис модулю

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}
interface

uses
  Windows, Messages, SysUtils, Classes, Dialogs, M_API;

const
  NULL_IP = '0.0.0.0';

//---перетворення добре відомих номерів портів до імен сервісів--------
type
  TWellKnownPort = record
    Prt: DWORD;
    Srv: string[20];
  end;

const
// тільки найбільш популярні сервіси...
  WellKnownPorts: array[1..32] of TWellKnownPort
  = (
    ( Prt: 0; Srv:   ' RESRVED' ),    {Зарезервовано}
    ( Prt: 7; Srv:   ' ECHO   ' ),     {Ping    }
    ( Prt: 9; Srv:   ' DISCARD' ),
    ( Prt: 13; Srv:  ' DAYTIME' ),
    ( Prt: 17; Srv:  ' QOTD   ' ),     {Показчик на день}
    ( Prt: 19; Srv:  ' CHARGEN' ),     {Генератор символів}
    ( Prt: 20; Srv:  ' FTPDATA' ),     { File Transfer Protocol- данні}
    ( Prt: 21; Srv:  ' FTPCTRL' ),     { File Transfer Protocol- управління}
    ( Prt: 22; Srv:  ' SSH    ' ),
    ( Prt: 23; Srv:  ' TELNET ' ),
    ( Prt: 25; Srv:  ' SMTP   ' ),     { Simple Mail Transfer Protocol}
    ( Prt: 37; Srv:  ' TIME   ' ),     { Часовий протокол }
    ( Prt: 43; Srv:  ' WHOIS  ' ),     { Сервіс – Кто це  }
    ( Prt: 53; Srv:  ' DNS    ' ),     { Domain Name Service }
    ( Prt: 67; Srv:  ' BOOTPS ' ),     { BOOTP Сервер }
    ( Prt: 68; Srv:  ' BOOTPC ' ),     { BOOTP Кієнт }
    ( Prt: 69; Srv:  ' TFTP   ' ),     { стандартний  FTP  }
    ( Prt: 70; Srv:  ' GOPHER ' ),     { Протокол Gopher       }
    ( Prt: 79; Srv:  ' FINGER ' ),     { Протокол Finger       }
    ( Prt: 80; Srv:  ' HTTP   ' ),     { Протокол HTTP         }
    ( Prt: 88; Srv:  ' KERBROS' ),     { Протокол Kerberos     }
( Prt: 109; Srv: ' POP2   ' ),     { Протокол Post Office Protocol Version 2 }
( Prt: 110; Srv: ' POP3   ' ),     { Протокол Post Office Protocol Version 3 }
( Prt: 111; Srv: ' SUN_RPC' ),     { Протокол SUN Remote Procedure Call }
( Prt: 119; Srv: ' NNTP   ' ),     { Протокол Network News Transfer Protocol }
( Prt: 123; Srv: ' NTP    ' ),     { Протокол Network Time protocol          }
( Prt: 135; Srv: ' DCOMRPC' ),     { Протокол Location Service               }
    ( Prt: 137; Srv: ' NBNAME ' ),     { NETBIOS сервіс імен          }
```

```
    ( Prt: 138; Srv: ' NBDGRAM' ),     { NETBIOS сервіс датаграм     }
    ( Prt: 139; Srv: ' NBSESS '  ),     { NETBIOS сервіс сесій        }
(Prt: 143; Srv: ' IMAP   '  ),     { Протокол Internet Message Access Protocol }
( Prt: 161; Srv: ' SNMP '  ),     { Протокол Simple Netw. Management Protocol }
( Prt: 169; Srv: ' SEND   '  )
);

// перетворення ICMP кодів помилок до рядків

const
  ICMP_ERROR_BASE = 11000;
  IcmpErr : array[1..22] of string =
  (
' IP_BUFFER_TOO_SMALL','IP_DEST_NET_UNREACHABLE', ' IP_DEST_HOST_UNREACHABLE','
IP_PROTOCOL_UNREACHABLE', ' IP_DEST_PORT_UNREACHABLE', ' IP_NO_RESOURCES',
 ' IP_BAD_OPTION','IP_HARDWARE_ПОМИЛКА', ' IP_PACKET_TOO_BIG',
'IP_REQUEST_TIMED_OUT', ' IP_BAD_REQUEST','IP_BAD_ROUTE', '
IP_TTL_EXPIRED_TRANSIT',' IP_TTL_EXPIRED_REASSEM','IP_PARAMETER_PROBLEM', '
IP_SOURCE_QUENCH', ' IP_OPTION_TOO_BIG', '
IP_BAD_DESTINATION','IP_ADDRESS_DELETED', ' IP_SPEC_MTU_CHANGE', '
IP_MTU_CHANGE', ' IP_UNLOAD');

// Перетворення різних перерахованих величин у рядки
ARPEntryType:array[1..4] of string = ( ' інший', ' неправильний',
    ' динамічний', ' статичний');
TCPConnState:
    array[1..12] of string =
    ( ' closed', ' listening', ' syn_sent',
    ' syn_rcvd', ' established', ' fin_wait1',
    ' fin_wait2', ' close_wait', ' closing',
    ' last_ack', ' time_wait', ' delete_tcb'
);

  IPForwTypes   : array[1..4] of string =
    ( ' інший', ' invalid', ' local', ' remote' );

  IPForwProtos  : array[1..18] of string =
    ( ' інший', ' LOCAL', ' NETMGMT', ' ICMP', ' EGP',
    ' GGP', ' HELO', ' RIP', ' IS_IS', ' ES_IS',
    ' CISCO', ' BBN', ' OSPF', ' BGP', ' BOOTP',
    ' AUTO_STAT', ' STATIC', ' NOT_DOD' );

type
// для IpHlpNetworkParams
  TNetworkParams = record
    HostName: string ;
    DomainName: string ;
    CurrentDnsServer: string ;
    DnsServerTot: integer ;
    DnsServerNames: array [0..9] of string ;
    NodeType: UINT;
    ScopeID: string ;
    EnableRouting: UINT;
    EnableProxy: UINT;
    EnableDNS: UINT;
  end;

  TIfRows = array of TMibIfRow ; // динамічний масив колонок

// для IpHlpAdaptersInfo
```

```
  TAdaptorInfo = record
    AdapterName: string ;
    Description: string ;
    MacAddress: string ;
    Index: DWORD;
    aType: UINT;
    DHCPEnabled: UINT;
    CurrIPAddress: string ;
    CurrIPMask: string ;
    IPAddressTot: integer ;
    IPAddressList: array of string ;
    IPMaskList: array of string ;
    GatewayTot: integer ;
    GatewayList: array of string ;
    DHCPTot: integer ;
    DHCPServer: array of string ;
    HaveWINS: BOOL;
    PrimWINSTot: integer ;
    PrimWINSServer: array of string ;
    SecWINSTot: integer ;
    SecWINSServer: array of string ;
    LeaseObtained: LongInt ; // UNIX час, секунди з 1970
    LeaseExpires: LongInt;   // UNIX час, секунди з 1970
  end ;

  TAdaptorRows = array of TAdaptorInfo ;

// експортуємі данні

function IpHlpAdaptersInfo(var AdpTot: integer;var AdpRows: TAdaptorRows):
integer ;
procedure Get_AdaptersInfo( List: TStrings );
function IpHlpNetworkParams (var NetworkParams: TNetworkParams): integer ;
procedure Get_NetworkParams( List: TStrings );
procedure Get_ARPTable( List: TStrings );
procedure Get_TCPTable( List: TStrings );
procedure Get_TCPStatistics( List: TStrings );
function IpHlpTCPStatistics (var TCPStats: TMibTCPStats): integer ;
procedure Get_UDPTable( List: TStrings );
procedure Get_UDPStatistics( List: TStrings );
function IpHlpUdpStatistics (UdpStats: TMibUDPStats): integer ;
procedure Get_IPAddrTable( List: TStrings );
procedure Get_IPForwardTable( List: TStrings );
procedure Get_IPStatistics( List: TStrings );
function IpHlpIPStatistics (var IPStats: TMibIPStats): integer ;
function Get_RTTAndHopCount( IPAddr: DWORD; MaxHops: Longint;
  var RTT: longint; var HopCount: longint ): integer;
procedure Get_ICMPStats( ICMPIn, ICMPOut: TStrings );
function IpHlpIfTable(var IfTot: integer; var IfRows: TIfRows): integer ;
procedure Get_IfTable( List: TStrings );
function IpHlpIfEntry(Index: integer; var IfRow: TMibIfRow): integer ;
procedure Get_RecentDestIPs( List: TStrings );

// функції перетворення
function MacAddr2Str( MacAddr: TMacAddress; size: integer ): string;
function IpAddr2Str( IPAddr: DWORD ): string;
function Str2IpAddr( IPStr: string ): DWORD;
function Port2Str( nwoPort: DWORD ): string;
function Port2Wrd( nwoPort: DWORD ): DWORD;
function Port2Svc( Port: DWORD ): string;
```

```
function ICMPErr2Str( ICMPErrCode: DWORD) : string;
implementation

var
  RecentIPs     : TStringList;

//-------Основні функції

{ отримання наступного "токена" з рядка }
function NextToken( var s: string; Separator: char ): string;
var
  Sep_Pos       : byte;
begin
  Result := ' ' ;
  if length( s ) > 0 then begin
    Sep_Pos := pos( Separator, s );
    if Sep_Pos > 0 then begin
      Result := copy( s, 1, Pred( Sep_Pos ) );
      Delete( s, 1, Sep_Pos );
    end
    else begin
      Result := s;
      s := ' ' ;
    end;
  end;
end;

//-----------------------------------
{ перетворення числового MAC-адреса до ww-xx-yy-zz рядка }
function MacAddr2Str( MacAddr: TMacAddress; size: integer ): string;
var
  i : integer;
begin
  if Size = 0 then
  begin
    Result := '  00-00-00';
    EXIT;
  end
  else Result := ' ' ;
  for i := 1 to Size do
    Result := Result + IntToHex( MacAddr[i], 2) + ' -';
  Delete( Result, Length( Result ), 1 );
end;
//-----------------------------------
{ перетворення IP-адреси в мережний байт типу DWORD }
function IpAddr2Str( IPAddr: DWORD ): string;
var
  i : integer;
begin
  Result := ' ' ;
  for i := 1 to 4 do
  begin
    Result := Result + Format( ' %3d.', [IPAddr and $FF] );
    IPAddr := IPAddr shr 8;
  end;
  Delete( Result, Length( Result ), 1 );
end;

{ перетворення крапкової десяткової IP-адреси в мережний байт типу DWORD}
function Str2IpAddr( IPStr: string ): DWORD;
```

```
var
  i : integer;
  Num : DWORD;
begin
  Result := 0;
  for i := 1 to 4 do
  try
    Num := ( StrToInt( NextToken( IPStr, ‘ .’ ) ) ) shl 24;
    Result := ( Result shr 8 ) or Num;
  except
    Result := 0;
  end;

end;

//-----------------------------------
{ перетворення номеру порту в мережний байт типу DWORD }
function Port2Wrd( nwoPort: DWORD ): DWORD;
begin
  Result := Swap( WORD( nwoPort ) );
end;

//-----------------------------------
{ перетворення номеру порту в мережний байт типу string }
function Port2Str( nwoPort: DWORD ): string;
begin
  Result := IntToStr( Port2Wrd( nwoPort ) );
end;

//-----------------------------------
{ перетворення номеру порту в сервіс ID }
function Port2Svc( Port: DWORD ): string;
var
  i             : integer;
begin
  Result := Format( ‘ %4d’, [Port] ); // у випадку, якщо порт не знайдено
  for i := Low( WellKnownPorts ) to High( WellKnownPorts ) do
    if Port = WellKnownPorts[i].Prt then
    begin
      Result := WellKnownPorts[i].Srv;
      BREAK;
    end;
end;

//-----------------------------------
{ головна частина, фіксація мережний параметрів }

procedure Get_NetworkParams( List: TStrings );
var
    NetworkParams: TNetworkParams ;
    I, ErrorCode: integer ;
begin
    if not Assigned( List ) then EXIT;
    List.Clear;
    ErrorCode := IpHlpNetworkParams (NetworkParams) ;
    if ErrorCode <> 0 then
    begin
        List.Add (SysErrorMessage (ErrorCode));
        exit;
    end ;
```

```
    with NetworkParams do
    begin
        List.Add( ' Ім'я хосту          : '  + HostName );
        List.Add( ' Домен            : '  + DomainName );
        List.Add( ' NETBIOS тип : '  + NETBIOSTypes[NodeType] );
        List.Add( ' DHCP область        : '  + ScopeID );
        List.Add( ' ROUTING визначено   : '  + IntToStr( EnableRouting ) );
        List.Add( ' PROXY   визначено   : '  + IntToStr( EnableProxy ) );
        List.Add( ' DNS     визначено   : '  + IntToStr( EnableDNS ) );
        if DnsServerTot <> 0 then
        begin
            for I := 0 to Pred (DnsServerTot) do
                List.Add( ' DNS адреса серверу   : '  + DnsServerNames [I] ) ;
        end ;
    end ;
end ;

//-------------------------------------//
function IpHlpNetworkParams (var NetworkParams: TNetworkParams): integer ;
var
  FixedInfo     : PTFixedInfo;         // данні
  InfoSize      : Longint;
  PDnsServer    : PTIP_ADDR_STRING ;   // данні
begin
    InfoSize := 0 ;   // данні
    result := ERROR_NOT_SUPPORTED ;
    if NOT LoadIpHlp then exit ;
    result := GetNetworkParams( Nil, @InfoSize );  // данні
    if result <> ERROR_BUFFER_OVERFLOW then exit ; // данні
    GetMem (FixedInfo, InfoSize) ;                 // данні
    try
    result := GetNetworkParams( FixedInfo, @InfoSize );   // данні
    if result <> ERROR_SUCCESS then exit ;
    NetworkParams.DnsServerTot := 0 ;
    with FixedInfo^ do
    begin
        NetworkParams.HostName := trim (HostName) ;
        NetworkParams.DomainName := trim (DomainName) ;
        NetworkParams.ScopeId := trim (ScopeID) ;
        NetworkParams.NodeType := NodeType ;
        NetworkParams.EnableRouting := EnableRouting ;
        NetworkParams.EnableProxy := EnableProxy ;
        NetworkParams.EnableDNS := EnableDNS ;
        NetworkParams.DnsServerNames [0] := DNSServerList.IPAddress ;  // данні
        if NetworkParams.DnsServerNames [0] <> ' '  then
                                    NetworkParams.DnsServerTot := 1 ;
        PDnsServer := DnsServerList.Next;
        while PDnsServer <> Nil do
        begin
            NetworkParams.DnsServerNames [NetworkParams.DnsServerTot] :=
                                    PDnsServer^.IPAddress ;  // данні
            inc (NetworkParams.DnsServerTot) ;
            if NetworkParams.DnsServerTot >=
                            Length (NetworkParams.DnsServerNames) then exit ;
            PDnsServer := PDnsServer.Next ;
        end;
    end ;
    finally
       FreeMem (FixedInfo) ;                    // данні
    end ;
```

```
end;
//-----------------------------------
function ICMPErr2Str( ICMPErrCode: DWORD) : string;
begin
   Result := ' UnknownError : '  + IntToStr( ICMPErrCode );
   dec( ICMPErrCode, ICMP_ERROR_BASE );
   if ICMPErrCode in [Low(ICMpErr)..High(ICMPErr)] then
     Result := ICMPErr[ ICMPErrCode];
end;
//-----------------------------------

// включення байтів у/з для кожного адаптера

function IpHlpIfTable(var IfTot: integer; var IfRows: TIfRows): integer ;
var
  I,
  TableSize   : integer;
  pBuf, pNext : PChar;
begin
  result := ERROR_NOT_SUPPORTED ;
  if NOT LoadIpHlp then exit ;
  SetLength (IfRows, 0) ;
  IfTot := 0 ; // данні
  TableSize := 0;
   // перший виклик: необхідно отримати розмір пам'яті
  result := GetIfTable (Nil, @TableSize, false) ;  // данні
  if result <> ERROR_INSUFFICIENT_BUFFER then exit ;
  GetMem( pBuf, TableSize );
  try
      FillChar (pBuf^, TableSize, #0);  // очищаємо буфер з  W98 не беремо
крапку таблиці
      result := GetIfTable (PTMibIfTable (pBuf), @TableSize, false) ;
      if result <> NO_ERROR then exit ;
      IfTot := PTMibIfTable (pBuf)^.dwNumEntries ;
      if IfTot = 0 then exit ;
      SetLength (IfRows, IfTot) ;
      pNext := pBuf + SizeOf(IfTot) ;
      for i := 0 to Pred (IfTot) do
      begin
         IfRows [i] := PTMibIfRow (pNext )^ ;
         inc (pNext, SizeOf (TMibIfRow)) ;
      end;
  finally
      FreeMem (pBuf) ;
  end ;
end;

procedure Get_IfTable( List: TStrings );
var
  IfRows        : TIfRows ;
  Error, I      : integer;
  NumEntries    : integer;
  sDescr, sIfName: string ;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;
  SetLength (IfRows, 0) ;
  Error := IpHlpIfTable (NumEntries, IfRows) ;
  if (Error <> 0) then
      List.Add( SysErrorMessage( GetLastError ) )
```

```
  else if NumEntries = 0 then
      List.Add( ' данних немає ' )
  else
  begin
      for I := 0 to Pred (NumEntries) do
      begin
          with IfRows [I] do
          begin
             if wszName [1] = #0 then
                 sIfName := ' '
             else
                 sIfName := WideCharToString (@wszName) ;
// конвертуємо Юнікод до рядка
             sIfName := trim (sIfName) ;
             sDescr := bDescr ;
             sDescr := trim (sDescr);
             List.Add (Format (
               ' %0.8x |%3d | %16s |%8d |%12d |%2d |%2d |%10d |%10d | %-s| %-s',
               [dwIndex, dwType, MacAddr2Str( TMacAddress( bPhysAddr ),
               dwPhysAddrLen ), dwMTU, dwSpeed, dwAdminStatus,
               dwOPerStatus, Int64 (dwInOctets), Int64 (dwOutOctets),
// конвертуємо до 32-біт
               sIfName, sDescr] )  // данні, додані в/з
               );
          end;
      end ;
  end ;
  SetLength (IfRows, 0) ;  // вільна пам'ять
end ;

function IpHlpIfEntry(Index: integer; var IfRow: TMibIfRow): integer ;
begin
  result := ERROR_NOT_SUPPORTED ;
  if NOT LoadIpHlp then exit ;
  FillChar (IfRow, SizeOf (TMibIfRow), #0);
// очищаємо буфер з  W98 не беремо
  IfRow.dwIndex := Index ;
  result := GetIfEntry (@IfRow) ;
end ;

//-----------------------------------
{ інформація про інстальовані адаптери }

function IpHlpAdaptersInfo(var AdpTot: integer; var AdpRows: TAdaptorRows):
integer ;
var
  BufLen        : DWORD;
  AdapterInfo   : PTIP_ADAPTER_INFO;
  PIpAddr       : PTIP_ADDR_STRING;
  PBuf          : PCHAR ;
  I             : integer ;
begin
  SetLength (AdpRows, 4) ;
  AdpTot := 0 ;
  BufLen := 0 ;
  result := GetAdaptersInfo( Nil, @BufLen );
  if (result <> ERROR_INSUFFICIENT_BUFFER) and (result = NO_ERROR) then exit ;
  GetMem( pBuf, BufLen );
  try
      FillChar (pBuf^, BufLen, #0);
```

```
// очищуємо буфер
      result := GetAdaptersInfo( PTIP_ADAPTER_INFO (PBuf), @BufLen );
      if result = NO_ERROR then
      begin
         AdapterInfo := PTIP_ADAPTER_INFO (PBuf) ;
         while ( AdapterInfo <> nil ) do
         begin
            AdpRows [AdpTot].IPAddressTot := 0 ;
            SetLength (AdpRows [AdpTot].IPAddressList, 2) ;
            SetLength (AdpRows [AdpTot].IPMaskList, 2) ;
            AdpRows [AdpTot].GatewayTot := 0 ;
            SetLength (AdpRows [AdpTot].GatewayList, 2) ;
            AdpRows [AdpTot].DHCPTot := 0 ;
            SetLength (AdpRows [AdpTot].DHCPServer, 2) ;
            AdpRows [AdpTot].PrimWINSTot := 0 ;
            SetLength (AdpRows [AdpTot].PrimWINSServer, 2) ;
            AdpRows [AdpTot].SecWINSTot := 0 ;
            SetLength (AdpRows [AdpTot].SecWINSServer, 2) ;
            AdpRows [AdpTot].CurrIPAddress := NULL_IP;
            AdpRows [AdpTot].CurrIPMask := NULL_IP;
AdpRows [AdpTot].AdapterName := Trim( string( AdapterInfo^.AdapterName ) );
AdpRows [AdpTot].Description := Trim( string( AdapterInfo^.Description ) );
AdpRows [AdpTot].MacAddress := MacAddr2Str( TMacAddress(
                 AdapterInfo^.Address ),AdapterInfo^.AddressLength ) ;
            AdpRows [AdpTot].Index := AdapterInfo^.Index ;
            AdpRows [AdpTot].aType := AdapterInfo^.aType ;
            AdpRows [AdpTot].DHCPEnabled := AdapterInfo^.DHCPEnabled ;
            if AdapterInfo^.CurrentIPAddress <> Nil then
            begin
AdpRows [AdpTot].CurrIPAddress := AdapterInfo^.CurrentIPAddress.IpAddress ;
AdpRows [AdpTot].CurrIPMask := AdapterInfo^.CurrentIPAddress.IpMask ;
            end ;
        // беремо список IP адрес та конвертуємо в IPAddressList
            I := 0 ;
            PIpAddr := @AdapterInfo^.IPAddressList ;
            while (PIpAddr <> Nil) do
            begin
                AdpRows [AdpTot].IPAddressList [I] := PIpAddr.IpAddress ;
                AdpRows [AdpTot].IPMaskList [I] := PIpAddr.IpMask ;
                PIpAddr := PIpAddr.Next ;
                inc (I) ;
                if Length (AdpRows [AdpTot].IPAddressList) <= I then
                begin
                     SetLength (AdpRows [AdpTot].IPAddressList, I -2) ;
                     SetLength (AdpRows [AdpTot].IPMaskList, I -2) ;
                end ;
            end ;
            AdpRows [AdpTot].IPAddressTot := I ;
// беремо список IP адрес для GatewayList
            I := 0 ;
            PIpAddr := @AdapterInfo^.GatewayList ;
            while (PIpAddr <> Nil) do
            begin
                AdpRows [AdpTot].GatewayList [I] := PIpAddr.IpAddress ;
                PIpAddr := PIpAddr.Next ;
                inc (I) ;
                if Length (AdpRows [AdpTot].GatewayList) <= I then
                             SetLength (AdpRows [AdpTot].GatewayList, I -2) ;
            end ;
            AdpRows [AdpTot].GatewayTot := I ;
```

```
        // беремо список IP адрес для  GatewayList
             I := 0 ;
             PIpAddr := @AdapterInfo^.DHCPServer ;
             while (PIpAddr <> Nil) do
             begin
                 AdpRows [AdpTot].DHCPServer [I] := PIpAddr.IpAddress ;
                 PIpAddr := PIpAddr.Next ;
                 inc (I) ;
                 if Length (AdpRows [AdpTot].DHCPServer) <= I then
                             SetLength (AdpRows [AdpTot].DHCPServer, I -2) ;
             end ;
             AdpRows [AdpTot].DHCPTot := I ;
         // беремо список IP адрес для PrimaryWINSServer
             I := 0 ;
             PIpAddr := @AdapterInfo^.PrimaryWINSServer ;
             while (PIpAddr <> Nil) do
             begin
                 AdpRows [AdpTot].PrimWINSServer [I] := PIpAddr.IpAddress ;
                 PIpAddr := PIpAddr.Next ;
                 inc (I) ;
                 if Length (AdpRows [AdpTot].PrimWINSServer) <= I then
                             SetLength (AdpRows [AdpTot].PrimWINSServer, I -2) ;
             end ;
             AdpRows [AdpTot].PrimWINSTot := I ;
        // беремо список IP адрес для SecondaryWINSServer
             I := 0 ;
             PIpAddr := @AdapterInfo^.SecondaryWINSServer ;
             while (PIpAddr <> Nil) do
             begin
                 AdpRows [AdpTot].SecWINSServer [I] := PIpAddr.IpAddress ;
                 PIpAddr := PIpAddr.Next ;
                 inc (I) ;
                 if Length (AdpRows [AdpTot].SecWINSServer) <= I then
                             SetLength (AdpRows [AdpTot].SecWINSServer, I -2) ;
             end ;
             AdpRows [AdpTot].SecWINSTot := I ;
             AdpRows [AdpTot].LeaseObtained := AdapterInfo^.LeaseObtained ;
             AdpRows [AdpTot].LeaseExpires := AdapterInfo^.LeaseExpires ;
             inc (AdpTot) ;
             if Length (AdpRows) <= AdpTot then
                            SetLength (AdpRows, AdpTot -2) ;  // більше пам'яті
             AdapterInfo := AdapterInfo^.Next;
          end ;
          SetLength (AdpRows, AdpTot) ;
       end ;
  finally
       FreeMem( pBuf );
  end ;
end ;

procedure Get_AdaptersInfo( List: TStrings );
var
  AdpTot: integer;
  AdpRows: TAdaptorRows ;
  Error: DWORD ;
  I: integer ;
  //J: integer ;
  //S: string ;        id.
begin
  if not Assigned( List ) then EXIT;
```

```
  List.Clear;
  SetLength (AdpRows, 0) ;
  AdpTot := 0 ;
  Error := IpHlpAdaptersInfo(AdpTot, AdpRows) ;
  if (Error <> 0) then
      List.Add( SysErrorMessage( GetLastError ) )
  else if AdpTot = 0 then
      List.Add( ' данних немає ' )
  else
  begin
      for I := 0 to Pred (AdpTot) do
      begin
        with AdpRows [I] do
        begin
List.Add(AdapterName + ' |' + Description ); // jpt : не використовується
         List.Add( Format(' %8.8x | %6s | %16s | %2d | %16s | %16s | %16s',
         [Index, AdaptTypes[aType], MacAddress, DHCPEnabled,
         GatewayList [0], DHCPServer [0], PrimWINSServer [0]] ) );
if IPAddressTot <> 0 then    // jpt : не використовується
           begin
               S := ' '  ;
               for J := 0 to Pred (IPAddressTot) do
               S := S + IPAddressList [J] + ' /' + IPMaskList [J] + '  | '  ;
               List.Add(IntToStr (IPAddressTot) + '  IP Addresse(s): '  + S);
           end ;
           List.Add( '   '  );
        end ;
      end ;
  end ;
  SetLength (AdpRows, 0) ;
end ;
//-----------------------------------
{ моні торимо час доступу до IP }
function Get_RTTAndHopCount( IPAddr: DWORD; MaxHops: Longint; var RTT: Longint;
  var HopCount: Longint ): integer;
begin
  if not GetRTTAndHopCount( IPAddr, @HopCount, MaxHops, @RTT ) then
  begin
    Result := GetLastError;
    RTT :=-1; // Расположення BAD_HOST_NAME,etc...
    HopCount :=-1;
  end
  else
    Result := NO_ERROR;
end;
//-----------------------------------
{ ARP-таблиця включає відношення між віддаленим IP та віддаленим MAC-адресом }
procedure Get_ARPTable( List: TStrings );
var
  IPNetRow      : TMibIPNetRow;
  TableSize     : DWORD;
  NumEntries    : DWORD;
  ErrorCode     : DWORD;
  i             : integer;
  pBuf          : PChar;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;
// перший виклик: беремо довжину таблиці
  TableSize := 0;
```

```
  ErrorCode := GetIPNetTable( Nil, @TableSize, false );   // данні
  if ErrorCode = ERROR_NO_DATA then
  begin
    List.Add( '  ARP-кеш пустий.' );
    EXIT;
  end;
// беремо таблицю
  GetMem( pBuf, TableSize );
  NumEntries := 0 ;
  try
  ErrorCode := GetIpNetTable( PTMIBIPNetTable( pBuf ), @TableSize, false );
  if ErrorCode = NO_ERROR then
  begin
    NumEntries := PTMIBIPNetTable( pBuf )^.dwNumEntries;
    if NumEntries > 0 then
    begin
      inc( pBuf, SizeOf( DWORD ) );
// бермо розмір останьої таблиці
      for i := 1 to NumEntries do
      begin
        IPNetRow := PTMIBIPNetRow( PBuf )^;
        with IPNetRow do
          List.Add( Format( ' %8x | %12s | %16s | %10s',
                            [dwIndex, MacAddr2Str( bPhysAddr, dwPhysAddrLen ),
                            IPAddr2Str( dwAddr ), ARPEntryType[dwType]
                            ]));
        inc( pBuf, SizeOf( IPNetRow ) );
      end;
    end
    else
      List.Add( '  ARP-кеш пустий.' );
  end
  else
    List.Add( SysErrorMessage( ErrorCode ) );
  // необхідно відновити показник!
  finally
      dec( pBuf, SizeOf( DWORD ) + NumEntries -SizeOf( IPNetRow ) );
      FreeMem( pBuf );
  end ;
end;
//-----------------------------------
procedure Get_TCPTable( List: TStrings );
var
  TCPRow : TMIBTCPRow;
  i,
    NumEntries  : integer;
  TableSize     : DWORD;
  ErrorCode     : DWORD;
  DestIP        : string;
  pBuf          : PChar;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;
  RecentIPs.Clear;
// перший виклик: беремо довжину таблиці
  TableSize := 0;
  NumEntries := 0 ;
  ErrorCode := GetTCPTable(Nil, @TableSize, false );
// данні
  if Errorcode <> ERROR_INSUFFICIENT_BUFFER then
```

```
EXIT;
  // беремо розмір пам'яті, викликаємо знову
  GetMem( pBuf, TableSize );
  // беремо таблицю
  ErrorCode := GetTCPTable( PTMIBTCPTable( pBuf ), @TableSize, false );
  if ErrorCode = NO_ERROR then
  begin
    NumEntries := PTMIBTCPTable( pBuf )^.dwNumEntries;
    if NumEntries > 0 then
    begin
      inc( pBuf, SizeOf( DWORD ) );
// беремо розмір останьої таблиці
      for i := 1 to NumEntries do
      begin
        TCPRow := PTMIBTCPRow( pBuf )^;
// беремо останій запис
        with TCPRow do
        begin
          if dwRemoteAddr = 0 then
            dwRemotePort := 0;
          DestIP := IPAddr2Str( dwRemoteAddr );
          List.Add(
            Format( ' %15s : %-7s | %15s : %-7s | %-16s',
            [IpAddr2Str( dwLocalAddr ),
            Port2Svc( Port2Wrd( dwLocalPort ) ),
              DestIP,
              Port2Svc( Port2Wrd( dwRemotePort ) ),
              TCPConnState[dwState]
              ] ) );
            if (not ( dwRemoteAddr = 0 ))
            and ( RecentIps.IndexOf(DestIP) =-1 ) then
               RecentIPs.Add( DestIP );
        end;
        inc( pBuf, SizeOf( TMIBTCPRow ) );
      end;
    end;
  end
  else
    List.Add( SyserrorMessage( ErrorCode ) );
  dec( pBuf, SizeOf( DWORD ) + NumEntries -SizeOf( TMibTCPRow ) );
  FreeMem( pBuf );
end;

//-----------------------------------
procedure Get_TCPStatistics( List: TStrings );
var
  TCPStats      : TMibTCPStats;
  ErrorCode     : DWORD;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;
  if NOT LoadIpHlp then exit ;
  ErrorCode := GetTCPStatistics( @TCPStats );
  if ErrorCode = NO_ERROR then
    with TCPStats do
    begin
List.Add(' Алгоритм повторної передачі:'  + TCPToAlgo[dwRTOAlgorithm] );
List.Add('Мінімальний час виходу : '  + IntToStr( dwRTOMin ) + '  ms' );
List.Add(' Максимальний час виходу:' + IntToStr( dwRTOMax ) + '  ms');
List.Add( ' Максимальне число підключень : '  + IntToStr( dwRTOAlgorithm ) );
```

```
List.Add( ' Активні підключення             : '  + IntToStr( dwActiveOpens ) );
List.Add( ' пасивні підключення            : '  + IntToStr( dwPassiveOpens ) );
List.Add( ' Невдала спроба відкриття     : '  + IntToStr( dwAttemptFails ) );
List.Add( ' Скидання встановленого підключення:' + IntToStr( dwEstabResets ) );
List.Add( ' Поточне встановлене підключення.: '  + IntToStr( dwCurrEstab ) );
List.Add( ' Отримані сегменти        : '  + IntToStr( dwInSegs ) );
List.Add( ' Передані сегменти            : '  + IntToStr( dwOutSegs ) );
List.Add( ' Перепідключені сегменти   : '  + IntToStr( dwReTransSegs ) );
List.Add( ' помилка входу          : '  + IntToStr( dwInErrs ) );
List.Add( ' Перезавантаження вихідних          : '  + IntToStr( dwOutRsts ) );
List.Add( ' Совокупні зв'язки    : '  + IntToStr( dwNumConns ) );
    end
  else
List.Add( SyserrorMessage( ErrorCode ) );
end;

function IpHlpTCPStatistics (var TCPStats: TMibTCPStats): integer ;
begin
    result := ERROR_NOT_SUPPORTED ;
    if NOT LoadIpHlp then exit ;
    result := GetTCPStatistics( @TCPStats );
end;

//-----------------------------------
procedure Get_UDPTable( List: TStrings );
var
  UDPRow        : TMIBUDPRow;
  i,
    NumEntries  : integer;
  TableSize     : DWORD;
  ErrorCode     : DWORD;
  pBuf          : PChar;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;

  // перший виклик: беремо довжину таблиці
  TableSize := 0;
  NumEntries := 0 ;
  ErrorCode := GetUDPTable(Nil, @TableSize, false );
  if Errorcode <> ERROR_INSUFFICIENT_BUFFER then
    EXIT;
  // виділяємо пам'ять, викликаємо знову
  GetMem( pBuf, TableSize );

  // беремо таблицю
  ErrorCode := GetUDPTable( PTMIBUDPTable( pBuf ), @TableSize, false );
  if ErrorCode = NO_ERROR then
  begin
    NumEntries := PTMIBUDPTable( pBuf )^.dwNumEntries;
    if NumEntries > 0 then
    begin
      inc( pBuf, SizeOf( DWORD ) ); // беремо розмір останьої таблиці
      for i := 1 to NumEntries do
      begin
        UDPRow := PTMIBUDPRow( pBuf )^; // беремо останій запис
        with UDPRow do
          List.Add( Format( ' %15s : %-6s',
            [IpAddr2Str( dwLocalAddr ),
            Port2Svc( Port2Wrd( dwLocalPort ) )
```

```
              ] ) );
        inc( pBuf, SizeOf( TMIBUDPRow ) );
      end;
    end
    else
      List.Add( ' немає даних.' );
  end
  else
    List.Add( SyserrorMessage( ErrorCode ) );
  dec( pBuf, SizeOf( DWORD ) + NumEntries -SizeOf( TMibUDPRow ) );
  FreeMem( pBuf );
end;

//----------------------------------
procedure Get_IPAddrTable( List: TStrings );
var
  IPAddrRow     : TMibIPAddrRow;
  TableSize     : DWORD;
  ErrorCode     : DWORD;
  i             : integer;
  pBuf          : PChar;
  NumEntries    : DWORD;
begin
  if not Assigned( List ) then EXIT;
  List.Clear;
  TableSize := 0; ;
  NumEntries := 0 ;
  // перший виклик: беремо довжину таблиці
  ErrorCode := GetIpAddrTable(Nil, @TableSize, true );  // данні
  if Errorcode <> ERROR_INSUFFICIENT_BUFFER then
    EXIT;

  GetMem( pBuf, TableSize );
  // беремо таблицю
  ErrorCode := GetIpAddrTable( PTMibIPAddrTable( pBuf ), @TableSize, true );
  if ErrorCode = NO_ERROR then
  begin
    NumEntries := PTMibIPAddrTable( pBuf )^.dwNumEntries;
    if NumEntries > 0 then
    begin
      inc( pBuf, SizeOf( DWORD ) );
      for i := 1 to NumEntries do
      begin
        IPAddrRow := PTMIBIPAddrRow( pBuf )^;
        with IPAddrRow do
          List.Add( Format( ' %8.8x | %15s | %15s | %15s | %8.8d',
            [dwIndex,
            IPAddr2Str( dwAddr ),
              IPAddr2Str( dwMask ),
              IPAddr2Str( dwBCastAddr ),
              dwReasmSize
              ] ) );
        inc( pBuf, SizeOf( TMIBIPAddrRow ) );
      end;
    end
    else
      List.Add( ' немає даних.' );
  end
  else
    List.Add( SysErrorMessage( ErrorCode ) );
```

```
  // відновлюємо показчик!
  dec( pBuf, SizeOf( DWORD ) + NumEntries -SizeOf( IPAddrRow ) );
  FreeMem( pBuf );
end;
//-----------------------------------
{ отримуємо дані з таблиці маршрутизації; }
procedure Get_IPForwardTable( List: TStrings );
var
  IPForwRow     : TMibIPForwardRow;
  TableSize     : DWORD;
  ErrorCode     : DWORD;
  i             : integer;
  pBuf          : PChar;
  NumEntries    : DWORD;
begin

  if not Assigned( List ) then EXIT;
  List.Clear;
  TableSize := 0;

  // перший виклик: беремо довжину таблиці
  NumEntries := 0 ;
  ErrorCode := GetIpForwardTable(Nil, @TableSize, true);
  if Errorcode <> ERROR_INSUFFICIENT_BUFFER then
    EXIT;

  // беремо таблицю
  GetMem( pBuf, TableSize );
  ErrorCode := GetIpForwardTable( PTMibIPForwardTable( pBuf ), @TableSize,
true);
  if ErrorCode = NO_ERROR then
  begin
    NumEntries := PTMibIPForwardTable( pBuf )^.dwNumEntries;
    if NumEntries > 0 then
    begin
      inc( pBuf, SizeOf( DWORD ) );
      for i := 1 to NumEntries do
      begin
        IPForwRow := PTMibIPForwardRow( pBuf )^;
        with IPForwRow do
        begin
          if (dwForwardType < 1)
          or (dwForwardType > 4) then
                   dwForwardType := 1 ;   // данні
          List.Add( Format(
            ' %15s | %15s | %15s | %8.8x | %7s |   %5.5d |  %7s |   %2.2d',
            [IPAddr2Str( dwForwardDest ),
            IPAddr2Str( dwForwardMask ),
              IPAddr2Str( dwForwardNextHop ),
              dwForwardIFIndex,
              IPForwTypes[dwForwardType],
              dwForwardNextHopAS,
              IPForwProtos[dwForwardProto],
              dwForwardMetric1
              ] ) );
        end ;
        inc( pBuf, SizeOf( TMibIPForwardRow ) );
      end;
    end
```

```
    else
      List.Add( ' немає даних.' );
  end
  else
    List.Add( SysErrorMessage( ErrorCode ) );
  dec( pBuf, SizeOf( DWORD ) + NumEntries -SizeOf( TMibIPForwardRow ) );
  FreeMem( pBuf );
end;

//-----------------------------------
procedure Get_IPStatistics( List: TStrings );
var
  IPStats       : TMibIPStats;
  ErrorCode     : integer;
begin
  if not Assigned( List ) then EXIT;
  if NOT LoadIpHlp then exit ;
  ErrorCode := GetIPStatistics( @IPStats );
  if ErrorCode = NO_ERROR then
  begin
    List.Clear;
    with IPStats do
    begin
      if dwForwarding = 1 then
        List.add( ' Розблокована пересилка     : '  + ' так' )
      else
        List.add(' Розблокована пересилка     : '  + ' ні' );
      List.add( ' Любий TTL              : '  + inttostr( dwDefaultTTL ) );
   List.add( ' Датаграмма прийнята     : '  + inttostr( dwInReceives ) );
   List.add( ' Помилка заголовку      (In)  : '  + inttostr( dwInHdrErrors ) );
      List.add( ' Помилка адреси    (In)  : '  + inttostr( dwInAddrErrors ) );
      List.add( ' Датаграма переслана     : '  + inttostr( dwForwDatagrams ) );
// данні
List.add( ' Невизначений протокол (In)  : '  + inttostr( dwInUnknownProtos ) );
      List.add( ' Датаграмма відмовлена     : '  + inttostr( dwInDiscards ) );
      List.add( ' Датаграма встановлена     : '  + inttostr( dwInDelivers ) );
   List.add( ' Зовнішній запрос            : '  + inttostr( dwOutRequests ) );
      List.add( ' Маршрутизація не виконана      : '  + inttostr(
dwRoutingDiscards ) );
   List.add( ' Немає маршрутів        (Out)  : '  + inttostr( dwOutNoRoutes ) );
      List.add( ' Перебраний час     : '  + inttostr( dwReasmTimeOut ) );
      List.add( ' Запит перебору     : '  + inttostr( dwReasmReqds ) );
      List.add( ' Повний перебор : '  + inttostr( dwReasmOKs ) );
      List.add( ' Помилка перебору     : '  + inttostr( dwReasmFails ) );
      List.add( ' Повна фрагментація: '  + inttostr( dwFragOKs ) );
      List.add( ' Помилка фрагментації   : '  + inttostr( dwFragFails ) );
      List.add( ' Датаграмма фрагментована    : '  + inttostr( dwFRagCreates )
);
      List.add( ' Кількість інтерфейсів    : '  + inttostr( dwNumIf ) );
      List.add( ' Кількість IP-адрес  : '  + inttostr( dwNumAddr ) );
      List.add( ' Маршрут в таблиці маршрутизатора  : '  + inttostr( dwNumRoutes
) );
    end;
  end
  else
    List.Add( SysErrorMessage( ErrorCode ) );
end;

function IpHlpIPStatistics (var IPStats: TMibIPStats): integer ;      // данні
begin
```

```
    result := ERROR_NOT_SUPPORTED ;
    if NOT LoadIpHlp then exit ;
    result := GetIPStatistics( @IPStats );
end ;

//-----------------------------------
procedure Get_UdpStatistics( List: TStrings );
var
  UdpStats      : TMibUDPStats;
  ErrorCode     : integer;
begin
  if not Assigned( List ) then EXIT;
  ErrorCode := GetUDPStatistics( @UdpStats );
  if ErrorCode = NO_ERROR then
  begin
    List.Clear;
    with UDPStats do
    begin
      List.add( ' Датаграми (In)    : '  + inttostr( dwInDatagrams ) );
      List.add( '  Датаграми (Out)   : '  + inttostr( dwOutDatagrams ) );
      List.add( ' Немає портів         : '  + inttostr( dwNoPorts ) );
      List.add( ' Помилка    (In)    : '  + inttostr( dwInErrors ) );
      List.add( ' UDP список портів  : '  + inttostr( dwNumAddrs ) );
    end;
  end
  else
    List.Add( SysErrorMessage( ErrorCode ) );
end;

//-------------------------------------*//
function IpHlpUdpStatistics (UdpStats: TMibUDPStats): integer ;     // данні
begin
    result := ERROR_NOT_SUPPORTED ;
    if NOT LoadIpHlp then exit ;
    result := GetUDPStatistics (@UdpStats) ;
end ;

//-----------------------------------
procedure Get_ICMPStats( ICMPIn, ICMPOut: TStrings );
var
  ErrorCode     : DWORD;
  ICMPStats     : PTMibICMPInfo;
begin
  if ( ICMPIn = nil ) or ( ICMPOut = nil ) then EXIT;
  ICMPIn.Clear;
  ICMPOut.Clear;
  New( ICMPStats );
  ErrorCode := GetICMPStatistics( ICMPStats );
  if ErrorCode = NO_ERROR then
  begin
    with ICMPStats.InStats do
    begin
      ICMPIn.Add( '  Прийнято повідомлень    : '  + IntToStr( dwMsgs ) );
      ICMPIn.Add( ' Помилка              : '  + IntToStr( dwErrors ) );
      ICMPIn.Add( ' Розташування недосягенно    : '  + IntToStr( dwDestUnreachs
) );
      ICMPIn.Add( ' Час перевищений        : '  + IntToStr( dwTimeEcxcds ) );
      ICMPIn.Add( ' Проблеми з параметрами      : '  + IntToStr( dwParmProbs )
);
      ICMPIn.Add( ' Джерело відключено        : '  + IntToStr( dwSrcQuenchs ) );
```

```
      ICMPIn.Add( ' Переназначено            : '  + IntToStr( dwRedirects ) );
      ICMPIn.Add( ' Exo запит        : '  + IntToStr( dwEchos ) );
      ICMPIn.Add( ' Exo відповідь          : '  + IntToStr( dwEchoReps ) );
      ICMPIn.Add( ' Запит мітки часу   : '  + IntToStr( dwTimeStamps ) );
   ICMPIn.Add( ' Відповідь мітки часу    : '  + IntToStr( dwTimeStampReps ) );
      ICMPIn.Add( ' Запит маски адрес : '  + IntToStr( dwAddrMasks ) );
      ICMPIn.Add( ' Відповідь маски адрес   : '  + IntToStr( dwAddrReps ) );
    end;
     //
    with ICMPStats.OutStats do
    begin
      ICMPOut.Add( ' Повідомлення вдправлено        : '  + IntToStr( dwMsgs ) );
      ICMPOut.Add( 'Помилка               : '  + IntToStr( dwErrors ) );
  ICMPOut.Add( 'Розташування недосягенно    : '  + IntToStr( dwDestUnreachs ) );
      ICMPOut.Add( 'Час перевищений        : '  + IntToStr( dwTimeEcxcds ) );
   ICMPOut.Add( 'Проблеми з параметрами      : '  + IntToStr( dwParmProbs ) );
      ICMPOut.Add( 'Джерело відключено        : '  + IntToStr( dwSrcQuenchs ) );
      ICMPOut.Add( 'Переназначено            : '  + IntToStr( dwRedirects ) );
      ICMPOut.Add( 'Exo запит        : '  + IntToStr( dwEchos ) );
      ICMPOut.Add( 'Exo відповідь          : '  + IntToStr( dwEchoReps ) );
      ICMPOut.Add( 'Запит мітки часу   : '  + IntToStr( dwTimeStamps ) );
      ICMPOut.Add( 'Відповідь мітки часу    : '  + IntToStr( dwTimeStampReps )
);
      ICMPOut.Add( 'Запит маски адрес: '  + IntToStr( dwAddrMasks ) );
      ICMPOut.Add( 'Відповідь маски адрес   : '  + IntToStr( dwAddrReps ) );
    end;
  end
  else
    IcmpIn.Add( SysErrorMessage( ErrorCode ) );
  Dispose( ICMPStats );
end;

procedure Get_RecentDestIPs( List: TStrings );
begin
  if Assigned( List ) then
    List.Assign( RecentIPs )
end;

initialization

  RecentIPs := TStringList.Create;

finalization

  RecentIPs.Free;

end.
```

ФАЙЛ TCP_IP.PAS - МОНІТОР TCP/IP З'ЄДНАНЬ

```
unit TCP_IP; // опис модулю

interface

{
Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
тема: Програмне забезпечення системи кібербезпеки оцінки стійкості комп'ютерних
мереж до загроз
Виконав: студент  4 курсу,  групи КБ-21-3СК
         Бойко Олександр Андрійович, 2024 рік
Керівник:  Коваленко О.В.
}

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, IP, M_API, Buttons;

type
  TForm2 = class(TForm)
    StaticText2: TStaticText;
    StaticText3: TStaticText;
    TCPMemo: TMemo;
    UDPMemo: TMemo;
    Timer1: TTimer;
    cbTimer: TCheckBox;
    btRTTI: TSpeedButton;
    SpeedButton1: TSpeedButton;
    edtRTTI: TEdit;
    procedure Timer1Timer(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure btRTTIClick(Sender: TObject);
    procedure cbRecentIPsClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
      procedure DOIpStuff;
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.dfm}

procedure TForm2.Timer1Timer(Sender: TObject);
begin
  if cbTimer.State = cbCHECKED then
 begin
   Timer1.Enabled := false;
   DoIPStuff;
   Timer1.Enabled := true;
 end;
```

```
end;

procedure TForm2.DOIpStuff;
begin

  Get_TCPTable( TCPMemo.Lines );
  Get_UDPTable( UDPMemo.Lines );

end;

procedure TForm2.SpeedButton1Click(Sender: TObject);
begin
  Speedbutton1.Enabled := false;
  DoIPStuff;
  Speedbutton1.Enabled := true;
end;

procedure TForm2.btRTTIClick(Sender: TObject);
var
  IPadr         : dword;
  Rtt, HopCount : longint;
  Res           : integer;
begin
  btRTTI.Enabled := false;
  Screen.Cursor := crHOURGLASS;
  IPadr := Str2IPAddr( edtRTTI.Text );
  Res := Get_RTTAndHopCount( IPadr, 128, RTT, HopCount );
  if Res = NO_ERROR then
    ShowMessage( ' Час запиту '
      + inttostr( rtt ) + ' ms, '
      + inttostr( HopCount )
      + ' hops to : ' + edtRTTI.Text
      )
  else
    ShowMessage( ' Відбулася помилка:' + #13
                 + ICMPErr2Str( Res ) ) ;
  btRTTI.Enabled := true;
  Screen.Cursor := crDEFAULT;

end;

procedure TForm2.cbRecentIPsClick(Sender: TObject);
begin
 edtRTTI.Text := cbRecentIPs.Items[cbRecentIPs.ItemIndex];
end;

procedure TForm2.FormCreate(Sender: TObject);
begin

  if LoadIpHlp then
  begin
      DOIpStuff;
      Timer1.Enabled := true;
  end
  else
      ShowMessage( ' Інтернет помічник DLL не є доступним, або не
підтримується') ;
end;

end.
```