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 формування фільтрів від фішингу в мережі Internet”

Виконав здобувач вищої освіти
IV курсу, групи КБ-20
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Кондрашенко І.С.
«____» _____ 2024 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Смірнов С.А.
«____» _____ 2024 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет
Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Освітній ступінь *бакалавр*
Галузь знань ._12 _____*"*Інформаційні технології*"*_____
Спеціальність *125 "*Кібербезпека*"*
Освітньо-професійна (освітньо-наукова) програма _*"*Кібербезпека*"*_

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

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

Кондрашенку Іллі Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet*

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

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

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

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet*

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

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи кібербезпеки в промислову експлуатацію.
6. Висновки

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

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

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

КАЛЕНДАРНИЙ ПЛАН

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

Дата видачі завдання
« 17 » січня 2024 р.

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

Завдання прийнято до виконання
« 17 » січня 2024 р.

Підпис здобувача
_____ Кондрашенко І.С.
(прізвище та ініціали)

АНОТАЦІЯ

Кондрашенко І.С. Програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet. 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 формування фільтрів від фішингу в мережі Internet.

Метою розробки є програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

Результат роботи – програмна реалізація системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

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 10.4, XML, XUL.

Ключові слова: кібербезпека, фішинг

ABSTRACT

Kondrashenko I.S. Software of the cyber security system of creating filters against phishing on the Internet. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this graduation thesis for the first (bachelor) level of higher education, software is developed, which is intended for the cyber security system of forming filters against phishing on the Internet.

The goal of the development is the software of the cyber security system of forming filters against phishing on the Internet.

The result of the work is the software implementation of the cyber security system of creating filters against phishing on the Internet.

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 10.4, XML, XUL environment.

Keywords: cyber security, phishing

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

DHCP – протокол, що використовується для динамічного розподілу IP-адрес.

DNS – розподілена служба Інтернет, використовувана для зіставлення логічних (доменних) імен і IP-адрес. DNS використовується для забезпечення можливості роботи зі зрозумілими й іменами, що легко запам'ятовуються, замість IP-адрес у числовому форматі

ICMP – протокол

IDS – система виявлення атак

IP – адресний протокол

LAN – локальна мережа

NAT – трансляція IP-адрес із одного адресного простору в IP-адреси іншого адресного простору

Proxy – програма-посередник, що транслює запити різних протоколів із локальної мережі в зовнішню мережу

TCP – протокол обміну даними на транспортному рівні

UDP – протокол

URL – уніфікований покажчик інформаційного ресурсу (стандартизований рядок символів, що вказує місцезнаходження документа в мережі Інтернет)

ЕЦП – електронний цифровий підпис

КД – ключова дискета

НСД – несанкціонований доступ

ОС – операційна система

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ВСТУП

Актуальність теми. В Україні кількість тих, хто постійно користується інтернетом, уже перейшло оцінку в 20 мільйонів.

Для більшості комп'ютерів став другом і помічником, але мало хто знає, як захиститися від тих, хто по ту сторону монітора. Тільки торік інтернет-шахраї виманили в українців більше 100 мільйонів гривень.

Однією з розвинутих форм шахрайства в Інтернеті є безсумнівно фішинг. Фішинг – це різновид онлайн-шахрайства, який полягає в тому, що люди обманом змушують надати конфіденційну інформацію, як-от паролі чи номери кредитних карток, під виглядом надійного джерела. Фішинг може здійснюватися через електронну пошту, соціальні мережі або шкідливі веб-сайти.

Фішинг працює, надсилаючи повідомлення, які виглядають так, ніби вони надійшли від законної компанії чи веб-сайту. Фішингові повідомлення зазвичай містять посилання, яке спрямовує користувача на підроблений веб-сайт, який виглядає як справжній. Потім користувача просять ввести особисту інформацію, наприклад номер кредитної картки. Ця інформація потім використовується для викрадення особи людини або для шахрайського стягнення плати з її кредитної картки.

Більшість фішингових електронних листів надсилаються випадковим чином великій кількості одержувачів і покладаються на саму вагу чисел для успіху. (Чим більше електронних листів буде надіслано, тим більша ймовірність, що вони знайдуть жертву, яка їх відкриє.)

Однак існує також багато типів атак, відомих як фішинг, спрямованих на конкретні організації чи окремих осіб. Як і у випадку з ширшими фішинговими кампаніями, такі електронні листи можуть містити шкідливі посилання або вкладення.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

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

– Огляд існуючих систем формування фільтрів від фішингу в мережі Internet.

– Дослідження системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

– Програмна реалізація системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі формування фільтрів від фішингу в мережі Internet.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet, 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 захисту конфіденційних даних користувачів від фішинг атак у мережі Internet. Антифішинг (від анг.Anti-phishing) – це набір технологій, використовуваних для захисту від мережного шахрайства й розкрадання особистих даних, так званого фішингу. В основі антифішингу лежить механізм оповіщення інтернет-користувачів про влучення на підроблені веб-сайти, які спеціально створюються зловмисниками для збору конфіденційних даних (паролях доступу до онлайн-банків, платіжних систем і інших сервісів).

Антифішинг реалізується за допомогою двох взаємодоповнюючих технологій. Перша – це убудований в інтернет-браузер плагін (антифішинговий фільтр), що попереджає користувача про влучення на підроблені або підозрілі сайти. Такі плагіни вже убудовані в багато популярних браузерів (наприклад, Microsoft Edge або Firefox 2.0 і вище) або комплексні продукти по захисту ПК. Друга – це фільтрація фішингових листів, що розсилаються зловмисниками для заманювання жертв на підроблені веб-сайти, за допомогою персонального або серверного антиспаму.

1.2 Область застосування

Інформаційний захист від вторгнення в епоху W5 (Web 5.0) математично описується складними теоріями й формулами, а на практиці реалізується ще більш складними технічними засобами. Однак, чим більш важкою для розуміння є система, тим більш складними й небезпечними в ній будуть ставати найменші помилки.

Сучасна інформаційна безпека вимагає редукційного підходу, заснованого на поясненні складних явищ законами, властивими більше простим явищам. Відомість складного до простого й вищого до нижчого добре відбито у філософському принципі «бритви Оккама», що, нарівні з «законами Мерфі» і «теорією прогресивного хаосу» всі частіше застосовується для опису процесів забезпечення мережної безпеки в епоху W5.

Методологічний принцип «бритви Оккама» іноді виражають так: «те, що можна пояснити за допомогою меншого, не слід виражати за допомогою більшого». Цей принцип добре підходить для характеристик ущербності існуючих тактик забезпечення мережної безпеки. Грамотно виставлені настроювання фаєрволла (у тому числі й апаратного), постійний обновлюваний антивірус і щодня встановлювані відновлення не закриють головної уразливості – діри в голові користувача. За ордами хакерів системні адміністратори часто намагаються сховати власну лінь і не можливість забезпечити грамотний захист на рівні ядра користувача.

Серед системних адміністраторів поширені дві збиткові моделі поводження з користувачами: модель «дурного» користувача й модель користувача «який розвивається сам». «Дурний» знає як включити комп'ютер і працювати з основними додатками. Завдання забезпечення безпеки лягає повністю на плечі адміністратора. «Той що розвивається сам» «розуміє» які дії дозволяють вишиковувати ефективну оборону. Йому досить скинути файл «Як правильно відкривати посилання» і проблеми заочно вирішені.

Як розуміють всі хакери, обидві стратегії помилкові. Знімаючи відповідальність із людини за її дії, ви, тим самим, провокуєте його на здійснення необдуманих дій. Намагаючись скинути свої обов'язки на інших – підкидаєте зайві ключі до дверей у вашу систему. Устояні принципи забезпечення безпеки не захищають «складних» і «досвідчених» користувачів від фішингових і спуфінгових атак, а так само від методів соціальної інженерії. Антифішингова й

антиспуфінгова політика безпеки вибудовується на програмному рівні, при взаємодії як адміністратора, так і користувача.

Щодня інфікується більше шести тисяч веб-сторінок. Разом з ними росте й кількість інфікованих комп'ютерів. Високий ступінь зараження обумовлюється низьким рівнем превентивного захисту в сучасних браузерах. Використовувати, як єдину перешкоду, убудовані в браузер елементи антифішингу й сторонній антивірус – найкоротший шлях до машин-зомбі. На жаль, навіть серед висококваліфікованих фахівців існує думка, що правильно настроєний фаєрволл здатний вирішити більшість проблем, пов'язаних із зараженими веб-сторінками. На жаль, це не так. Зараженим може виявитися не тільки «потенційно небезпечний сайт», але й цілком легальний ресурс. Відвідуючи знайомий сайт, користувач може не звернути уваги на попередження про небезпеку або понизити рівень захисту заздалегідь. Забезпечити 100 % безпеку веб-серфінгу не можливо відповідно до фундаментальних законів всесвіту. Однак привести ймовірність зараження до величини, що нескінченно прагне до нуля, нам цілком під силу.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Розглянемо клас програм, що інспектують html код веб-сайтів у реальному часі.

LinkScanner

Програма від компанії Exploit Prevention Labs, відомих фахівців в області експлойтів і 0-денних вірусів. LinkScanner представлений як Online-Версією, так і demo free оффлайновим додатком. Повна версія програми обійдеться в $29.95.

Основний напрямок протидії: «нульові» віруси («не засвічені» антивірусними компаніями) і експлойти. LinkScanner стежить за трафіком на рівні сокетів і, відповідно до результатів аналізу, блокує або пропускає дані на комп'ютер клієнта. Розроблювачі затверджують, що унікальна технологія аналізу й глибоке знання технік написання шкідливого коду дозволяє надійно захищати комп'ютер незалежно від того, скільки часу буде потрібно для випуску патча або влучення вірусу в бази. Продукт також автоматично аналізує сторінки, знайдені в Google, і видозмінює сторінку результатів, додаючи позначки про небезпеку або надійність тих або інших сайтів.

Відомості про небезпечні сайти передаються в загальну базу, попереджаючи інших користувачів, що відвідують ті ж сайти. LinkScanner автоматично аналізує результати пошуку в Google і інших великих пошукових системах і розміщає прапорці поруч із безпечними сайтами. Всім обробленим сайтам виставляє відповідний ранг репутації. LinkScanner проводить ідентифікацію сайтів за номером ID, що виключає ймовірність переходу на підробку.

LinkScanner – це нічим не замутнена простота, життєво необхідна будь-якому не кваліфікованому користувачеві. При виявленні погрози в правому нижньому куті браузера спливає загрозливо-червоне віконце, а доступ на сторінку блокується. Можливість прорватися на заражену ділянку мережі з'являється тільки з вимиканням програми.

Websense Express

Посередині сьогоднішнього огляду Websense – це одна із провідних компаній інформаційної безпеки, що працює переважно на корпоративний ринок. Однак демократичні до нашої країни демпінгові ціни й наявність free версій популярних продуктів дозволяють нам ближче познайомитися із цією цікавою програмою.

Основна лінійка Websense представлена п'ятьома провідними продуктами. Вибрати з яких одне єдиний завдання не з легких – відмінності як по функціоналі (захист від всіх відомих і невідомих погроз), так і за ціною (200-1600 грн. за 1 ліцензійну копію) не істотні.

Для проби я зупинив свій вибір на Websense Web Security Suite («провідне рішення для безпеки організації проти всіх відомих веб-погроз»), оскільки в 30-денному триалі бонусом ішов Websense Enterprise («провідне рішення по фільтрації веб-трафіку»).

Розмір програми: Dynamic – Windows (29 MB), Full – Windows (160 MB). Підтримує як Windows Server, так і Linux. Будьте готові віддати додатку до 200 Мб оперативної пам'яті.

Яким образом здійснюється захист користувачів? Роботу з пошуку й класифікації інтернет-ресурсів загального характеру бере на себе спеціалізована група експертів Websense. На сьогоднішній день у базі Websense Master Database більше 20 мільйонів URL на більш ніж 50 мовах, розбитих на 90 тематичних категорій. База поповнюється щодня. За рахунок застосування технології зворотного зв'язку WebCatcher нерозпізнані сайти анонімно (і тільки за бажанням адміністратора) відправляються із клієнтських інсталяцій в експертний підрозділ Websense на категоризацію.

Websense проводить фільтрацію додатків на основі корпоративних політик безпеки, що опираються на тематичну базу, у яку включене майже 2 мільйони відомих додатків. Категорії безпеки описують безліч відомих шкідливих програм, і є можливість блокувати нерозпізнані додатки, запобігаючи запуску нових невідомих зразків шкідливого коду. Рішення передбачає додаткові елементи захисту, такі, як запобігання читання, зміни або видалення певних ключів регістра Windows або блокування змінних носіїв інформації. Інтеграція з міжмережним екраном Windows або мережною інфраструктурою Cisco NAC вирішує завдання контролю доступу додатків до корпоративної мережі й ізоляції робочих станцій, що не відповідають політиці безпеки.

У загальному виді ми маємо справу з корпоративним продуктом, що прекрасно виправдує свої високі характеристики. Однак користувачам поза бізнес-сектора він по багатьом параметрам просто марний.

McAfee SiteAdvisor

McAfee, на відміну від інших антивірусних компаній, антифішинговий захист сприймає не як формальність, «данину моді» (мол, куди ви з вашим захистом без сигнатур дінетеся), а як повноцінний розділ безпеки, що не уступає своїм більше відомим побратимам. McAfee відмовилися від популярної політики «а сьогодні ми продемонструємо додаткову функціональність вашого антивірусу» і випустили на ринок безкоштовний плагін McAfee SiteAdvisor, що працює з Internet Explorer і Firefox. Його функціональність виражається в тому, що в куті вашого браузера з'являється маленький «світлофор», що сигналізує вам, наскільки безпечний сайт, що ви збираєтеся відвідати. Як це працює? Заходите ви на Xakep.ru і бачите, що сайт позначений жовтим значком. «This is a security-related site that may contain hacking tools or examples of malicious code. It may have legitimate use for research purposes, but in some circumstances, may also be misused».

Також на сайті SiteAdvisor є цікава функція – можна просто проаналізувати зразкову безпеку того або іншого сайту й без цього плагіна: http://www.siteadvisor.com/analysis/.

Finjan SecureBrowsing

Ще один плагін, live оцінюючий код і репутацію сайту. Підтримка основних пошукових систем дозволяє бачити статус сайту ще до переходу по посиланню. Статус може бути зелений (все Ок!), жовтий (можливі нерозпізнані погрози) і червоний (є погроза). У випадку прямого переходу по посиланню спливе віконце відповідного кольору.

Ледве більше функціональним аналогом Finjan можна виділити CallingID Link Advisor.

Comodo Verification Engine

Побоюєтеся, що посилання в листі від вашого банку приведе на сайт шахраїв? Наведіть курсор миші на посилання й, якщо сайт справжній, навколо вікна браузера з'явиться звична нам зелена рамка з текстовим заголовком. У настроюваннях програми можна встановити програвання звукового файлу при наведенні мишки на логотип. Наприклад, розпачливий крик свині, що убивається. Плагін сполучимо із браузерами Internet Explorer, FireFox, Mozilla, AOL і Netscape. Крім того, даний плагін здатний визначати наявність або відсутність у сайту сертифіката безпеки, про що він інформує користувача спливаючим вікном.

Netcraft Toolbar

Панелька Netcraft вбудовується в браузер і відображає практично всі атрибути відвідуваних вами ресурсів: справжню адресу сайту, домен, IP-адресу, вік сайту протягом його мережного життя, назву організації власника ресурсу, ім'я сервера, адресу адміністратора доменної системи імен, і т.д. Чим це зручно? Можна перевірити «істинність» ресурсу, усього лише ознайомившись із розташуванням його хостера або за іншим даними, точно відомим про ресурс. Всі виявлені вами погрози будуть доступні в загальній базі для всіх користувачів.

Механізми Netcraft Toolbar відслідковують XSS (Cross Site Scripting) і різні підозрілі гіперпосилання, що містять спецсимволи, які здебільшого використовуються для того, щоб завуалювати щира адреса й обдурити користувача.

Реалізовано функцію примусового відображення інтерфейсних елементів навігації (панелі інструментів і адресного рядка) в усіх без винятку вікнах web-браузера. Це робиться для того, щоб запобігти можливому обману з боку «спливаючих» віконець, що намагаються сховати елементи навігації й своє щире походження (наприклад, шляхом приховання адресного рядка із вказівкою щирої мережної адреси).

FortiGuard Web Filtering

Fortinet забезпечуємо фільтрацію по «чорним» URL, блокування неприпустимих матеріалів і злобливих скриптів, включаючи Java Applets, Cookies і Active. Бази даних Fortinet містить інформацію про більш ніж 25 мільйони доменів і мільярдів web-сторінок, що гарантує захист від злобливих кодів під час перегляду сторінок в web-мережі. Система фільтрації web-умісту FortiGuard працює динамічно із системами FortiGate, забезпечуючи автоматичне відновлення по 56 категоріям. Послуга FortiGate допускає можливість конфігурування користувачами, додаючи до існуючих даних відомості про небажані сайти й фішинг web-сайтах.

FraudEliminator Pro

Додаток відображає вичерпну інформацію про відвідуваний сайт, включаючи координати хостинг-провайдеру й дату реєстрації доменного ім'я. Дієвість пропонованого захисту забезпечується за рахунок регулярного відновлення бази даних, що містить опису погроз і списки шахрайських сайтів, а також використання вдосконалених аналітичних алгоритмів для виявлення потенційної погрози.

Користувачі можуть вносити власний вклад у боротьбу із шахрайством, відправляючи звіти про виявлені погрози в центральну базу даних FraudEliminator. Для цього досить нажати на кнопку «Fraud Report» в інтерфейсі web-браузера. FraudEliminator працює під керуванням всіх видів Windows, для браузерів Internet Explorer і FireFox. Стандартна версія FraudEliminator поширюється безкоштовно. Версія FraudEliminator Pro (яка припускає більше часте завантаження відновлень) обійдеться вам в 19.99 доларів.

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 WebVieW5 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 WebVieW5 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.

XML

Розши́рювана мо́ва розмі́тки (англ. Extensible Markup Language, скорочено XML) – запропонований консорціумом World Wide Web (W3C) стандарт

побудови мов розмітки ієрархічно структурованих даних для обміну між різними додатками, зокрема, через Інтернет. Є спрощеною підмножиною мови розмітки SGML. XML документ складається із текстових знаків, і придатний до читання людиною.

Стандарт XML визначає набір базових лексичних та синтаксичних правил для побудови мови описання інформації шляхом застосування простих тегів. Цей формат достатньо гнучкий для того, аби бути придатним для застосування в різних галузях. Іншими словами, запропонований стандарт визначає метамову, на основі якої, шляхом запровадження обмежень на структуру та зміст документів визначаються специфічні, предметно-орієнтовані мови розмітки даних. Ці обмеження описуються мовами схем (англ. Schema), таких як DTD, RELAX NG або XML Schema. Прикладами мов, основаних на XML є: RSS, MathML, GraphML, XHTML,Scalable Vector Graphics, і також XML Schema.

Основні поняття

Коректність

Коректний документ (англ. well-formed document) відповідає всім синтаксичним правилам XML. Документ, що не є коректним, не може називатись XML-документом. Сумісний синтаксичний аналізатор (англ. Conforming parser) не повинен обробляти такі документи. Зокрема, коректний XML документ має:

– Документ має лише один елемент в корені.

– Непорожні елементи розмічено початковим та кінцевим тегами (наприклад, <пункт>Пункт 1</пункт>). Порожні елементи можуть помічатись «закритим» тегом, наприклад <IAmEmpty />. Така пара еквівалентна <IAmEmpty></IAmEmpty>.

– Один елемент не може мати декілька атрибутів з однаковим іменем. Значення атрибутів знаходяться або в одинарних ('), або у подвійних (") лапках.

– Теги можуть бути вкладені, але, не можуть перекриватись. Кожен некореневий елемент мусить повністю знаходитись в іншому елементі.

– Фактичне та задекларованe кодування (англ. character encoding) документа мають збігатись. Кодування може бути задекларовано ззовні, як в заголовку «Content-Type» при передачі по протоколу HTTP, або в самому документі використанням явної розмітки на самому початку документа. Якщо така декларація відсутня, обирається кодування Юнікод, як вказано в перших байтах документа позначених Byte-order mark. Якщо і ця позначка відсутня, обирається кодування UTF-8.

Валідність

Документ називається валідним (англ. valid), якщо він є коректним, містить посилання на граматичні правила, та повністю відповідає обмеженням, вказаним у цих правилах (DTD або XML Schema або іншому подібному документі).

Синтаксичний аналізатор

Синтаксичним аналізатором (часто, парсер від англ. parser) називається програма або компонента, що читає XML-документ, проводить синтаксичний аналіз, та відтворює його структуру. Якщо синтаксичний аналізатор перевіряє документ на валідність, то такий аналізатор називають валідуючим (англ. validating).

Назви елементів чутливі до регістра літер. Наприклад, наступна пара елементів коректна:

<Step> … </Step>

в той час як ця, – ні:

<Step> … </step>

Правильний вибір імен для XML елементів підкреслюватиме значення даних в створеній мові розмітки. Це сприятиме полегшенню роботи людей з такими документами, зберігаючи можливості для комп'ютерної обробки даних. Вибір значущих імен передає семантику елементів та атрибутів для людини, без посилання на зовнішню документацію. Однак, це може призвести до надмірності розмітки, що ускладнює редагування і збільшує розмір файлів.

Структура

XML-документи мають як фізичну, так і логічну структуру.

Фізична структура

– Сутності (англ. Entity). Головною сутністю є зміст документа. Інші можливі сутності вказуються із допомогою

– Посилання на сутності (&назва. в самому документі, та, наприклад %назва. у визначені його типу) можуть слугувати в якості як позначення спеціальних символів, посилань на спеціальні символи (вказуючи коди символів&#десяткове., або &#хшістнадцяткове.) або окремих документів чи фрагментів тексту.

– XML декларація, в ній вказується версія XML, кодування, та інша допоміжна інформація.

– Декларація типу документа може застосовуватись для того, аби додавати нові типи сутностей, та визначати логічну структуру документа.

Логічна структура

XML документ має ієрархічну логічну структуру, і може представлятись у вигляді дерева. Вузлами цього дерева можуть бути:

– Елементи, фізична структура яких складається із:

– коректної пари відкриваючого та закриваючого тегів (<Назва-тега>) та (</Назва-тега>), або тега порожнього елемента (<Назва-тега />),

– Атрибути, що мають вигляд пар ключ-значення (назва атрибута="значення атрибута") і знаходяться або у відкриваючому, або у порожньому тезі (подібно до метаданих),

– Вказівки щодо обробки документа (англ. Processing Instruction) (<?Обробник параметр ?>)

– Коментарі (<!-– Текст коментаря -->)

– Текст, або у вигляді простого тексту, або фрагментів CDATA (<![CDATA[довільний текст]]>).

XML-документ повинен мати лише один кореневий елемент. Решта елементів є піделементами цього кореневого елемента.

Деякі веб-браузери здатні безпосередньо відображати XML-документи. Це може досягатись шляхом застосування таблиці стилів (англ. Stylesheet). Вказані у таблиці стилів операції можуть призводити до перетворення XML-документа в інший, відмінний від XML формат.

XUL

XUL (XML User Interface Language) – мова розмітки для створення графічних інтерфейсів користувача, основана на XML. XUL поширюється та розробляється в межах проекту Mozilla.

XUL розроблено для створення інтерфейсів у таких програмах, як браузери, поштовий клієнт, програма-календар, редактор HTML NVU, медіа-програвач. XUL можна ефективно використовувати для створення будь-яких програм та розширень, пов'язаних з роботою з веб-ресурсами і не тільки.

XUL, як і HTML, описує інтерфейси за допомогою мови розмітки і дозволяє задавати зовнішній вигляд програми через CSS та визначати поведінку за допомогою JavaScript. Однак, на відміну від HTML, XUL дозволяє створювати динаміку користувацького інтерфейсу набагато швидше та зручніше. XUL надає багатий набір компонентів, з можливим побудувати інтерфейс розширення чи програми.

Знання XUL – основа для створення додатків для продуктів Mozilla, оскільки більша частина їх інтерфейсу написана на XUL.

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 формування фільтрів від фішингу в мережі Internet.

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

Запозичене слово фішинг (phishing) утворено від англійського password – пароль і fishing – рибний лов, вивудження. Ціль цього виду інтернет-шахрайства – облудне відведення користувача на підроблений сайт для того, щоб надалі украсти його особисту інформацію (логіни, паролі, адреси електронної пошти й т.п.) або, наприклад, заразити комп'ютер користувача, переспрямованого на підроблений сайт трояном. Заражений комп'ютер може активно використовуватися в ботнет-мережах для розсилання спаму, організації DDoS-атак, а так само для збору даних об користувача й відправлення їхньому зловмисникові. Спектр застосування «вивудженої» у користувача інформації досить широкий. В останні роки активне поширення Інтернет-шахрайств привело до формування т.зв. чорних ринків зі своїми замовниками й виконавцями. Останні звіти аналітиків говорять про наявність у сучасному світі складної вірусної "екосистеми". Так, основна маса вірусів і троянських програм створювалася в 2013 році з метою наступного продажу. Причому, якщо по кількості створюваного шкідливого ПЗ світовим лідером в 2013 році став Китай, то по складності й "інноваційності" програм на першому місці виявилися російські хакери й розробники вірусів.

Традиційні методи протидії

Виниклі з появою фішингу погрози зажадали впровадження адекватних мір захисту. У рамках даної статті будуть розглянуті як уже широко розповсюджені способи протидії фішингу, так і нові ефективні методи.

Поділ це досить умовний: до традиційного віднесемо добре відомі (у тому числі й самих зловмисниках) способи протидії фішингу й проаналізуємо їхню ефективність.

Унікальний дизайн сайту

Суть цього методу така: клієнт, наприклад, банку при укладанні договору вибирає одне із запропонованих зображень. Надалі при вході на сайт банку йому буде показуватися саме це зображення. У випадку якщо користувач його не бачить або бачить інше, він повинен покинути підроблений сайт і негайно сповістити про це службу безпеки. Передбачається, що зловмисники, що не були присутнім під час підписання договору, апріорі не зможуть угадати правильне зображення й обдурити клієнта.

Інший варіант – видати клієнтові фальшиве попередження про витікання терміну дії його зображення й запропонувати вибрати нове ...

Одноразові паролі

Класичні паролі є багаторазовими: користувач уводить той самий пароль щораз при проходженні процедури автентифікації, не міняючи його часом роками. Перехоплений зловмисником цей пароль може неодноразово використовуватися їм без ведена хазяїна.

На відміну від класичного одноразовий пароль використовуються тільки один раз. При кожному запиті на надання доступу користувач уводить новий пароль.

Однобічна автентифікація

Використання протоколу безпечних з'єднань SSL (Secure Sockets Layer) забезпечує захищений обмін даними між веб-сервером і користувачами. Даний протокол, розроблений в 1996 році компанією Netscape, на сьогодні став одним із самих популярних методів забезпечення захищеного обміну даними в мережі Інтернет. Протокол SSL, що використовує асиметричний криптографічний алгоритм RSA інтегрований у більшість браузерів і веб-серверів, а для реалізації захищеного з'єднання з використанням російської криптографії буде потрібно додаткове програмне забезпечення, як на сервері, так і на кожному клієнтському робочому місці.

URL-фільтрація

Виявленням фішерских сайтів і внесенням їх у чорні аркуші займаються багато компаній від виробників антивірусних рішень до банків, платіжних систем і правоохоронних органів. Зокрема, створюються спеціальні організації для боротьби з фішерами, такі як Anti Phishing Work Group (APWG).

Спільні заходи зацікавлених сторін у тісному співробітництві з реєстраторами й хостинговими компаніями дозволяють оперативно закривати підроблені сайти. Відповідно до звіту самої APWG за першу половину 2013 року було виявлено 47,324 фішингових сайтів. Спільні зусилля спрямовані на максимально швидке відновлення чорних списків і блокування роботи сайтів зловмисників. Не можна не відзначити певні успіхи в цьому напрямку – середній час життя фішерского сайту становить усього 49.5 годин. У тому же звіті, однак, наведені й середні втрати користувачів і компаній у результаті роботи фішерского сайту, вони становлять не менш $300 у годину. Нескладні множення дозволяють зробити вивід про високу прибутковість цього виду чорного бізнесу.

Цілком імовірно, що далеко не всі виробники засобів антивірусного захисту можуть похвастатися настільки високою оперативністю у відновленні баз, до того ж багато користувачів не використовують ніяких засобів захисту на своїх комп'ютерах або водять номера кредитних карт і іншу конфіденційну інформацію з випадкових робочих місць. Ну й, нарешті, не слід забувати, що наведені дані є усередненими, що означає, що реальні атаки можуть наносити істотно більшу втрату конкретному користувачеві або компанії, можливо, ставлячи їх на грань банкрутства.

Описані вище методи протидії фішинговим атакам, особливо застосовувані спільно, дозволяють підвищити безпека, однак у кожному разі залишаються підданими тим або іншим видам атак і при відомій наполегливості зловмисника не зможуть захистити гроші й дані користувача.

Нові методи протидії

Описані вище методи протидії фішинговим атакам, особливо застосовувані спільно, дозволяють підвищити безпеку, однак у кожному разі залишаються підданими тим або іншим видам атак і при відомій наполегливості зловмисника не зможуть захистити гроші й дані користувача.

Уже розглянуті способи володіють одним загальним недоліком – застосовувані міри легко зводяться на немає недбалістю або не уважністю користувача. Згода прийняти сертифікат, підписаний недовіреним УЦ або перехід по посиланню зі спамового листа на підроблений сайт взагалі без установлення захищеного SSL-з'єднання, несвоєчасно оновлені бази персонального антивірусу, неправильно настроєний локальний файрвол, уведені три одноразові паролі, що підряд ідуть, на фішерскому сайті, та ж згода вибрати нову картинку для сайту або ігнорування повідомлення про неможливість завантажити її – все це й багато чого іншого в остаточному підсумку може привести й, на жаль, приводить до втрати грошей.

Як правило, у договорах, що містяться з користувачами платіжних систем або клієнтами банків, вся відповідальність за недбалість у діях покладає на самих користувачів. Спроба в такий спосіб убезпечити себе юридично вже приводить до відповідних дій клієнтів. Не рідкістю стають судові процеси, у яких адвокати доводять, що при наявній системі автентифікації забезпечити схоронність даних клієнт був не в змозі, про що в момент укладання договору співробітники банку не могли не знати, а виходить, і покладання відповідальності було не правочинним. З іншого боку, втрата грошей користувачами, нехай навіть зі своєї вини в кожному разі негативно позначається на репутації банку в їхніх очах, а при масових втратах – так само й в очах ще не постраждалих клієнтів.

Розглянемо методи боротьби з фішинговими атаками, що представляються найбільш ефективними на сьогоднішній день і позбавлені описаних вище недоліків.

Пропаганда культури поводження

Як ми вже з'ясували, сама слабка ланка в сучасних системах захисту взагалі й від фішингових атак зокрема це людина. Саме тому, найважливіша увага компанії, стурбованої потенційними втратами грошей, варто звернути на пропаганду основ інформаційної безпеки серед своїх співробітників і клієнтів.

Приведемо деякі із правил, розповідям про які варто приділити ледве більше уваги, ніж просте згадування на передостанній сторінці багатосторінкового договору на обслуговування.

– Не довіряйте посиланням в електронних листах.

– Не відправляйте особисту інформацію у відповідь на прохання по електронній пошті.

– Перевіряйте правильність URL-адреси.

– Уводите адресу в рядок браузера самостійно.

– Використовуйте тільки телефонні номери, зазначені на кредитній карті або в договорі.

– Не відкривайте невідомих вкладень у листах.

Можливо, комусь цей список здасться елементарним, але якщо говорити про користувачів у цілому, то загальноприйняте виконання навіть таких простих правил здатно істотно зменшити доходи фішерів, потенційно зробивши даний бізнес менш рентабельним, а виходить, менш привабливим. Правила дорожнього руху, особливо для пішоходів, теж не можна назвати надскладними, але загальне їхнє знання й більш-менш виконання щорічно рятує чимало життів.

Зрозуміло, що в масштабах держави пропаганда правил комп'ютерної безпеки не є настільки високопріоритетним завданням, і саме тому основна надія тут на керівників організацією. Адже саме їхньому бізнесу і їхнім грошам прямо через співробітників або опосередковано через клієнтів загрожують фішери.

Протидія фішингу в корпоративному середовищі

Основним пріоритетом при побудові захисту від фішингу в рамках компанії варто зробити максимальний відхід від людського фактора. Саме тому

найбільш перспективними представляються шлюзові рішення, які на відміну від персональних продуктів не тільки знижують навантаження на робочі станції й спрощують адміністрування, але й дозволяють закрити всю комп'ютерну мережу організації єдиним надійним «парасолькою».

Сучасні ефективні шлюзові рішення борються з фішерскими атаками на чотирьох рівнях:

– Рівень доступу. Основа антифішингової безпеки – це вже розглянута URL-фільтрація (заборона доступу до сайтів з категорії фішингових), що, незважаючи на свою низьку ефективність «у бої один на один», доповнена рядом технологій, що дозволяють відрізнити посилання на фішерский сайт від легітимного, здатна вчинити опір фішерам.

– Рівень активного контенту. Кращі в цьому класі рішення реалізують фільтрацію 100% HTML-коду й впроваджених об'єктів на наявність шкідливого коду, у тому числі схованих каскадних переадресацій, коли тіло трояна збирається з невеликих нешкідливих окремо фрагментів і тому важко детектуємих частин на декількох сайтах, по яких користувача прозоро для нього «прокидають». Завдяки ефективному очищенню трафіку реалізується захист користувача від потенційних небажаних наслідків у випадку переходу, що відбувся все-таки, на фішерский сайт.

– Рівень комунікацій. У тому випадку, коли метою залучення користувача на підроблений сайт є зараження його комп'ютера яким-небудь шкідливим кодом, ще одним рівнем блокування захисту бути запобігання передачі приватних даних, зібраних ботами. Незважаючи на велику кількість і величезну розмаїтість видів самих троянів і ботів, існує всього лише кілька десятків комунікаційних протоколів, по яких вони взаємодіють зі своїм керуючим центром. Таким чином, блокування таких комунікацій найбільше ефективно здійснюється по сигнатурах протоколів, а не самого шкідливого коду.

– Рівень передачі даних. Отримавші широке поширення в останні роки DLP-рішення (Data Leak Prevention) дозволяють у рамках компанії побудувати ще

один рубіж оборони у вигляді контролю потенційних каналів витоку даних. Такі рішення можуть допомогти у виявленні й запобіганні відправлення шкідливим кодом, наприклад, номера кредитних карт або іншу конфіденційну інформацію.

Мабуть, єдиним слабким місцем таких систем може виявитися неможливість захисту мобільних співробітників, що працюють віддалено по відкритих каналах зв'язку. Для рішення даної проблеми в якості одного з варіантів можна запропонувати проксірование, тобто вихід в інтернет з ноутбуків компанії тільки через головний офіс. Таке ж рішення для спрощення адміністрування й зниження фінансових витрат можна запропонувати й для філій. Варто відзначити, що провідні гравці цього сегмента ринку готові запропонувати своїм клієнтам самим відчути себе в ролі таких філій, пропонуючи не здобувати й супроводжувати їхні продукти, а орендувати для фільтрації поштового й веб-трафіку обчислювальні потужності самих виробників.

Протидія фішингу як конкурентна перевага

Крім турботи про власну конфіденційну інформацію й захист співробітників у філіях і офісах багато компаній піклуються й про своїх клієнтів. Ділова репутація часом коштує дорожче, ніж витрати на побудову дійсно безпечної системи по автентифікації користувачів.

Уже розглянутий раніше протокол SSL має можливість проводити двосторонню автентифікацію, коли перевіряється валідність не тільки сервера, але й самого користувача. Для цього клієнтам, наприклад, банку необхідно одержати цифровий сертифікат. Зробити це можна, як правило, при укладанні договору на обслуговування або пізніше в будь-який час.

Відмова від паролів при доступі користувачів до рахунків серйозно ускладнює життя фішерам. Використання цифрових сертифікатів на стороні сервера й клієнта знімає проблему атаки «людина усередині» і робить прослуховування й перехоплення трафіку марними.

Основою безпеки при використанні цифрових сертифікатів є схоронність закритого ключа. Організація має набагато більше, ніж рядовий користувач,

фінансових і технічних можливостей по надійному захисті закритого ключа, використовуваного для автентифікації її веб-сайта. Зберігання клієнтом свого закритого ключа в реєстрі операційній системі або на жорсткому диску не є безпечним. У випадку зараження комп'ютера користувача ці дані легко можуть бути викрадені шкідливим програмним забезпеченням, і захист закритого ключа паролем не буде надійною гарантією схоронності грошей користувача. Застосовувані на практиці паролі рідко перевищують 8 символів і найчастіше якщо й не є осмисленим словом, то складаються тільки із прописних букв латинського алфавіту.

Самим надійним способом зберігання закритих ключів користувача на сьогоднішній день є використання криптографічних токенів. На відміну від інших зовнішніх носіїв (наприклад, тих же USB-флеш) при використанні токенів немає необхідності в копіюванні секретної інформації в оперативну пам'ять комп'ютера при проведенні операції автентифікації, тому що подібні пристроїв не тільки надійно зберігають закриті ключі, але й апаратно виконують необхідні криптографічні обчислення. При цьому важливо, що скористатися токеном може тільки його власник, що знає пароль від нього (пін-код).

Багато банків уже сьогодні пропонують своїм клієнтам можливість автентифікації не тільки по одноразових паролях, але й з використанням цифрових сертифікатів. Однак поки не так широко поширене використання апаратних криптографічних токенів для підвищення безпеки зберігання закритих ключів, проте й таких банків стає з кожним роком усе більше, адже даний механізм на сьогодні є без сумніву одним із самих надійних для автентифікації при здійсненні він-лайн транзакцій.

Висновок

Наявні сьогодні як добре й давно відомі, так і ефективні засоби, що з'явилися в останні роки нові, боротьби з фішинговими атаками, здатні звести ризики втрати конфіденційної інформації користувачів до мінімальних значень. У силу особливості реалізації он-лайн сервісів основна робота із забезпечення

безпеки даних користувачів лягає на власників веб-сайтів. Застосовувані ними рішення з кожним роком стають усе більше надійними, особливо в сфері телебанкигу. Однак, до впровадження тієї ж апаратної автентифікації користувачів у таких сервісах як інтернет-аукціони, веб-почта або соціальні мережі поки ще далеко, а виходить, цілком імовірним представляється все більша концентрація фішерів саме на них.

Кажуть, що надійний замок зупинить зловмисника не стільки складністю його злому, скільки можливістю вибору злодієм сусідніх дверей з більше простим механізмом закриття. Саме тому для банків, що піклуються про своїх клієнтів, очевидним є необхідність застосування саме надійних і ефективних методів боротьби з фішерами.

3.2 Розробка структурної схеми

На рисунку 3.1 зображена структурна схема, розроблена під час бакалаврського проектування, системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

Фільтр фішингу – одна з можливостей браузера програми захисту конфіденційних даних користувачів від фішинг атак у мережі Internet, що дозволяє виявляти підроблені веб-сайти. Фільтр фішингу запускається у фоновому режимі при перегляді веб-сторінок і використовує три способи захисту від фішингу:

– По-перше, він порівнює адреси відвіданих веб-сайтів зі списком сайтів, позначених у якості справжніх. Цей список зберігається на комп'ютері.

– По-друге, він допомагає аналізувати відвідані сайти для перевірки наявності ознак, характерних для шахрайських веб-сайтів.

– По-третє, за згодою користувача фільтр фішингу відправляє адреси деяких веб-сайтів На сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet для подальшої перевірки на присутність у списку виявлених підроблених веб-сайтів.

Якщо відвідуваний сайт перебуває в подібному списку, програма захисту конфіденційних даних користувачів від фішинг атак у мережі Internet відобразить попереджуючу веб-сторінку й повідомлення в адресному рядку. На попереджуючій веб-сторінці можна продовжити перегляд сайту або закрити його. Якщо веб-сайт містить ознаки підробленого сайту, але не доданий у список, програма захисту конфіденційних даних користувачів від фішинг атак у мережі Internet тільки повідомить користувача в адресному рядку про те, що сайт може бути підробленим. Використання фільтра фішингу регулюється угодою про обслуговування.

При установці програми захисту конфіденційних даних користувачів від фішинг атак у мережі Internet у перший раз фільтр фішингу тільки порівнює адреси відвіданих користувачем веб-сайтів зі списком справжніх веб-сайтів, збереженим на комп'ютері. Він також допомагає аналізувати відвідані веб-сайти для перевірки наявності ознак, характерних для шахрайських веб-сайтів. Ніякі відомості не відправляються на сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet без згоди користувача.

При першому відвідуванні веб-сайту, що не доданий у список справжніх веб-сайтів, буде відображений запит на автоматичну перевірку веб-сайтів. Якщо обрано цей параметр, фільтр фішингу буде відправляти на сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet певні адреси веб-сайтів для перевірки на наявність у часто обновлюваному списку виявлених підроблених сайтів і сповіщати користувача про підозрілі або виявлених шахрайських веб-сайти.

Фільтр фішингу блокує тільки сайти, засвідчені як підроблені рецензентами або співробітниками сторонніх постачальників даних. Фільтр фішингу також пропонує веб-систему відкликань і пропозицій, щоб допомогти користувачам і власникам веб-сайтів передавати звіти про помилки якнайшвидше. Ці звіти перевіряються, а виниклі помилки виправляються.

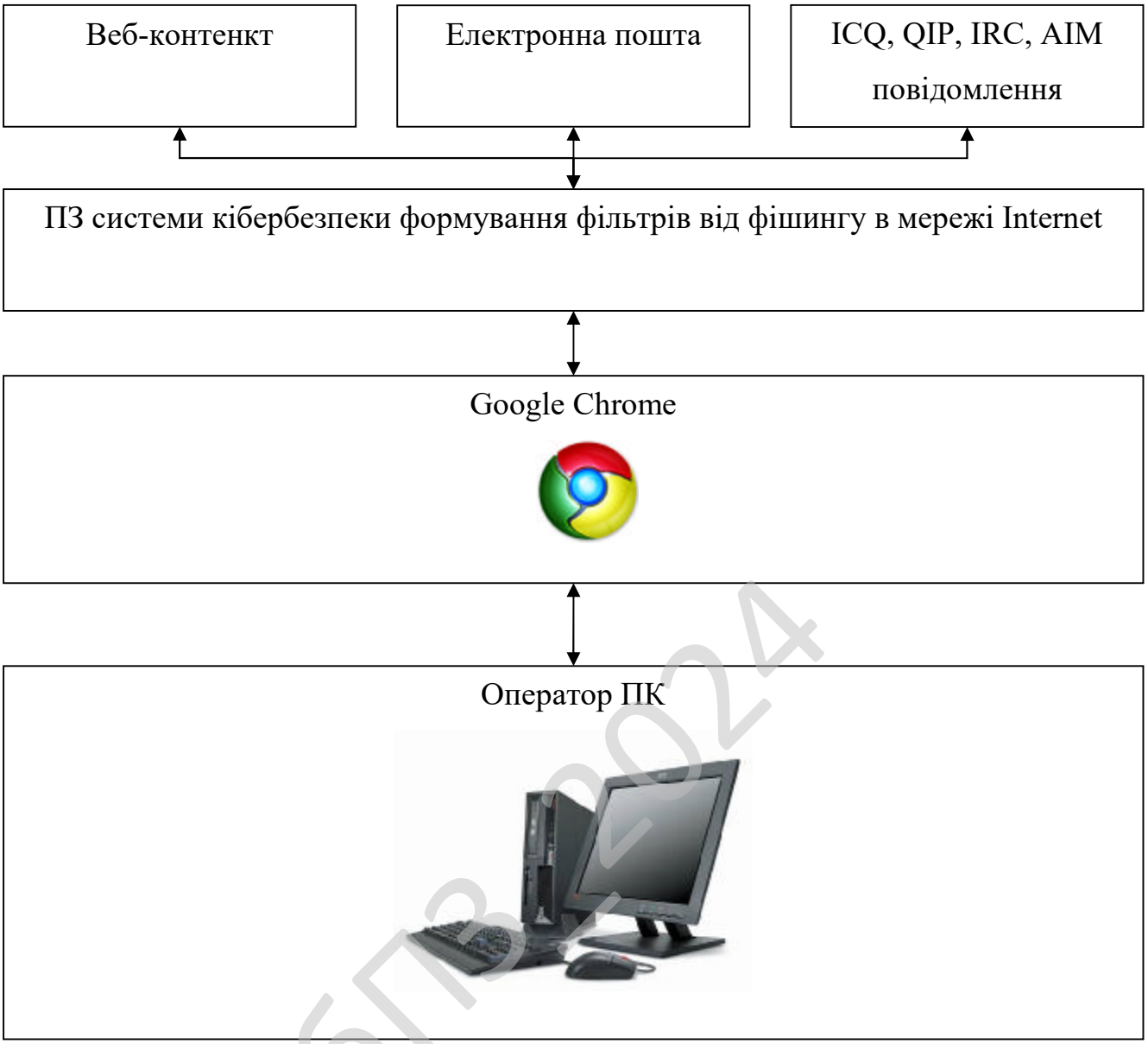


Рисунок 3.1 – Структурна схема системи

Користувач використовуючи Інтернет браузер Google Chrome одержує доступ до глобальної мережі Інтернет та через розроблене програмне забезпечення взаємодіє з Інтернетом (Web-контент), розмовляє за допомогою Інтернет пейджерів (IRC, ICQ, AIM), користується електронною поштою.

Google Chrome є релізом наступного покоління веб-браузерів, який має безліч нагород, він містить у собі безліч спеціальних можливостей, що дозволяють зробити веб-браузер і веб-контент доступним для всіх користувачів.

Програмне забезпечення є системою розширення (плагіном) можливостей Google Chrome надаючи користувачеві захист від фішингових атак.

Розробка системи антифішингу ґрунтується на потужній базі – браузері з розширеними можливостями. Так як найпоширеніший браузер Microsoft EDGE поширюється із закритим вихідним кодом, а також має сховану структуру, що негативно впливає при написанні додаткових програм і плагинів на його основі й проаналізувавши існуючі на даний момент браузери, і їхні розподілені системи захисту, я зупинив свій вибір на браузері Google Chrome.

У нього велика кількість переваг, головна з яких – надання великої кількості підпрограм що дозволяють одержати доступ до пошти, використанню Інтернет-пейджерів різних систем.

Google Chrome розповсюджується із частково відкритим кодом і має розширений набір засобів (Software Development Kit) для написання й розповсюдження додатків на його основі.

Як показано на рисунку 3.1, система антифішингу робить взаємодію через Software Development Kit і заснована на браузері Google Chrome.

3.3 Розробка функціональної схеми

Функціональна схема системи кібербезпеки формування фільтрів від фішингу в мережі Internet зображена на рисунку 3.2. На ній визначена можливість прослідкувати за шляхами проходження функціональних сигналів від одного функціонального блоку до іншого та побачити рівні надходження даннях з мережі та міри їхньої обробки.

При використанні фільтра фішингу для перевірки веб-сайтів автоматично або вручну, адреса відвідуваного веб-сайту буде відправлена на сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet разом з деякими загальними відомостями про комп'ютер, наприклад IP-адреса комп'ютера, тип браузера й номер версії фільтра фішингу. Щоб захистити конфіденційність користувачів,

відомості про адресу, що відправляються На сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet, шифруються по протоколі SSL і обмежуються доменом і адресою відвідуваного веб-сайта. Інші відомості, які можуть бути пов'язані з веб-адресою, наприклад умови пошуку, інформація, уведена у форми, або файли cookie, не відправляються.

Розглянемо загальні можливості розробленого програмного забезпечення зображені на схемі. Через розроблену систему антифішингу відбуваються наступні дії:

1. Вбудований фільтр сканування відвідуваних веб-сторінок – розширені можливості переходу на сторінки, які відвідувалися, з перевіркою на можливу переадресацію, а також база шаблонів відомих шкідливих кодів з можливістю редагування:

– Додавання, Редагування БД – можливість редагування й ручного додавання шаблона відомих шкідливих кодів.

– Локальна БД фішингових атак – шаблонів відомих шкідливих кодів.

2. Інтерактивна служба (одержання попередження про підозрілі вузли “www.”) – складається із трьох підрозділів, які дозволяють інтерактивно контролювати WEB контент, який попадає на машину користувача.

2.1. Реального часу на основі IE 7.0, Opera Software (GeoTrust) у глобальній мережі – з версії браузерів IE 7.0 і Opera 8.0 з'явився новий безкоштовний Інтернет сервіс, який надає доступ до всесвітньої бази перевірки веб-вузлів. В Інтернеті існує величезна кількість посилань на сайти при переході на які, відбувається запуск шкідливого програмного забезпечення й крадіжка особистої інформації, у даних випадках антивирустні програми неспроможні тому що використовуючи помилки ОС шкідливі програми одержують статус перевірених. За допомогою цього безкоштовного сервісу й розробленого в бакалаврському проекті можна значною мірою усунути можливість запуску такого шкідливого коду.

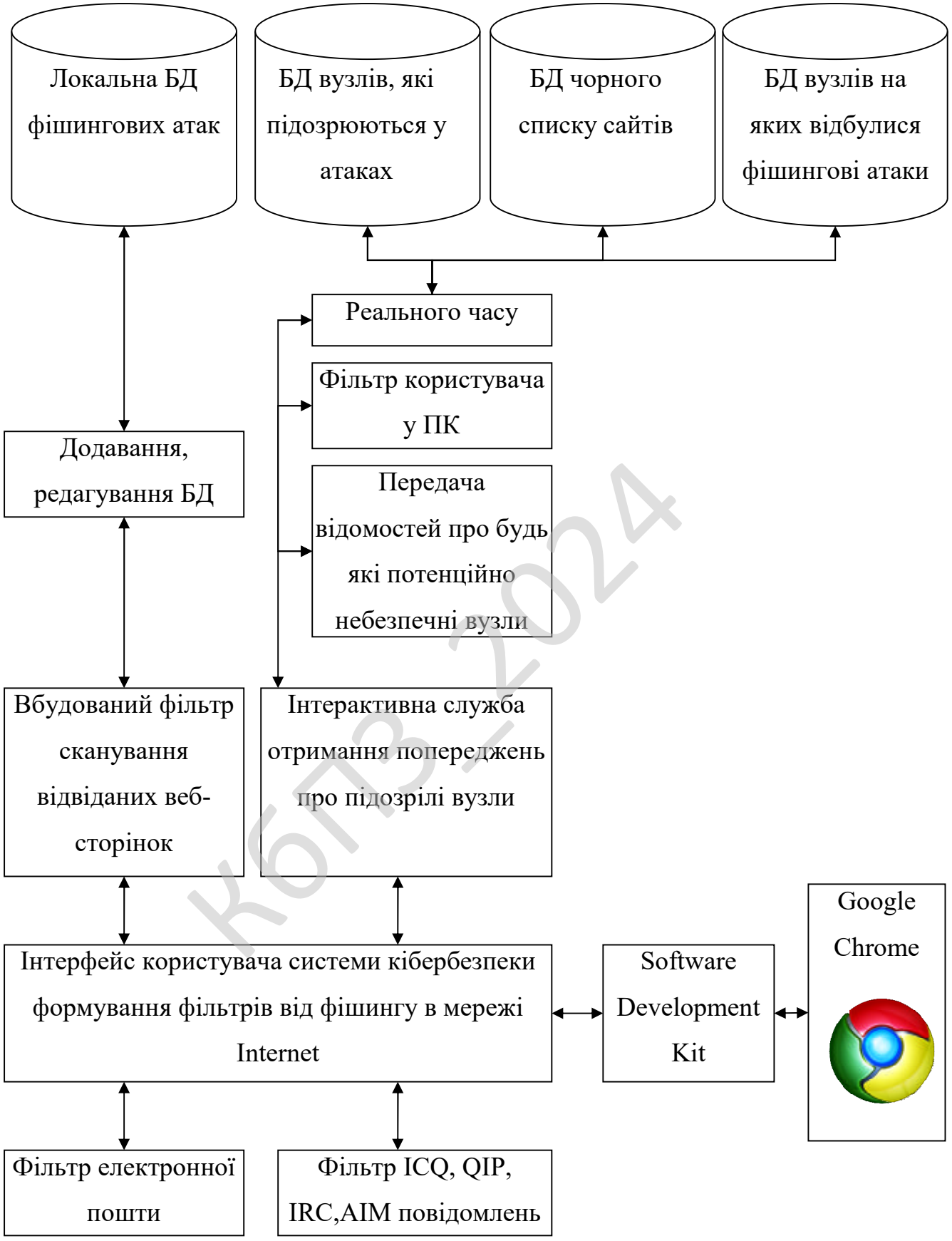


Рисунок 3.2 – Функціональна схема системи

Інтерактивний сервіс повертає наступні повідомлення:

– Веб-вузли, підозрювані в атаках.

– Веб-вузли, на яких фішинг-атаки вже відбувалися.

– Чорний список.

2.2 Фільтр користувача в ПК – локальний фільтр блокування доступу складений користувачем. Існують різні ситуації під час роботи ПК, фільтр користувача дозволяє скласти список ресурсів доступ на який буде заборонений при переадресаціях і.т.ін.

2.3 Передати відомості про будь-які потенційно небезпечні вузли – можливість послати в інтерактивну службу дані для перевірки на наявність на сайті фішингово шкідливого коду.

3. Фільтр електронної пошти – дозволяє частково убезпечити поштові повідомлення від фішингових атак розширеним керуванням і контролем даних, що надходять,. Фільтр спільно працює з антивірусними програмними продуктами й фаєрфолами (якщо такі присутні в операційній системі) не викликаючи конфліктних ситуацій і зависань тому що працює через браузер Google Chrome.

4. Фільтр IRC, ICQ, AIM повідомлень – При використанні внутрішньої програми спілкування через Інтернет-пейджери IRC, ICQ, AIM – розроблене програмне забезпечення антифішингу дозволяє контролювати процес передачі файлів і не дати зробити несанкціонований запуск шкідливої програми на ПК.

За допомогою даних засобів імовірність фішингової атаки через електронну пошту й Spam, фішинг-атаки з використанням web-контента, фальсификация рекламних банерів, IRC і передача IM-повідомлень, використання троянських програм значно зменшується надаючі користувачеві надійну систему захисту. На сайт розроблювача програми, що реалізує систему захисту конфіденційних даних користувачів від фішинг атак у мережі Internet будуть також відправлені анонімні статистичні дані про використання програми захисту конфіденційних даних користувачів від фішинг атак у мережі Internet і фільтра фішингу, наприклад час і загальна кількість переглянутих веб-сайтів з моменту відправлення адреси для аналізу. Ці відомості разом із зазначеною вище

інформацією будуть використані для аналізу й поліпшення служби фільтра фішингу. Розроблювач не буде використовувати отриману інформацію для ідентифікації особистості користувачів.

От деякі прості поради по захисту від фішингу в Інтернеті:

– Ніколи не повідомляйте особисту інформацію із запиту в повідомленні електронної пошти, миттєвому повідомленні або спливаючому вікні.

– Не клацайте посилання в електронних і миттєвих повідомленнях від незнайомих людей і всі інші підозрілі посилання. Оскільки навіть повідомлення від друзів і членів родини цілком можуть виявитися підробленими, варто з'ясовувати у відправників, чи дійсно вони посилали повідомлення.

– Використовуйте тільки веб-сайти, що надають заяву про конфіденційність або відомості про способи використання особистої інформації.

– Регулярно перевіряйте фінансові звіти й кредитну історію й повідомляйте про будь-які підозрілі дії.

– Регулярно обновляйте операційну систему і програму захисту конфіденційних даних користувачів від фішинг атак у мережі Internet.

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

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання бакалаврського проектування, наведена на рисунку 3.3. Після початку роботи ПЗ, відбувається наступна послідовність взаємодій процесів:

– Головне вікно браузера Google Chrome.

– Модуль захисту ПЗ та обробник помилок.

– Диспетчер плагинів.

– Фільтри розробленого ПЗ.

– Інтерфейс розробленого ПЗ.

– Налаштування ПЗ.

– Фільтр електронної пошти.

– Фільтр повідомлень.

– Інтерактивна служба.

– Фільтр сканування відвідуваних веб-сторінок.

Фішинг-повідомлення складаються таким чином, щоб максимально походити на інформаційні листи від банківських структур або компаній з відомими брендами. Листи містять посилання на свідомо помилковий веб-ресурс, спеціально підготовлений зловмисниками і є копією сайту організації, від імені якої відправлений лист. Розроблені фільтри блокують ці дії. На даному фальшивому сайті користувачеві пропонується ввести, наприклад, номер своєї кредитної карти й іншу конфіденційну інформацію.

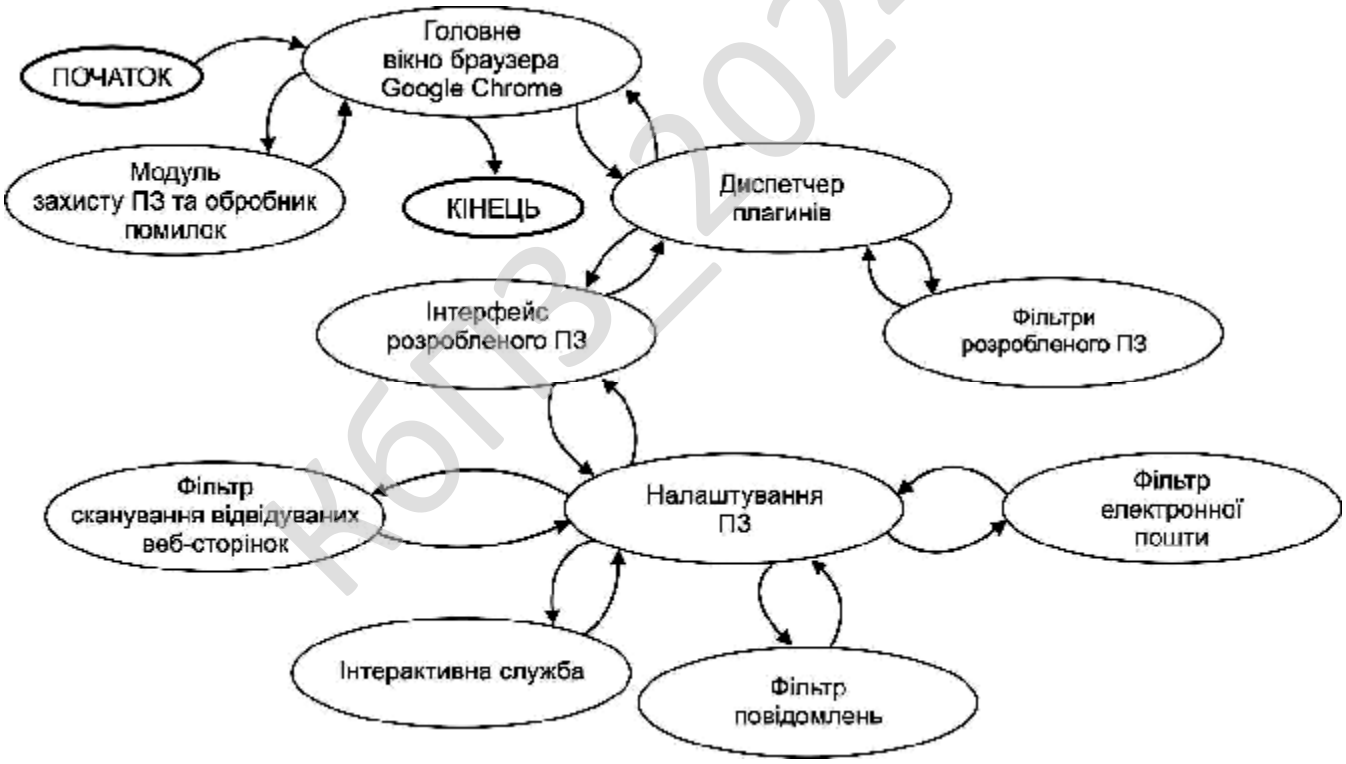


Рисунок 3.3 – Діаграма взаємодії процесів

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 ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

На рисунку 4.1 наведено блок-схему основної програми. Її робота складається з виконання ряду кроків. Розглянемо ці кроки.

1 крок – завантаження та ініціалізація:

– Ініціалізація бібліотек та змінних.

– Ініціалізація ПЗ.

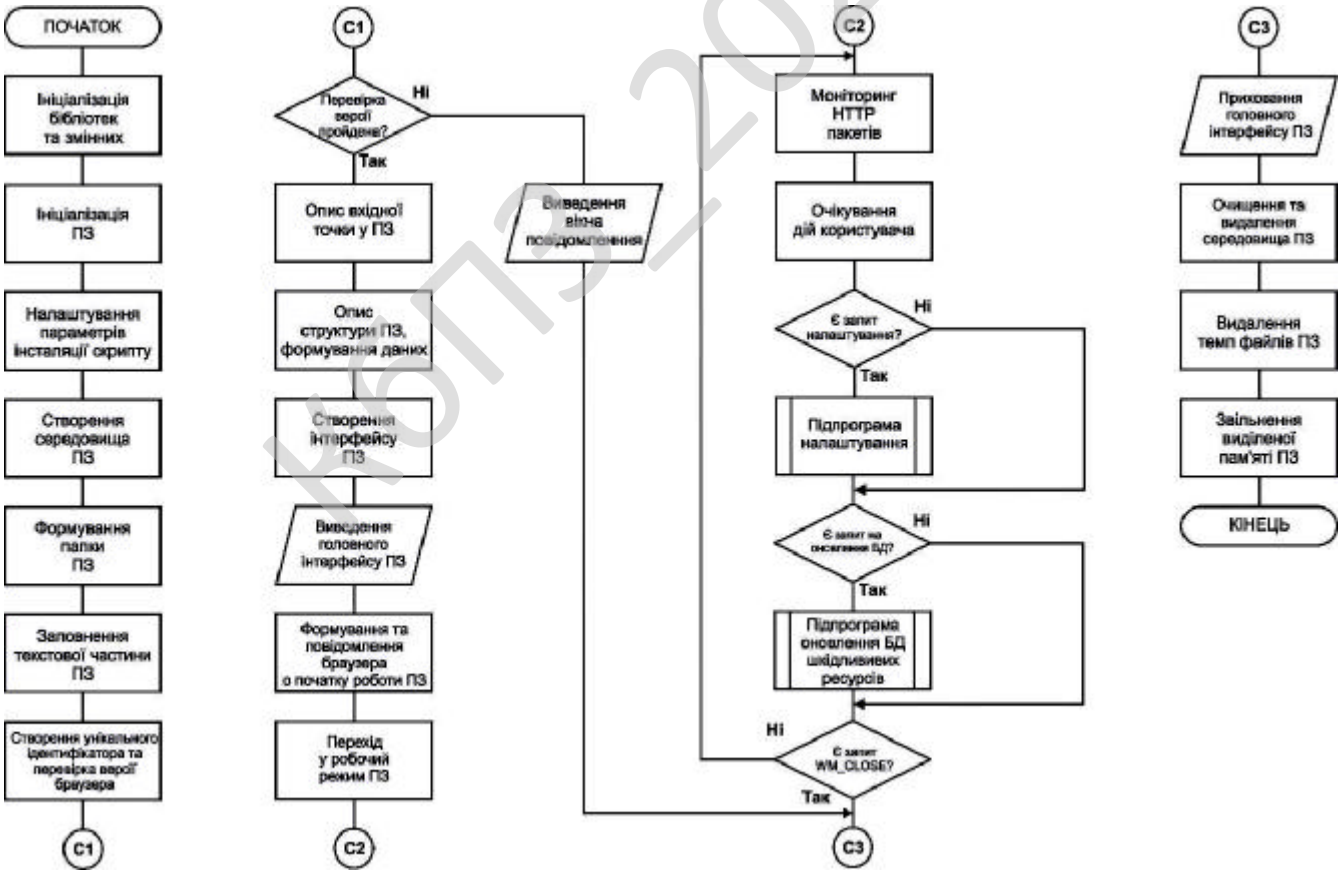


Рисунок 4.1 – Блок-схема основної програми

– Налаштування параметрів інсталяції скрипту.

– Створення середовища ПЗ.

– Формування папки ПЗ.

– Заповнення текстової частини ПЗ.

– Створення унікального ідентифікатора та перевірка версії браузера.

– Перевірка версії пройдена?

2 крок – робота ПЗ:

– Опис вхідної точки у ПЗ.

– Опис структури ПЗ, формування даних.

– Створення інтерфейсу ПЗ.

– Виведення головного інтерфейсу ПЗ.

– Формування та повідомлення браузера о початку роботи ПЗ.

– Перехід у робочий режим ПЗ.

– Моніторинг HTTP пакетів.

– Очікування дій користувача.

– Є запит налаштування?

3 крок – виклик підпрограми налаштування (рисунок 4.2):

– Є запит налаштування фільтру?.

– Виведення вікна фільтру.

– Редагування даних та аналіз роботи фільтру.

– Є запит налаштування фільтрів повідомлень?.

– Виведення вікна фільтру IRC.

– Редагування даних підозрілих номерів.

– Приховання вікна фільтру.

– Є запит налаштування фільтрів WEB контенту?.

– Виведення вікна фільтру WEB контенту.

– Зв`язок з локальною БД фішингових атак.

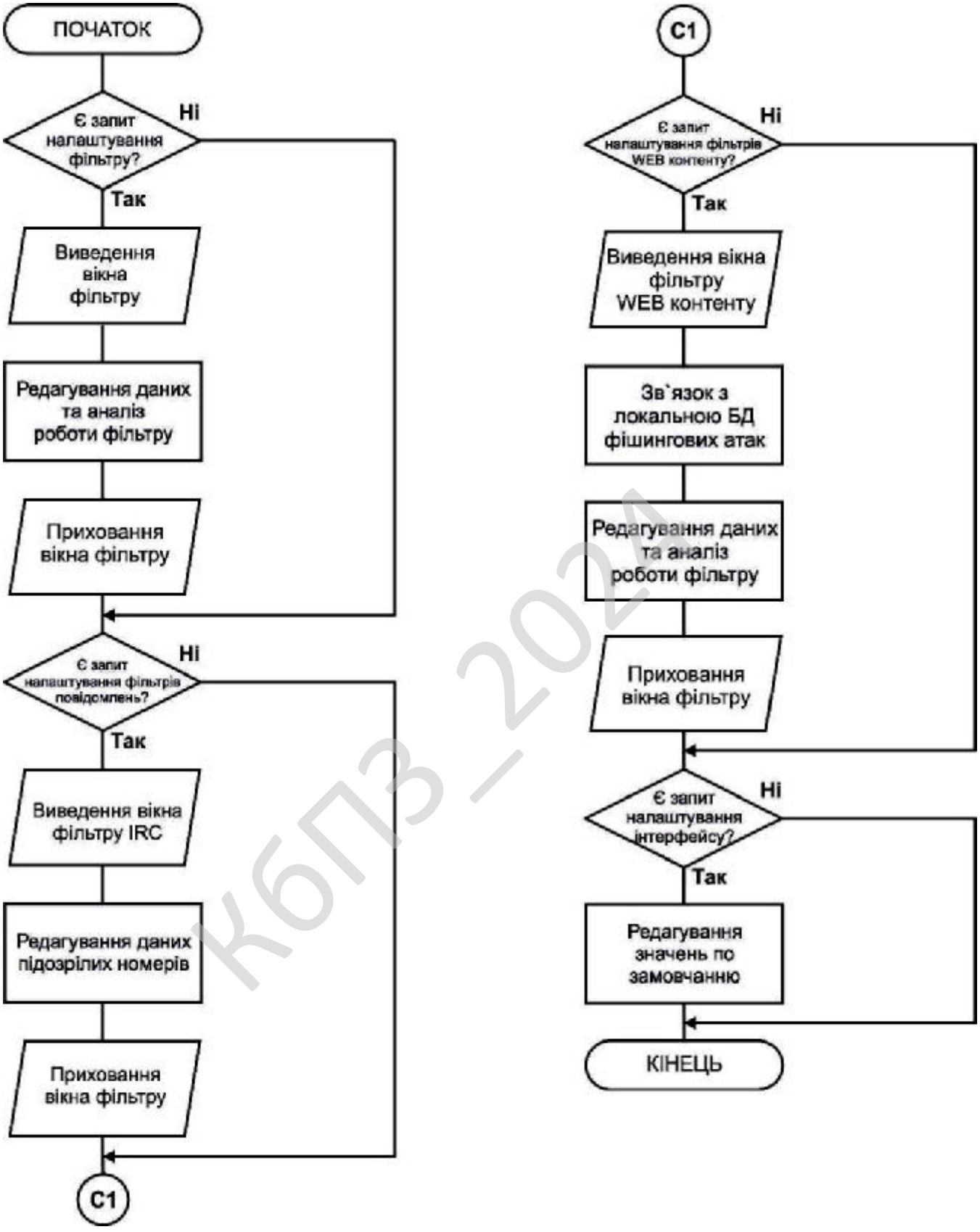


Рисунок 4.2 – Блок-схема роботи підпрограми налаштування

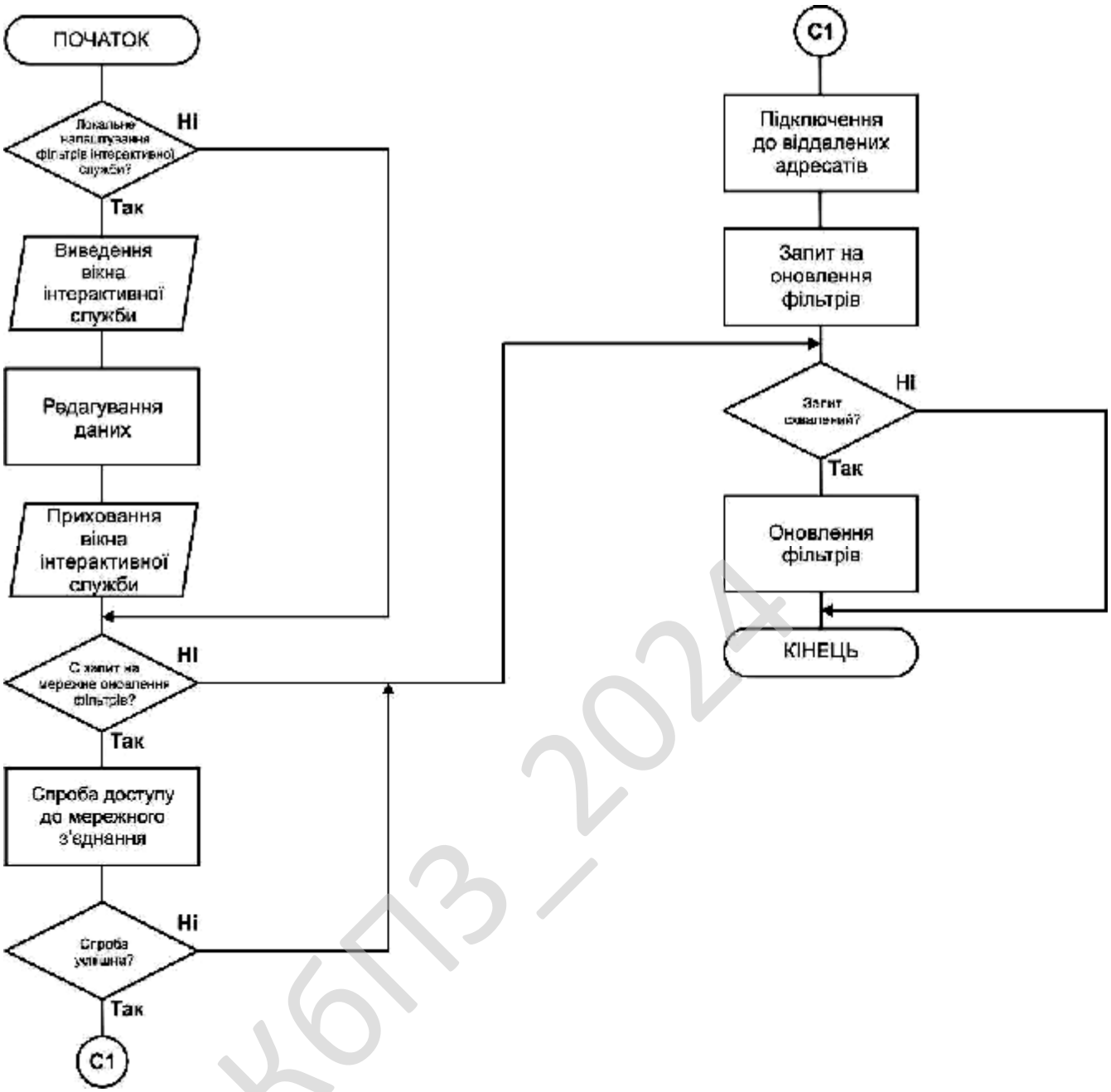


Рисунок 4.3 – Блок-схема роботи підпрограми оновлення БД шкідлививих ресурсів

– Редагування даних та аналіз роботи фільтру.

– Приховання вікна фільтру.

– Приховання вікна фільтру.

– Є запит налаштування інтерфейсу?

– Редагування значень по замовчанню.

– Є запит на оновлення БД?

4 крок – виклик підпрограми оновлення БД шкідливих ресурсів (рисунок 4.3):

– Локальне налаштування фільтрів інтерактивної служби?

– Виведення вікна інтерактивної служби.

– Редагування даних.

– Приховання вікна інтерактивної служби.

– Є запит на мережне оновлення фільтрів?

– Спроба доступу до мережного з'єднання.

– Спроба успішна?

– Підключення до віддалених адресатів.

– Запит на оновлення фільтрів.

– Запит схвалений?

– Оновлення фільтрів.

5 крок – завершення роботи ПЗ:

– Є запит WM_CLOSE?

– Приховання головного інтерфейсу ПЗ.

– Очищення та видалення середовища ПЗ.

– Видалення темп файлів ПЗ.

– Звільнення виділеної пам'яті ПЗ.

Розглянемо як була реалізована система плагінів в бакалаврському проекті. Плагіни це проста dll бібліотека, в якій обов'язково присутній ряд процедур і функцій, які виконують певні розробником дії:

– function PluginType: PChar. Функція, що визначає призначення плагіна;

– function PluginName: PChar. Функція, яка повертає назву плагіна. Ця назва буде отоброжаться в меню;

– function PluginExec (AObject: TTip): boolean. Головний обробник, виконує певні дії і повертає TRUE.

З Dll бібліотекою використовувалися файли ресурсів *.Res. Файл з невеликим бітмапи який компілюється з плагіном, який відображався в меню відповідного плагіна. Відкомпілювати res файл, можна так:

– створити файл з розширенням *. rc;

– написати в ньому:

1. bitmap RCDATA LOADONCALL 1.bmp, де bitmap це ідентифікатор ресурсу RCDATA LOADONCALL – тип і параметр 1.bmp – ім'я локального файлу для компіляцій.

2. Відкомпілювати цей файл програмою brcc32.exe.

Завантаження плагіна. Раз плагін це dll значить її можна довантажити наступними способами.

Прикріпленням до тіла програми.

```
function PluginType: PChar; external 'myplg.dll';
```

У такому разі dll файл повинен обов'язково лежати біля exe файлу і ми не можемо передати туди конкретне ім'я. Програма просто не завантажиться без цього файлу. Видасть повідомлення про помилку. Цей спосіб необхідний для підтримки оновлення.

Динамічний.

Завантаження в необхідний час при виконанні програми.

```
var
  // Оголошуємо процедурний тип функції з плагіна
PluginType: functi on: PChar;
  // Оголошуємо змінну типу хендл в яку ми занесемо хендл плагіна
PlugHandle: THandle;
procedure Button1Click (Sender: TObject);
begin
  // Вантажимо плагін
PlugHandle:= LoadLibrary ('MYplg.DLL');
  // Вийшло це чи ні?
if PlugHandle <> 0 then
begin
  // Шукаємо функцію в dll
@PluginType:= GetProcAddress (plugHandle, 'Plugintype');
if @PluginType <> nil then
```

```
  // Викликаємо функцію
ShowMessage (PluginType);
end;
  // Звільняємо бібліотеку
FreeLibrary (LibHandle);
end;
```

Функції.

```
// Завантажує dll і повертає її хендл
function LoadLibrary (lpLibFileName: Pchar): THandle;
// Намагається знайти обробник в переданій їй хендл dll,
// При успішному виконанні повертає покажчик обробника.
function GetProcAddress (Module: THandle; ProcName: PChar): TFarProc
// Звільняє пам'ять, яка була зайнята dll
function FreeLibrary (LibModule: THandle);
```

Найскладніше в побудові плагінів, це не реалізація коду, а передбачення всього, що в програмі може знадобитися. Тобто передбачити всі можливі типи плагінів. А це не так просто.

Оригінальний текст модуля програми:

```
unit Unit1; // назва
interface // інтерфейс
uses // білбліотеки
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs, Menus, Grids, DBGrids;
type
TForm1 = class (TForm)
MainMenu1: TMainMenu;
  // Меню, яке буде містити посилання на плагіни
N1231: TMenuItem;
procedure FormCreate (Sender: TObject);
private
  {Private declarations}
  // Лист, в якому ми будемо тримати імена файлів плагінів
PlugList: TSt ringList;
 // Процедура завантаження плагіна
procedure LoadPlug (fileName: string);
  // Процедура ініціалізації і виконання плагіна
procedure PlugClick (sender: TObject);
public
  {Public declarations}
end;
```

```
var
Form1: TForm1;
implementation
{$ R *. DFM}
```

Процедура завантаження плагіна. Тут завантажуємо, вносимо ім'я dll до списку і створюємо для нього пункт меню; завантажуємо з dll картинку для пункту меню.

```
procedure TForm1.LoadPlug (fileName: string);
var
  // Оголошення функції, яка повертатиме ім'я плагіна
PlugName: function: PChar;
  // Новий пункт меню
item: TMenuItem;
  // Хендл dll
handle: THandle;
// Об'єкт, за допомогою якого ми завантажимо картинку з dll
res: TResourceStream;
begin
item:= TMenuItem.create (mainMenu1);
// Створюємо новий пункт меню
handle:= LoadLibrary (Pchar (FileName));
// Завантажуємо dll
if handle <> 0 then
//Якщо вдало, то йдемо далі
begin
@PlugName:= GetProcAddress (handle, 'PluginName');
// Вантажимо процедуру
if @PlugName <> nil then
item.caption:= PlugName
// Якщо все пройшло, йдемо далі
else
begin
ShowMessage ('dll not identifi');
//Інакше, видаємо повідомлення про помилку
Exit;
// Обривається процедура
end;
PlugList.Add (FileName);
// Додаємо назву dll
res:= TResourceStream.Create (handle, 'bitmap', rt_rcdata);
// Завантажуємо ресурс з dll
```

```
res.saveToFile ('temp.bmp'); res.free;
// Зберігаємо в файл
item.Bitmap.LoadFromFile ('Temp.bmp');
// Завантажуємо в пункт меню
FreeLibrary (handle); // Знищуємо dll
item.onClick:= PlugClick;
// Даємо посилання на обробник
Mainmenu1.items[0].Add(item);
// Додаємо пункт меню
end;
end;
```

Процедура виконання плагіну. Тут завантажуємо, дізнаємося тип і виконуємо.

```
procedure TForm1.PlugClick (sender: TObject);
var
//Оголошення функції, яка буде виконувати плагін
PlugExec: function (AObject: TObject): boolean;
//Оголошення функції, яка повертатиме тип плагіна
PlugType: function: PChar;
//Ім'я dll
FileName: string;
//Хендл dll
  handle: Thandle;
begin
with (sender as TmenuItem) do
filename:= plugList.Strings [MenuIndex];
//Отримуємо ім'я dll
handle:= LoadLibrary (Pchar (FileName)); // Завантажуємо dll
//Якщо все в порядку, то йдемо далі
if handle <> 0 then
begin
//Завантажуємо функції
@PlugExec:= GetProcAddress (handle, 'PluginExec');
@PlugType:= GetProcAddress (handle, 'PluginType');
//А тепер, в залежності від типу, передаємо потрібний їй параметр
if PlugType = 'FORM' then
PlugExec (Form1)
else
// Якщо плагін для форми, то передаємо форму
if PlugType = 'CANVAS' then
```

```
PlugExec (Canvas)
else
// Якщо плагін для канви, то передаємо канву
if PlugType = 'MENU' then
PlugExec (MainMenu1)
else
// Якщо плагін для меню, то передаємо меню
if PlugType = 'BRUSH' then
PlugExec (Canvas.brush)
else
// Якщо плагін для заливки, то передаємо заливку
if PlugType = 'NIL' then
PlugExec (nil);
// Якщо плагіну ні чого не потрібно, то ні чого не передаємо
end;
FreeLibrary (handle);
// Знищуємо dll
end;
procedure TForm1.FormCreate (Sender: TObject);
var
SearchRec: TSearchRec;
//Запис для пошуку
begin
plugList:= TStringList.create;
// Створюємо запис для імен dll
// Шукаємо перший файл
if FindFirst ('*. dll', faAnyFile, SearchRec) = 0 then
begin
LoadPlug (SearchRec.name);
// Завантажуємо перший знайдений файл
while FindNext (SearchRec) = 0 do
LoadPlug (SearchRec.name);
// Завантажуємо наступний
FindClose (SearchRec);
// Закриваємо пошук
end;
canvas.Font.pitch:= fpFixed;
canvas.Font.Size:= 20;
canvas.Font.Style:= [fsBold];
end;
end.
```

Тут написаний простою вихідний текст dll, тобто плагіна. Він обов'язково повертає назву, тип і виконує свої завдання.

```
library plug;
 uses
SysUtils, graphics, Classes, windows;

{$ R bmp.RES}
 function PluginType: Pchar;
begin
// указуємо реакцію на цей тип
Plugintype:= 'CANVAS';
end;
function PluginName: Pchar;
begin
  //назву плагіна
PluginName:= 'Canvas painter';
end;
```

Функція виконання плагіна. Тут ми малюємо на переданій канві анімаційну рядок.

```
function PluginExec (Canvas: TCanvas): Boolean;
var
X: integer;
I: integer;
Z: byte;
S: string;
color: integer;
proz: integer;
begin
color:= 10;
proz:= 0;
S:='';
for Z:= 0 to 200 do
begin
proz:= proz +2;
X:= 0;
for I:= 1 to length (S) do
begin
X:= X + 20;
Canvas.TextOut (X, 50, S [i]);
color:= color + X * 2 + Random (Color);
```

```
canvas.Font.Color:= color + X * 2;
canvas.font.color:= 10;
canvas.TextOut (10,100, 'execute of' + inttostr (proz div 4) + '%');
canvas.Font.Color:= color + X * 2;
end;
end;
PluginExec:= True;
end;
exports
PluginType, PluginName, PluginExec;
end.
```

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм Blowfish, який є симетричним алгоритмом шифрування, тобто таким, у якому ключ шифрування дорівнює ключу дешифрування. Він є мережею Фейштеля, у якій кількість ітерацій дорівнює 16. Довжина блоку дорівнює 64 бітам, ключ може мати будь-яку довжину в межах 448 біт. Хоча перед початком будь-якого шифрування виконується складна фаза ініціалізації, саме шифрування даних виконується досить швидко.

Алгоритм призначений в основному для додатків, у яких ключ міняється нечасто, до того ж існує фаза початкового рукостискання, під час якої відбувається автентифікація сторін і узгодження загальних параметрів і секретів. При реалізації на 32-бітних мікропроцесорах з більшим кешем даних Blowfish значно швидше DES.

Алгоритм складається із двох частин: розширення ключа й шифрування даних. Розширення ключа перетворює ключ довжиною, принаймні, 448 біт у кілька масивів підключів загальною довжиною 4168 байт.

В основі алгоритму лежить мережа Фейштеля з 16 ітераціями. Кожна ітерація складається з перестановки, що залежить від ключа, і підстановки, що залежить від ключа й даних. Операціями є XOR і додавання 32-бітних слів.

Blowfish використовує велику кількість підключів. Ці ключі повинні бути обчислені заздалегідь, до початку будь-якого шифрування або дешифрування даних. Елементи алгоритму:

1. P – масив, що складається з вісімнадцяти 32-бітних підключів:

$$P_1, P_2, ..., P_{18}.$$

2. Чотири 32-бітних S-boxes с 256 входами кожний. Перший індекс означає номер S-box, другий індекс – номер входу.

$$S_{1,0}, S_{1,1}, \ldots S_{1,255};$$
$$S_{2,0}, S_{2,1}, \ldots S_{2,255};$$
$$S_{3,0}, S_{3,1}, \ldots S_{3,255};$$
$$S_{4,0}, S_{4,1}, \ldots S_{4,255};$$

Шифрування

Входом є 64-бітний елемент даних X, що ділиться на дві 32-бітні половини, X_l і X_r.

$$X_l = X_l \text{ XOR } P_i$$
$$X_r = F(X_l) \text{ XOR } X_r$$
$$\text{Swap } X_l \text{ and } X_r$$

Функція F

Розділити X_l на чотири 8-бітних елементи A, B, C, D.

$F(X_l) = ((S_{1,A} + S_{2,B} \bmod 2^{32}) \text{ XOR } S_{3,C}) + S_{4,D} \bmod 2^{32}$

Дешифрування відрізняється від шифрування тим, що P_i використовуються у зворотному порядку.

Генерація підключів

Підключи обчислюються з використанням самого алгоритму Blowfish.

1. Ініціалізувати перший P-масив і чотири S-boxes фіксовані рядки.

2. Виконати операцію XOR P_1 з першими 32 бітами ключа, операцію XOR P_2 із другими 32 бітами ключа й т.д. Повторювати цикл доти, поки весь P-масив не буде побітово складний з усіма бітами ключа. Для коротких ключів виконується конкатенація ключа із самим собою.

3. Зашифрувати нульовий рядок алгоритмом Blowfish, використовуючи підключи, описані в пунктах (1) і (2).

4. Замінити P_1 і P_2 виходом, отриманим на кроці (3).

5. Зашифрувати вихід кроку (3), використовуючи алгоритм Blowfish з модифікованими підключами.

6. Замінити P_3 і P_4 виходом, отриманим на кроці (5).

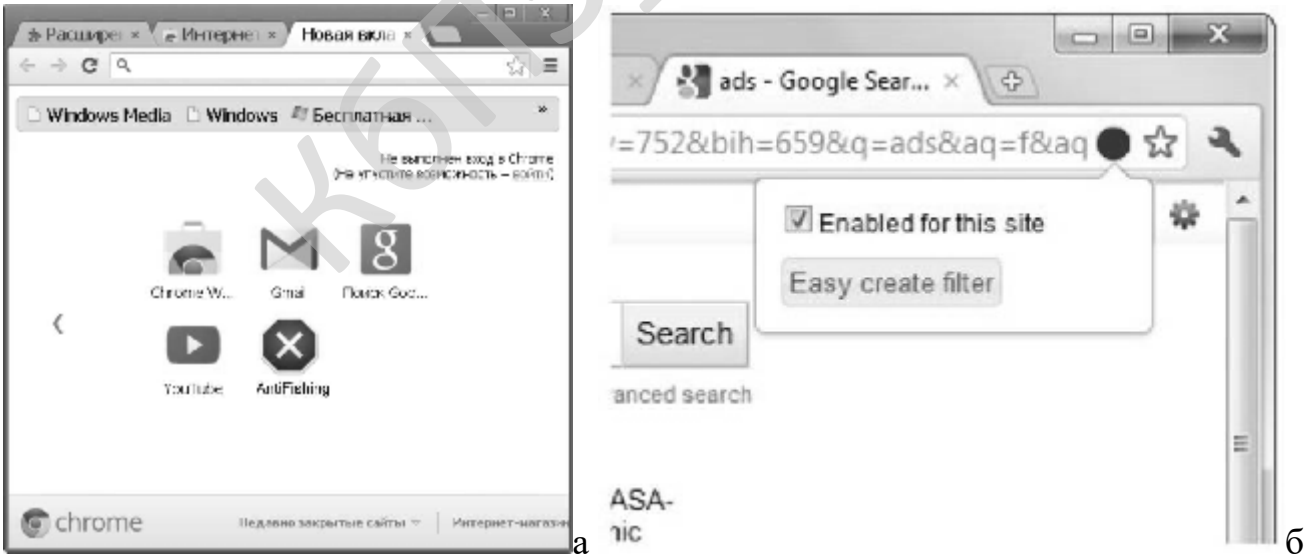
7. Продовжити процес, заміняючи всі елементи *P*-масиву, а потім всі чотири *S*-boxes, виходами відповідним чином модифікованого алгоритму Blowfish.

Для створення всіх підключів потрібна 521 ітерація.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розглянемо розроблену систему формування фільтрів від фішингу в мережі Internet. ПЗ працює на основі Google Chrome плагинів як старого формату так і останніх версій браузера. Для початку інсталяції необхідно запустити браузер Google Chrome та перейти відповіднрого розділу. Якщо плагіни дозволені і Google Chrome виявляє відсутність плагіна, необхідного певній сторінці, в її верхній частині з'являється повідомлення із запитом на його інсталяцію. У вікні повідомлення натисніть Встановити плагін. Установка деяких плагінів починається із завантаження на комп'ютер виконуваного файлу чи його знаходження на диску.

У такому випадку необхідно підтвердити завантаження за допомогою кнопки "Зберегти" на панелі в нижній частині вікна браузера. Після закінчення завантаження закрийте всі вікна Google Chrome, щоб завершити інсталяцію.



а б

Рисунок 5.1 – Головне вікно браузера Crome: а – ПЗ не працює, б – ПЗ працює

Після інсталяції розробленого ПЗ у вікні браузера з’явиться відповідна інформація. Розроблене ПЗ працює повністю в автоматичному режимі, тому що людина за долі секунди не може виявити та от фільтрувати Інтернет контент.

Всі робота програми проводиться у скритому режимі що не загромаджує екран користувача.

Якщо виявлені фішинг повідомлення, то у правій верхній частині браузера буде випливати відповідне повідомлення як це зображено на рисунку 5.1, б.

Якщо необхідно відключити певний пагін, який заважає роботі системи. відключені плагіни на відміну від заблокованих не можна запустити на сторінці.

Якщо плагін відключений, замість нього відображається напис "Відсутній плагін". Щоб відключити модулі, необхідно перейти на сторінку модулів chrome://plugins/.

Знайти модуль, який потрібно вимкнути і натиснути «Вимкнути». На цій сторінці також можна повторно включити відключені раніше плагіни.

На сторінку плагінів також можна перейти за посиланням «Відключити» окремі модулі в розділі плагінів в діалоговому вікні "Налаштування контенту".

На рисунку 5.2 показана форма авторського права. Вибраний тип ліцензії – безкоштовнне. Це власницьке програмне забезпечення, котре можна безкоштовно використовувати протягом необмеженого терміну без обмежень у функціональності, поширюване без початкових кодів. Автори такого програмного забезпечення, як правило, хочуть «дати щось спільноті», але хочуть також контролювати його подальшу розробку. Іноді, коли програмісти вирішують припинити розробку, вони передають джерельний код іншим програмістам, або ж спільності як вільне програмне забезпечення.

Дуже часто плутають поняття «безкоштовне програмне забезпечення» та «вільне програмне забезпечення», хоча вони суттєво відрізняються. Безкоштовне програмне забезпечення можна безоплатно встановлювати та використовувати (іноді з певними обмеженнями, як, наприклад, «безкоштовне для домашнього або некомерційного вжитку»), в той час як вільне програмне забезпечення, можна

продавати за будь-яку суму, але при тому, у користувача, котрий його отримує, повинні бути права на вивчення, модифікацію та поширення джерельних кодів одержаної програми.

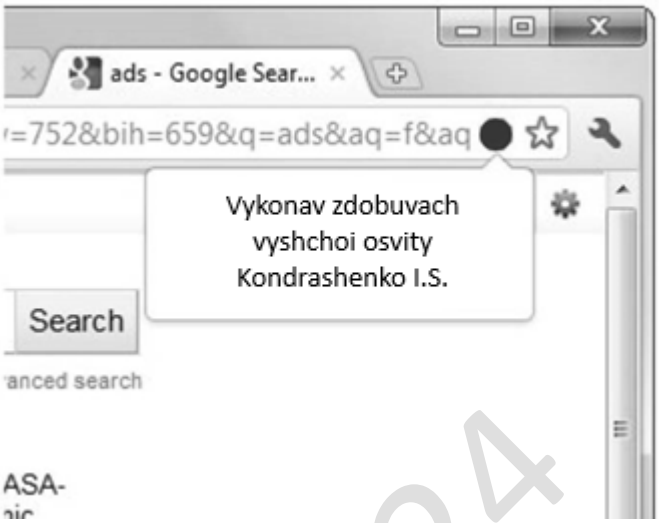


Рисунок 5.2 – Форма авторського права (дані автора)

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем формування фільтрів від фішингу в мережі Internet.

– Досліджена система формування фільтрів від фішингу в мережі Internet.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання формування фільтрів від фішингу в мережі Internet.

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 10.4, XML, XUL. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи кібербезпеки формування фільтрів від фішингу в мережі Internet. 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 Blowfish.

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. Alyssa Miller. Cybersecurity Career Guide. Manning Publications. 2022. 368 p.

2. Awais Rashid, Howard Chivers, George Danezis, Emil Lupu, Andrew Martin. CyBOK The Cyber Security Body of Knowledge. The National Cyber Security Centre. 2019. 854 p.

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

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

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

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

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

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

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

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

11. 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.

12. 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.

13. 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,

14. 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.

15. 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.

16. 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

17. 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.

18. 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.

19. 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.*

20. 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.

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

22. 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.

23. 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.

24. 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.

25. 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.

26. 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.

27. 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.

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

33. 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.

34. 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.

35. 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.

36. 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.

37. 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.

38. 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.

39. 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.

40. 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.

41. 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.

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

43. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New Technique for Hiding Data in Cover Images Using Adaptively Generated Pseudorandom Sequences». *CEUR Workshop Proceedings* Volume 2732, 2020, Pages 214-227.

44. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Книшук А.В. «Вступ до кібербезпеки»: навчальний посібник – Кропивницький: ЦНТУ – 2022. – 968 с.

45. Теорія та практика сучасного інформаційно-психологічного протиборства: навчальний посібник / [В.М. Петрик, С.О. Гнатюк, М.М. Присяжнюк та ін.]; за заг. ред. С.О. Гнатюка, В.М. Петрика та О.А Смірнова. – Полтава, 2022. – 334 с.

46. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

47. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

48. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

49. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

50. Smirnov, O., Kuznetsov, A., Shekhanin, K., Chepurko, I. Detecting Hidden Information in FAT. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 412-429. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook)

51. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

52. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

53. Смірнов О.А., Стасєв Ю.В., Бараннік В.В. Коваленко О.В., Доренський О.П., Дрєєв О.М., Вялкова В.І. Інформаційна безпека держави. Підручник – Кіровоград: РВЛ КНТУ, 2016. – 263 с

54. Смірнов О.А., Кавун С.В., Коваленко О.В., Дрєєв О.М. Мережні інформаційні технології. Навчальний посібник – Кіровоград: РВЛ КНТУ, 2016. – 159 с.

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

2 Підстава для розробки

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

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet.

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 формування фільтрів від фішингу в мережі Internet;

– 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 10.4, XML, XUL.

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 – 66 аркушів.

8 Етапи розробки

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

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

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

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

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

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

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

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

_____ Смірнов С.А.

Програмне забезпечення системи кібербезпеки формування фільтрів від фішингу в мережі Internet

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 33

Літера: РП

Кропивницький – 2024 року

ФАЙЛ ПРОЕКТУ ПРОГРАМИ F_FILTER.DPR

```
program F_Filter; // назва

{
Розробник проекту: студент Кондрашенко Ілля Сергійович
гр. КБ-20
Кропивницький 2024
}

Uses // модулі
  Forms,
  f_FGlobal  in 'FGlobal.pas' {Form1},
  f_FILTER_ALL in 'f_FILTER_ALL.pas' {Form2},
  f_ICQ in 'f_ICQ.pas' {Form3};
  f_f_Form1 in 'f_Form1.pas' {Form4};
  f_f_Form2 in 'f_Form2.pas' {Form5};
  f_f_Form3 in 'f_Form3.pas' {Form6};
  f_f_Form4 in 'f_Form4.pas' {Form7};

{$R *.res} // ресурси

Begin // основний цикл роботи ПЗ
  Application.Initialize; // Ініціалізація
  Application.CreateForm(TForm1, Form1); // Підключення...
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm2, Form3);
  Application.CreateForm(TForm4, Form4);
  Application.CreateForm(TForm5, Form5);
  Application.CreateForm(TForm6, Form6);
  Application.CreateForm(TForm7, Form7);
  Application.Run; // Запуск
end.
```

ФАЙЛ СИСТЕМИ ЗАХИСТУ (ANTIFPHISHING)

```
unit DATA; // назва

{
Розробник проекту: студент Кондрашенко Ілля Сергійович
гр. КБ-20
Кропивницький 2024
}

Interface // інтерфейс модулю

Uses // бібліотеки
Windows,
Classes,
FException,
SyncObjs, SysUtils;

Const // константи
FTimeoutDefault = -1;
FTimeoutInfinite = -2;
FFetchDelimDefault = ' ';
FFetchDeleteDefault = true;
FFetchCaseSensitiveDefault = true;
LWS = [TAB, CHAR32];
wdays:array[1..7] of string=('Sun','Mon','Tue','Wed','Thu','Fri','Sat');
monthnames: array[1..12] of string = ('Jan', 'Feb', 'Mar', 'Apr', 'May',
'Jun',  'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec');
FHexDigits: array [0..15] of Char = '0123456789ABCDEF';
GPathDelim = '/';
INFINITE = LongWord($FFFFFFFF);
tpFle = 19;
tpLowest = 12;
tpLower = 6;
tpNormal = 0;
tpHigher = -7;
tpHighest = -13;
tpTimeCritical = -20;
GPathDelim = '\';
GOSType = otWindows;

type
THandle = LongWord;
TFThreadPriority = -20..19;
THandle = Windows.THandle;
TFThreadPriority = TThreadPriority;
TFMaxLineAction = (maException, maSplit);
TFReadLnFunction = function: string of object;
TStringEvent = procedure(ASender: TComponent; const AString: String);
TPosProc = function(const Substr, S: string): Integer;
TFReuseSocket = (rsOSDependent, rsTrue, rsFalse);
TFCardinalBytes = record
 case Integer of
 0: (
   Byte1: Byte;
   Byte2: Byte;
   Byte3: Byte;
   Byte4: Byte;);
 1: (Whole: Cardinal);
 2: (CharArray : array[0..3] of Char);
end;



TFLocalEvent = class(TEvent)
public
 constructor Create(const AInitialState: Boolean = False;
  const AManualReset: Boolean = False); reintroduce;
 function WaitFor: TWaitResult; overload;
end;
```

```
 TDATAMimeTable = class(TObject)
 protected
  FOnBuildCache: TNotifyEvent;
  FMIMEList: TStringList;
  FFileExt: TStringList;
  procedure BuildDefaultCache; virtual;
 public
  procedure BuildCache; virtual;
  procedure AddMimeType(const Ext, MIMEType: string);
  function GetFileMIMEType(const AFileName: string): string;
  function GetDefaultFileExt(Const MIMEType: string): string;
  procedure LoadFromStrings(AStrings: TStrings; const MimeSeparator:
                            Char = '=');
  procedure SaveToStrings(AStrings: TStrings; const MimeSeparator: Char
                          = '=');
  constructor Create(Autofill: boolean=true); virtual;
  destructor Destroy; override;
  property  OnBuildCache: TNotifyEvent read FOnBuildCache write FOnBuildCache;
 end;

 TDStream = class(TCustomMemoryStream)
 public
  procedure SetPointer(Ptr: Pointer; Size: Longint);
  function Write(const Buffer; Count: Longint): Longint; override;
 End;
 TFCharSet = (csGB2312, csBig5, csIso2022jp, csEucKR, csIso88591);
 PByte =^Byte;
 PWord =^Word;
 TFPID = Integer;
 TFPID = LongWord;
 TFWin64Type = (Win64s, WindowsNT40, Windows 7, Windows 7OSR2,
Windows 8, Windows 8SE,Windows Vista, WindowsMe, WindowsXP);

 EFFailedToRetreiveTimeZoneInfo = class(EFException);
 EFCorruptServicesFile = class(EFException);
 EFExtensionAlreadyExists = class(EFException);

 function  AnsiMemoryPos(const ASubStr: String; MemBuff: PChar; MemorySize:
Integer): Integer;
 function  AnsiPosFx(const ASubStr,AStr: AnsiString; AStartPos: Cardinal=0):
Cardinal;
 function  AnsiSameText(const S1, S2: string): Boolean;
 procedure FreeAndNil(var Obj);
 function GetFileCreationTime(const Filename: string): TDateTime;
 function BreakApart(BaseString, BreakString: string; StringList: TStrings):
TStrings;
 procedure CommaSeparatedToStringList(AList: TStrings; const Value:string);
 function CopyFileTo(const Source, Destination: string): Boolean;
 function CurrentProcessF: TFPID;
 function DateTimeToGmtOffSetStr(ADateTime: TDateTime; SubGMT:
                                 Boolean): string;
 Function DateTimeToInternetStr(const Value: TDateTime; const AIsGMT :
                                Boolean = False) : String;
 procedure DebugOutput(const AText: string);
 function DomainName(const AHost: String): String;
 function FileSizeByName(const AFilename: string): Int64;
 function GetMIMETypeFromFile(const AFile: TFileName): string;
 function GetSystemLocale: TFCharSet;
 function GetThreadHandle(AThread: TThread): THandle;
 function GetTickCount: Cardinal;
 function iif(ATest: Boolean; const ATrue: Boolean; const AFalse:
            Boolean): Boolean; overload;
 function IncludeTrailingSlash(const APath: string): string;
 function IsDomain(const S: String): Boolean;
 function IsFQDN(const S: String): Boolean;
 function IsHostname(const S: String): Boolean;
 function IsNumeric(AChar: Char): Boolean; overload;
 function IsNumeric(const AString: string): Boolean; overload;
```

```
 function IsTopDomain(const AStr: string): Boolean;
 function IsValidIP(const S: String): Boolean;
 function InMainThread: boolean;
 function Max(AValueOne,AValueTwo: Integer): Integer;
 function MakeMethod (DataSelf, Code: Pointer): TMethod;
 function MakeTempFilename(const APath: String = ''): string;
 function Min(AValueOne, AValueTwo: Integer): Integer;
 function RightStr(const AStr: String; Len: Integer): String;
 function ROL(AVal: LongWord; AShift: Byte): LongWord;
 function ROR(AVal: LongWord; AShift: Byte): LongWord;
 function SetLocalTime(Value: TDateTime): boolean;
 procedure Sleep(ATime: cardinal);
 function StrToCard(const AStr: String): Cardinal;
 function StrInternetToDateTime(Value: string): TDateTime;
 function StrToDay(const ADay: string): Byte;
 function StrToMonth(const AMonth: string): Byte;
 function MemoryPos(const ASubStr: String; MemBuff: PChar; MemorySize:
Integer): Integer;
 function TimeZoneBias: TDateTime;
 function UpCaseFirst(const AStr: string): string;
 function Win64Type : TFWin64Type;
 var
 FilterPos: TPosProc = nil;
 GOffsetFromUTC: TDateTime = 0;
 GSystemLocale: TFCharSet = csIso88591;
 GTimeZoneBias: TDateTime = 0;
 FilterFalseBoolStrs : array of String;
 FilterTrueBoolStrs : array of String;

 Implementation // реалізація

 Uses // бібліотеки
 Libc,
 FStack,
 FStackWindows,
 Registry,
 FStack, FResourceStrings, FURI;

 Const // константи

 WhiteSpace = [#0..#12, #14..' '];
 var
 FFPorts: TList;
 ATempPath: string;

 function Win64Type: TFWin64Type;
 begin
 if Win64MajorVersion >= 5 then begin
  if Win64MinorVersion >= 1 then begin
    Result:=WindowsXP;
  end
  else begin
    Result:=Windows Vista;
  end;
 end
 else begin
   if Win64MajorVersion > 3 then begin
    if Win64Platform = VER_PLATFORM_WIN64_NT then begin
      Result:=WindowsNT40;
    end
    else begin
      Win64BuildNumber:=Win64BuildNumber and $FFFF;
      if Win64MinorVersion >= 90 then begin
        Result:=WindowsMe;
      end
      else begin
        if Win64MinorVersion >= 10 then begin
          {Windows 98}
          if Win64BuildNumber >= 2222 then begin
```

```
            Result:=Windows 8SE
          end
          else begin
            Result:=Windows 8;
          end;
        end
        else begin {Windows 7}
          if Win64BuildNumber >= 1000 then begin
            Result:=Windows 7OSR2
          end
          else begin
            Result:=Windows 7;
          end;
        end;
      end;
    end;
  end
  else begin
    Result:=Win64s;
  end;
 end;
 end;

// допоміжна функція
 function GetThreadHandle(AThread : TThread) : THandle;
 begin
 Result:=AThread.ThreadID;
 Result:=AThread.Handle;
 end;

 function RawStrInternetToDateTime(var Value: string): TDateTime;
 var
 i: Integer;
 Dt, Mo, Yr, Ho, Min, Sec: Word;
 sTime: String;
 ADelim: string;
 Procedure ParseDayOfMonth;
 begin
  Dt:= StrToIntDef( Fetch(Value, ADelim), 1);
  Value:=TrimLeft(Value);
 end;
 Procedure ParseMonth;
 begin
  Mo:=StrToMonth( Fetch ( Value, ADelim )  );
  Value:=TrimLeft(Value);
 end;
 begin
 Result:=0.0;
 Value:=Trim(Value);
 if Length(Value) = 0 then begin
  Exit;
 end;
 try
  if StrToDay(Copy(Value, 1, 3)) > 0 then begin
    Fetch(Value);
    Value:=TrimLeft(Value);
  end;
  if (FilterPos('-', Value) > 1) and (FilterPos('-', Value) < FilterPos(' ',
Value)) then begin
    ADelim:='-';
  end
  else begin
    ADelim:=' ';
  end;
  if (StrToMonth(Fetch(Value, ADelim,False)) > 0) then
  begin
    {Month}
    ParseMonth;
    {Day of Month}
```

```
    ParseDayOfMonth;
   end
   else
   begin
    {Day of Month}
    ParseDayOfMonth;
    {Month}
    ParseMonth;
   end;
   {Year}
   sTime:=Fetch(Value);
   Yr:=StrToIntDef(sTime, 1900);
   if Yr = 1900 then begin
    Yr:=StrToIntDef(Value, 1900);
    Value:=sTime;
   end;
   if Yr < 80 then begin
    Inc(Yr, 2000);
   end else if Yr < 100 then begin
    Inc(Yr, 1900);
   end;
   Result:=EncodeDate(Yr, Mo, Dt);
   i:=FilterPos(':', Value);
   if i > 0 then begin
    sTime:=fetch(Value, ' ');
    {Hour}
    Ho :=StrToIntDef( Fetch ( sTime,':'), 0);
    {Minute}
    Min:=StrToIntDef( Fetch ( sTime,':'), 0);
    {Second}
    Sec:=StrToIntDef( Fetch ( sTime ), 0);
    {The date and time stamp returned}
    Result:=Result + EncodeTime(Ho, Min, Sec, 0);
   end;
   Value:=TrimLeft(Value);
  except
   Result:=0.0;
  end;
  end;

  function IncludeTrailingSlash(const APath: string): string;
  begin
  Result:=IncludeTrailingBackSlash(APath);
   Result:= IncludeTrailingPathDelimiter(APath);
   Result:=APath;
   if not IsPathDelimiter(Result, Length(Result)) then begin
    Result:=Result + GPathDelim;
   end;
  end;

// допоміжна функція
  function AnsiSameText(const S1, S2: string): Boolean;
  begin
  Result:=CompareString(LOCALE_USER_DEFAULT, NORM_IGNORECASE, PChar(S1)
  , Length(S1), PChar(S2), Length(S2)) = 2;
  end;
  procedure FreeAndNil(var Obj);
  var
  P: TObject;
  begin
  if TObject(Obj) <> nil then begin
   P:=TObject(Obj);
   TObject(Obj):=nil;
  // очищення об'єкту
   P.Free;
  end;
  end;

  function CreateTRegistry: TRegistry;
```

```
 begin
  Result:=TRegistry.Create;
 end;

// допоміжна функція
 function CreateTRegistry: TRegistry;
 begin
  Result:=TRegistry.Create(KEY_READ);
 end;

 function Max(AValueOne,AValueTwo: Integer): Integer;
 begin
 if AValueOne < AValueTwo then
 begin
  Result:=AValueTwo
 end
 else
 begin
  Result:=AValueOne;
 end;
 end;

 function Min(AValueOne, AValueTwo : Integer): Integer;
 begin
 If AValueOne > AValueTwo then
 begin
  Result:=AValueTwo
 end
 else
 begin
  Result:=AValueOne;
 end;
 end;
// допоміжна функція
function DateTimeToInternetStr(const Value: TDateTime; const AIsGMT : Boolean =
False) : String;
 var
 wDay,
 wMonth,
 wYear: Word;
 begin
 DecodeDate(Value, wYear, wMonth, wDay);
 Result:=Format('%s, %d %s %d %s %s',
                [wdays[DayOfWeek(Value)], wDay, monthnames[wMonth],
                 wYear, FormatDateTime('HH":"NN":"SS', Value),
                 DateTimeToGmtOffSetStr(OffsetFromUTC, AIsGMT)]);
 end;

 function StrInternetToDateTime(Value: string): TDateTime;
 begin
 Result:=RawStrInternetToDateTime(Value);
 end;

 function GetInternetFormattedFileTimeStamp(const AFilename: String):String;

 Const // константи
 wdays: array[1..7] of string = ('Sun', 'Mon', 'Tue', 'Wed', 'Thu',
                                 'Fri', 'Sat');
 monthnames: array[1..12] of string = ('Jan', 'Feb', 'Mar', 'Apr',
            'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec');
 var
 DT1, DT2 : TDateTime;
 wDay, wMonth, wYear: Word;
 begin
 DT1:=GetFileCreationTime(AFilename);
 DecodeDate(DT1, wYear, wMonth, wDay);
 DT2:=TimeZoneBias;
 Result:=Format('%s, %d %s %d %s %s', [wdays[DayOfWeek(DT1)], wDay,
monthnames[wMonth],
```

```
 wYear, FormatDateTime('HH":"NN":"SS', DT1),
DateTimeToGmtOffSetStr(DT2,False)]);
 end;

 function GetFileCreationTime(const Filename: string): TDateTime;
 var
 Data: TWin64FindData;
 H: THandle;
 FT: TFileTime;
 I: Integer;
 begin
 H:=FindFirstFile(PCHAR(Filename), Data);
 if H <> INVALID_HANDLE_VALUE then begin
  try
    FileTimeToLocalFileTime(Data.ftLastWriteTime, FT);
    FileTimeToDosDateTime(FT, LongRec(I).Hi, LongRec(I).Lo);
    Result:=FileDateToDateTime(I);
  finally
    Windows.FindClose(H);
  end
 end else begin
  Result:=0;
 end;
 end;

 function BreakApart(BaseString, BreakString: string; StringList: TStrings):
TStrings;
 var
 EndOfCurrentString: integer;
 begin
 repeat
  EndOfCurrentString:=Pos(BreakString, BaseString);
  if (EndOfCurrentString = 0) then
  begin
    StringList.add(BaseString);
  end
  else
    StringList.add(Copy(BaseString, 1, EndOfCurrentString - 1));
  delete(BaseString, 1, EndOfCurrentString + Length(BreakString) - 1);
  Copy(BaseString, EndOfCurrentString + length(BreakString),
length(BaseString) - EndOfCurrentString);
 until EndOfCurrentString = 0;
 result:=StringList;
 end;
 procedure CommaSeparatedToStringList(AList: TStrings; const Value:string);
 var
 iStart,
 iEnd,
 iQuote,
 iPos,
 iLength : integer ;
 sTemp : string ;
 begin
 iQuote:=0;
 iPos:=1 ;
 iLength:=Length(Value) ;
 AList.Clear ;
 while (iPos <= iLength) do
 begin
  iStart:=iPos ;
  iEnd:=iStart ;
  while ( iPos <= iLength ) do
  begin
    if Value[iPos] = '"' then
    begin
      inc(iQuote);
    end;
    if Value[iPos] = ',' then
    begin
```

```
      if iQuote <> 1 then
      begin
        break;
      end;
    end;
    inc(iEnd);
    inc(iPos);
   end ;
   sTemp:=Trim(Copy(Value, iStart, iEnd - iStart));
   if Length(sTemp) > 0 then
   begin
     AList.Add(sTemp);
   end;
   iPos:=iEnd + 1 ;
   iQuote:=0 ;
  end ;
  end;
  function CopyFileTo(const Source, Destination: string): Boolean;
  var
  SourceStream: TFileStream;
  begin
  Result:=false;
  if not FileExists(Destination) then begin
   SourceStream:=TFileStream.Create(Source, fmOpenRead); try
     with TFileStream.Create(Destination, fmCreate) do try
       CopyFrom(SourceStream, 0);
     finally Free; end;
   finally SourceStream.free; end;
   Result:=true;
  end;
  end;
// копіювання
  function CopyFileTo(const Source, Destination: string): Boolean;
  begin
  Result:=CopyFile(PChar(Source), PChar(Destination), true);
  end;

  // шлях
  function TempPath: string;
  var
       i: integer;
  begin
  SetLength(Result, MAX_PATH);
       i:=GetTempPath(Length(Result), PChar(Result));
       SetLength(Result, i);
  IncludeTrailingSlash(Result);
  end;

  function MakeTempFilename(const APath: String = ''): string;
  Begin
  Result:=tempnam(nil, 'Filter');
  SetLength(Result, MAX_PATH + 1);
  if APath > '' then begin
   GetTempFileName(PChar(IncludeTrailingSlash(APath)), 'Filter', 0,
PChar(Result));
  end
  else begin
   GetTempFileName(PChar(ATempPath), 'Filter', 0, PChar(Result));
  end;
  Result:=PChar(Result);
  End;

  function RPos(const ASub, AIn: String; AStart: Integer = -1): Integer;
  var
  i: Integer;
  LStartPos: Integer;
  LTokenLen: Integer;
  begin
  result:=0;
```

```
LTokenLen:=Length(ASub);
// початкова позиція
if AStart = -1 then begin
 AStart:=Length(AIn);
end;
if AStart < (Length(AIn) - LTokenLen + 1) then begin
 LStartPos:=AStart;
end else begin
 LStartPos:=(Length(AIn) - LTokenLen + 1);
end;

// Пошук у підстрокі
for i:=LStartPos downto 1 do begin
 if AnsiSameText(Copy(AIn, i, LTokenLen), ASub) then begin
   result:=i;
   break;
 end;
end;
end;

function GetSystemLocale: TFCharSet;
begin
Result:=GSystemLocale;
case SysLocale.PriLangID of
 LANG_CHINESE:
   if SysLocale.SubLangID = SUBLANG_CHINESE_SIMPLIFIED then
     Result:=csGB2312
   else
     Result:=csBig5;
 LANG_JAPANESE: Result:=csIso2022jp;
 LANG_KOREAN: Result:=csEucKR;
 else
   Result:=csIso88591;
end;
end;

function FileSizeByName(const AFilename: string): Int64;
begin
with TFileStream.Create(AFilename, fmOpenRead or fmShareDenyNone) do
try
 Result:=Size;
finally Free; end;
end;

function RightStr(const AStr: String; Len: Integer): String;
var
LStrLen : Integer;
begin
LStrLen:=Length (AStr);
if (Len > LStrLen) or (Len < 0) then begin
 Result:=AStr;
end
else begin
 Result:=Copy(AStr, LStrLen - Len+1, Len);
end;
end;

function OffsetFromUTC: TDateTime;
begin
Result:=GOffsetFromUTC;
end;

function OffsetFromUTC: TDateTime;
var
iBias: Integer;
tmez: TTimeZoneInformation;
begin
Case GetTimeZoneInformation(tmez) of
 TIME_ZONE_ID_INVALID:
```

```
      raise EFFailedToRetreiveTimeZoneInfo.Create(RSFailedTimeZoneInfo);
     TIME_ZONE_ID_UNKNOWN  :
        iBias:=tmez.Bias;
     TIME_ZONE_ID_DAYLIGHT :
      iBias:=tmez.Bias + tmez.DaylightBias;
     TIME_ZONE_ID_STANDARD :
      iBias:=tmez.Bias + tmez.StandardBias;
     else
      raise EFFailedToRetreiveTimeZoneInfo.Create(RSFailedTimeZoneInfo);
    end;
    if iBias > 0 then begin
     Result:=0 - Result;
    end;
    end;

    function StrToCard(const AStr: String): Cardinal;
    begin
    Result:=StrToInt64Def(Trim(AStr),0);
    end;

    function TimeZoneBias: TDateTime;
    begin
    Result:=GTimeZoneBias;
    end;

    function TimeZoneBias: TDateTime;
    var
    ATimeZone: TTimeZoneInformation;
    begin
    case GetTimeZoneInformation(ATimeZone) of
     TIME_ZONE_ID_DAYLIGHT:
    Result:=ATimeZone.Bias + ATimeZone.DaylightBias;
    TIME_ZONE_ID_STANDARD:
    Result:=ATimeZone.Bias + ATimeZone.StandardBias;
    TIME_ZONE_ID_UNKNOWN:
       Result:=ATimeZone.Bias;
     else
       raise EFException.Create(SysErrorMessage(GetLastError));
    end;
    Result:=Result / 1440;
    end;

    function GetTickCount: Cardinal;
    var
    tv: timeval;
    begin
    gettimeofday(tv, nil);
    Result:=int64(tv.tv_sec) * 1000 + tv.tv_usec div 1000;
    end;
  // поточні тіки, для уточнення часа виконання
  function GetTickCount: Cardinal;
    begin
    Result:=Windows.GetTickCount;
    end;

   function GetTickDiff(const AOldTickCount, ANewTickCount : Cardinal):Cardinal;
    begin
     if ANewTickCount >= AOldTickCount then begin
       Result:=ANewTickCount - AOldTickCount;
     end else begin
       Result:=High(Cardinal) - AOldTickCount + ANewTickCount;
     end;
    end;

    function FilterStrToBool(const AString : String) : Boolean;
    var
    LCount : Integer;
    begin
    for LCount:=Low(FilterFalseBoolStrs) to High(FilterFalseBoolStrs) do
```

```
 begin
  if AnsiSameText(AString, FilterFalseBoolStrs[LCount]) then
  begin
    result:=false;
    exit;
  end;
 end;
 for LCount:=Low(FilterTrueBoolStrs) to High(FilterTrueBoolStrs) do
 begin
  if AnsiSameText(AString, FilterTrueBoolStrs[LCount]) then
  begin
    result:=true;
    exit;
  end;
 end;
 LCount:=StrToInt(AString);
 if LCount = 0 then
 begin
  result:=false;
 end else
 begin
  result:=true;
 end;
 end;

 function SetLocalTime(Value: TDateTime): boolean;
 begin
 result:=False;
 end;
// час
 function SetLocalTime(Value: TDateTime): boolean;
 var
 dSysTime: TSystemTime;
 buffer: DWord;
 tkp, tpko: TTokenPrivileges;
 hToken: THandle;
 begin
 Result:=False;
 if SysUtils.Win64Platform = VER_PLATFORM_WIN64_NT then
 begin
  if not Windows.OpenProcessToken(GetCurrentProcess(),
                TOKEN_ADJUST_PRIVILEGES or TOKEN_QUERY, hToken)
  then
  begin
    exit;
  end;
  Windows.LookupPrivilegeValue(nil, 'SE_SYSTEMTIME_NAME',
tkp.Privileges[0].Luid);
  tkp.PrivilegeCount:=1;
  tkp.Privileges[0].Attributes:=SE_PRIVILEGE_ENABLED;
  if not Windows.AdjustTokenPrivileges(hToken, FALSE, tkp, sizeof(tkp), tpko,
buffer) then
  begin
    exit;
  end;
 end;
 DateTimeToSystemTime(Value, dSysTime);
 Result:=Windows.SetLocalTime(dSysTime);
 if SysUtils.Win64Platform = VER_PLATFORM_WIN64_NT then
 begin
  Windows.AdjustTokenPrivileges(hToken, FALSE,tpko, sizeOf(tpko), tkp,
Buffer);
  Windows.CloseHandle(hToken);
 end;
 end;
// дані
 function FPorts: TList;
 var
 sLocation, s: String;
```

```
  idx, i, iPrev, iPosSlash: integer;
  sl: TStringList;
  begin
  if FFPorts = nil then
  begin
   FFPorts:=TList.Create;
   SetLength(sLocation, MAX_PATH);
   SetLength(sLocation, GetWindowsDirectory(pchar(sLocation), MAX_PATH));
   sLocation:=IncludeTrailingSlash(sLocation);
   if Win64Platform = VER_PLATFORM_WIN64_NT then begin
     sLocation:=sLocation + 'system32\drivers\etc\';
   end;
   sl:=TStringList.Create;
   try
     sl.LoadFromFile(sLocation + 'services');
     iPrev:=0;
     for idx:=0 to sl.Count - 1 do
     begin
       s:=sl[idx];
       iPosSlash:=FilterPos('/', s);
if (iPosSlash > 0) and (not (FilterPos('#', s) in [1..iPosSlash])) then
       begin
         i:=iPosSlash;
         repeat
           dec(i);
           if i = 0 then begin
             raise EFCorruptServicesFile.CreateFmt(RSCorruptServicesFile,
 [sLocation + 'services']);
           end;
         until s[i] in WhiteSpace;
         i:=StrToInt(Copy(s, i+1, iPosSlash-i-1));
         if i <> iPrev then begin
           FFPorts.Add(TObject(i));
         end;
         iPrev:=i;
       end;
     end;
   finally
     sl.Free;
   end;
  end;
  Result:=FFPorts;
  end;

  function FetchCaseInsensitive(var AInput: string; const ADelim:
  string = FFetchDelimDefault; const ADelete: Boolean =
 FFetchDeleteDefault): String;
  var
  LPos: integer;
  begin
  if ADelim = #0 then begin
   LPos:=Pos(ADelim, AInput);
  end else begin
   LPos:=FilterPos(UpperCase(ADelim), UpperCase(AInput));
  end;
  if LPos = 0 then begin
   Result:=AInput;
   if ADelete then begin
     AInput:='';
   end;
  end else begin
   Result:=Copy(AInput, 1, LPos - 1);
   if ADelete then begin
     AInput:=Copy(AInput, LPos + Length(ADelim), MaxInt);
   end;
  end;
  end;
```

```
 function Fetch(var AInput: string; const ADelim: string =
FFetchDelimDefault; const ADelete: Boolean = FFetchDeleteDefault;
 const ACaseSensitive: Boolean = FFetchCaseSensitiveDefault): String;
 var
 LPos: integer;
 begin
 if ACaseSensitive then begin
  if ADelim = #0 then begin
    LPos:=Pos(ADelim, AInput);
  end else begin
    LPos:=FilterPos(ADelim, AInput);
  end;
  if LPos = 0 then begin
    Result:=AInput;
    if ADelete then begin
      AInput:='';
    end;
  end
  else begin
    Result:=Copy(AInput, 1, LPos - 1);
    if ADelete then begin
      AInput:=Copy(AInput, LPos + Length(ADelim), MaxInt);
    end;
  end;
 end else begin
  Result:=FetchCaseInsensitive(AInput, ADelim, ADelete);
 end;
 end;
 begin
 for Result:=Low(Contents) to High(Contents) do begin
  if CaseSensitive then begin
    if SearchStr = Contents[Result] then begin
      Exit;
    end;
  end else begin
    if ANSISameText(SearchStr, Contents[Result]) then begin
      Exit;
    end;
  end;
 end;
 Result:=Low(Contents) to High(Contents) do
 end;
// поточний потік
 function IsCurrentThread(AThread: TThread): boolean;
 begin
 result:=AThread.ThreadID = GetCurrentThreadID;
 end;

 function IsNumeric(AChar: char): Boolean;
 begin
 Result:=AChar in ['0'..'9'];
 end;

 function IsNumeric(const AString: string): Boolean;
 var
 LCode: Integer;
 LVoid: Integer;
 begin
 Val(AString, LVoid, LCode);
 Result:=LCode = 0;
 end;

// перетворення
 function StrToDay(const ADay: string): Byte;
 begin
 Result:=Succ(PosInStrArray(Uppercase(ADay),
  ['SUN','MON','TUE','WED','THU','FRI','SAT']));
 end;
// перетворення
```

```
 function StrToMonth(const AMonth: string): Byte;
 begin
 Result:=Succ(PosInStrArray(Uppercase(AMonth),
  ['JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC']));
 end;

 function UpCaseFirst(const AStr: string): string;
 begin
 Result:=LowerCase(TrimLeft(AStr));
 if Result <> '' then begin
  Result[1]:=UpCase(Result[1]);
 end;
 end;

function DateTimeToGmtOffSetStr(ADateTime: TDateTime; SubGMT: Boolean):
                                  string;
 var
 AHour, AMin, ASec, AMSec: Word;
 begin
 if (ADateTime = 0.0) and SubGMT then
 begin
  Result:='GMT';
  Exit;
 end;
 DecodeTime(ADateTime, AHour, AMin, ASec, AMSec);
 Result:=Format(' %0.2d%0.2d', [AHour, AMin]);
 if ADateTime < 0.0 then
 begin
  Result[1]:='-';
 end
 else
 begin
  Result[1]:='+';
 end;
 end;
 procedure BuildMIMETypeMap(dest: TStringList);
 begin
 raise EFException.Create('BuildMIMETypeMap not implemented yet.');
 end;
 var
 Reg: TRegistry;
 slSubKeys: TStringList;
 i: integer;
 begin
 Reg:=CreateTRegistry; try
  Reg.RootKey:=HKEY_CLASSES_ROOT;
  Reg.OpenKeyreadOnly('\MIME\Database\Content Type');
  slSubKeys:=TStringList.Create;
  try
    Reg.GetKeyNames(slSubKeys);
    reg.Closekey;
    for i:=0 to slSubKeys.Count - 1 do
    begin
Reg.OpenKeyreadOnly('\MIME\Database\Content Type\' + slSubKeys[i]);
      dest.Append(LowerCase(reg.ReadString('Extension')) + '=' +
slSubKeys[i]);
      Reg.CloseKey;
    end;
  finally
    slSubKeys.Free;
  end;
 finally
  reg.free;
 end;
 end;
// робота з пам'ятю
 function GetMIMETypeFromFile(const AFile: TFileName): string;
 var
 MIMEMap: TFMIMETable;
```

```
 begin
 MIMEMap:=TFMimeTable.Create(true);
 try
  result:=MIMEMap.GetFileMIMEType(AFile);
 finally
  MIMEMap.Free;
 end;
 end;

 function GmtOffsetStrToDateTime(S: string): TDateTime;
 begin
 Result:=0.0;
 S:=Copy(Trim(s), 1, 5);
 if Length(S) > 0 then
 begin
  if s[1] in ['-', '+'] then
  begin
    try
 Result:=EncodeTime(StrToInt(Copy(s,2,2)),StrToInt(Copy(s,4,2)),0,0);
      if s[1] = '-' then
      begin
        Result:=-Result;
      end;
    except
      Result:=0.0;
    end;
  end;
 end;
 end;
```

```
// поточні часові значення
 function GMTToLocalDateTime(S: string): TDateTime;
 var
 DateTimeOffset: TDateTime;
 begin
 Result:=RawStrInternetToDateTime(S);
 if Length(S) < 5 then begin
  DateTimeOffset:=0.0
 end else begin
  DateTimeOffset:=GmtOffsetStrToDateTime(S);
 end;
 if DateTimeOffset < 0.0 then begin
  Result:=Result + Abs(DateTimeOffset);
 end else begin
  Result:=Result - DateTimeOffset;
 end;
 Result:=Result + OffSetFromUTC;
 end;
 procedure Sleep(ATime: cardinal);
 begin
 if (not Assigned(GStack)) then begin
  GStack:=TFStack.CreateStack;
 end;
 GStack.WSSelect(nil, nil, nil, ATime);
 Windows.Sleep(ATime);
 end;
 var
 i: Integer;
 begin
 SetLength(result, 32);
 for i:=1 to 32 do
 begin
  if ((Value shl (i-1)) shr 31) = 0 then
    result[i]:='0'
  else
    result[i]:='1';
```

```
 end;
 end;

// поточний поток плагіна
 function CurrentProcessF: TFPID;
 begin
 Result:=getpid;
 Result:=GetCurrentProcessID;
 end;

// потік
 function InMainThread: boolean;
 begin
 Result:=GetCurrentThreadID = MainThreadID;
 end;

 procedure LoadMIME(const AFileName : String; AMIMEList : TStringList);
 var
 KeyList: TStringList;
 i, p: Integer;
 s, LMimeType, LExtension: String;
 begin
 If FileExists(AFileName) Then
 Begin
  KeyList:=TStringList.Create;
  try
    KeyList.LoadFromFile(AFileName);
    for i:=0 to KeyList.Count -1 do begin
      s:=KeyList[i];
      p:=FilterPos('#', s);
      if (p>0) then
      begin
        setlength(s, p-1);
      end;
      if s <> '' then
      begin
        s:=Trim(s);
        LMimeType:=Fetch(s);
        if LMimeType <> '' then
        begin
           while (s<>'') do
           begin
             LExtension:=Fetch(s);
             if LExtension <> '' then
             try
               AMIMEList.Values['.'+LExtension]:= LMimeType;
             except
               on EListError do {ignore} ;
             end;
           end;
        end;
      end;
    end;
  except
    on EFOpenError do {ignore} ;
  end;
 End;
 end;

 procedure FillMimeTable(AMIMEList : TStringList);
 var
 reg: TRegistry;
 KeyList: TStringList;
 i: Integer;
 s: String;
 begin
 if not Assigned(AMIMEList) then
 begin
  Exit;
```

```
end;
if AMIMEList.Count > 0 then
begin
 Exit;
end;
AMIMEList.Duplicates:=dupError;
with AMIMEList do
begin
// РЕСУРСИ
 Add('.aiff=audio/x-aiff');
 Add('.au=audio/basic');
 Add('.mid=midi/mid');
 Add('.mp3=audio/x-mpg');
 Add('.m3u=audio/x-mpegurl');
 Add('.qcp=audio/vnd.qcelp');
 Add('.ra=audio/x-realaudio');
 Add('.wav=audio/x-wav');
 Add('.gsm=audio/x-gsm');
 Add('.wax=audio/x-ms-wax');
 Add('.wma=audio/x-ms-wma');
 Add('.ram=audio/x-pn-realaudio');
 Add('.mjf=audio/x-vnd.AudioExplosion.MjuiceMediaFile');
 { Image }
 Add('.bmp=image/bmp');
 Add('.gif=image/gif');
 Add('.jpg=image/jpeg');
 Add('.jpeg=image/jpeg');
 Add('.jpe=image/jpeg');
 Add('.pict=image/x-pict');
 Add('.png=image/x-png');
 Add('.svg=image/svg-xml');
 { Text }
 Add('.323=text/h323');
 Add('.xml=text/xml');
 Add('.uls=text/iuls');
 Add('.txt=text/plain');
 Add('.rtx=text/richtext');
 Add('.wsc=text/scriptlet');
 Add('.rt=text/vnd.rn-realtext');
 Add('.htt=text/webviewhtml');
 Add('.htc=text/x-component');
 Add('.vcf=text/x-vcard');
 { video/ }
 Add('.avi=video/x-msvideo');
 Add('.flc=video/flc');
 Add('.mpeg=video/x-mpeg2a');
 Add('.mov=video/quicktime');
 Add('.rv=video/vnd.rn-realvideo');
 Add('.ivf=video/x-ivf');
 Add('.movie=video/x-sgi-movie');
 { application/ }
 Add('.wmd=application/x-ms-wmd');
 Add('.wms=application/x-ms-wms');
 Add('.wmz=application/x-ms-wmz');
 Add('.p12=application/x-pkcs12');
 Add('.p7b=application/x-pkcs7-certificates');
 Add('.p7r=application/x-pkcs7-certreqresp');
 Add('.qtl=application/x-quicktimeplayer');
 Add('.rtsp=application/x-rtsp');
 Add('.swf=application/x-shockwave-flash');
 Add('.sit=application/x-stuffit');
 Add('.tar=application/x-tar');
 Add('.man=application/x-troff-man');
 Add('.urls=application/x-url-list');
 Add('.zip=application/x-zip-compressed');
 Add('.cdf=application/x-cdf');
end;
Reg:=CreateTRegistry; try
 KeyList:=TStringList.create;
```

```
  try
    Reg.RootKey:=HKEY_CLASSES_ROOT;
    if Reg.OpenKeyReadOnly('\') then
    begin
      Reg.GetKeyNames(KeyList);
    end;
    for i:=0 to KeyList.Count - 1 do
    begin
      if Copy(KeyList[i], 1, 1) = '.' then
      begin
        if reg.OpenKeyReadOnly(KeyList[i]) then
        begin
          s:=Reg.ReadString('Content Type');
        if Reg.ValueExists('Content Type') then
        begin
         FFileExt.Values[KeyList[i]]:=Reg.ReadString('Content Type');
        end;
          if Length(s) > 0 then
          begin
            AMIMEList.Values[KeyList[i]]:=s;
          end;
        end;
      end;
    end;
    if Reg.OpenKeyreadOnly('\MIME\Database\Content Type') then
    begin
      KeyList.Clear;
      Reg.GetKeyNames(KeyList);
      reg.Closekey;
      for i:=0 to KeyList.Count - 1 do
      begin
        if Reg.OpenKeyreadOnly('\MIME\Database\Content Type\' + KeyList[i])
then
        begin
          s:=reg.ReadString('Extension');
          AMIMEList.Values[s]:=KeyList[i];
          Reg.CloseKey;
        end;
      end;
    end;
  finally
    KeyList.Free;
  end;
 finally
  reg.free;
 end;
 end;
 procedure TFMimeTable.AddMimeType(const Ext, MIMEType: string);
 var
 LExt,
 LMIMEType: string;
 begin
 LExt:=AnsiLowerCase(Ext);
 if Length(LExt) = 0 then
 begin
  raise EFException.Create(RSMIMEExtensionEmpty);
 end
 else
 begin
 if LExt[1] <> '.' then
 begin
   LExt:='.' + LExt;
 end;
 end;
 LMIMEType:=AnsiLowerCase(MIMEType);
 if Length(LMIMEType) = 0 then
  raise EFException.Create(RSMIMEMIMETypeEmpty);
 if FFileExt.IndexOf(LExt) = -1 then
 begin
```

```
 FFileExt.Add(LExt);
 FMIMEList.Add(LMIMEType);
end
else
 raise EFException.Create(RSMIMEMIMEExtAlreadyExists);
end;
procedure TFMimeTable.BuildCache;
begin
if Assigned(FOnBuildCache) then
begin
 FOnBuildCache(Self);
end
else
begin
 if FFileExt.Count = 0 then
 begin
   BuildDefaultCache;
 end;
end;
end;

procedure TFMimeTable.BuildDefaultCache;
var LKeys : TStringList;
begin
LKeys:=TStringList.Create;
try
 FillMIMETable(LKeys);
 LoadFromStrings(LKeys);
finally
 FreeAndNil(LKeys);
end;
end;

constructor TFMimeTable.Create(Autofill: boolean);
begin
FFileExt:=TStringList.Create;
FFileExt.Sorted:=False;
FMIMEList:=TStringList.Create;
FMIMEList.Sorted:=False;
if Autofill then begin
 BuildCache;
end;
end;
destructor TFMimeTable.Destroy;
begin
FreeAndNil(FMIMEList);
FreeAndNil(FFileExt);
inherited Destroy;
end;

function TFMimeTable.getDefaultFileExt(const MIMEType: string): String;
var
Index : Integer;
LMimeType: string;
begin
Result:='';
LMimeType:=AnsiLowerCase(MIMEType);
Index:=FMIMEList.IndexOf(LMimeType);
if Index <> -1 then
begin
 Result:=FFileExt[Index];
end
else
begin
 BuildCache;
 Index:=FMIMEList.IndexOf(LMIMEType);
 if Index <> -1 then
   Result:=FFileExt[Index];
end;
```

```
 end;

 function TFMimeTable.GetFileMIMEType(const AFileName: string): string;
 var
 Index : Integer;
 LExt: string;
 begin
 LExt:=AnsiLowerCase(ExtractFileExt(AFileName));
 Index:=FFileExt.IndexOf(LExt);
 if Index <> -1 then
 begin
  Result:=FMIMEList[Index];
 end
 else
 begin
  BuildCache;
  Index:=FFileExt.IndexOf(LExt);
  if Index = -1 then
  begin
    Result:='application/octet-stream'
  end
  else
  begin
    Result:=FMIMEList[Index];
  end;
 end;
 end;

 procedure TFMimeTable.LoadFromStrings(AStrings: TStrings;const MimeSeparator:
Char = '=');
 var
 I   : Integer;
 Ext : string;
 begin
 FFileExt.Clear;
 FMIMEList.Clear;
 for I:=0 to AStrings.Count - 1 do
 begin
  Ext:=AnsiLowerCase(Copy(AStrings[I], 1, Pos(MimeSeparator, AStrings[I]) -
1));
  if Length(Ext) > 0 then
    if FFileExt.IndexOf(Ext) = -1 then
      AddMimeType(Ext, Copy(AStrings[I], Pos(MimeSeparator, AStrings[I]) + 1,
Length(AStrings[I])));
 end;
 end;

 procedure TFMimeTable.SaveToStrings(AStrings: TStrings;
 const MimeSeparator: Char);
 var
 I : Integer;
 begin
 AStrings.Clear;
 for I:=0 to FFileExt.Count - 1 do
  AStrings.Add(FFileExt[I] + MimeSeparator + FMIMEList[I]);
 end;
 procedure SetThreadPriority(AThread: TThread; const APriority:
TFThreadPriority; const APolicy: Integer = -MaxInt);
 begin
 if (getpriority(PRIO_PROCESS, 0) < APriority) or (geteuid = 0) then begin
  setpriority(PRIO_PROCESS, 0, APriority);
 end;
 AThread.Priority:=APriority;
 end;

 function SBPos(const Substr, S: string): Integer;
 begin
 Result:=Pos(Substr, S);
 end;
```

```
// позиція
 function MemoryPos(const ASubStr: String; MemBuff: PChar; MemorySize:
Integer): Integer;
 var
 LSearchLength: Integer;
 LS1: Integer;
 LChar: Char;
 LPS,LPM: PChar;
 begin
 LSearchLength:=Length(ASubStr);
 if (LSearchLength = 0) or (LSearchLength > MemorySize) then begin
  Result:=0;
  Exit;
 end;
 LChar:=PChar(Pointer(ASubStr))^; // перший символ
 LPS:=PChar(Pointer(ASubStr))+1;// Строковий параметр string
 LPM:=MemBuff;
 LS1:=LSearchLength-1;
 LSearchLength:=MemorySize-LS1;//Проводиться MemorySize-LS+1
 if LS1=0 then begin //оптимізація
  while LSearchLength>0 do begin
    if LPM^= LChar then begin
      Result:=LPM-MemBuff+1;
      EXIT;
    end;
    inc(LPM);
    dec(LSearchLength);
  end;//while
 end else begin
  while LSearchLength>0 do begin
    if LPM^= LChar then begin
      inc(LPM);
      if CompareMem(LPM,LPS,LS1) then begin
        Result:=LPM-MemBuff;
        EXIT;
      end;
    end
    else begin
      inc(LPM);
    end;
    dec(LSearchLength);
  end;
 end;
 Result:=0;
 End;

 // фільтрація
 function FilterGetHostName: string;
 var
 LHost: array[1..255] of Char;
 i: LongWord;
 begin
 if GetHostname(@LHost[1], 255) <> -1 then begin
  i:=FilterPos(#0, LHost);
  SetLength(Result, i - 1);
  Move(LHost, Result[1], i - 1);
 end;
 end;

 function FilterGetHostName: string;
 var
 i: LongWord;
 begin
 SetLength(Result, MAX_COMPUTERNAME_LENGTH + 1);
 i:=Length(Result);
 if GetComputerName(@Result[1], i) then begin
  SetLength(Result, i);
 end;
```

```
 end;

 function IsValidIP(const S: String): Boolean;
 var
 j, i: Integer;
 LTmp: String;
 begin
 Result:=True;
 LTmp:=Trim(S);
 for i:=1 to 4 do begin
  j:=StrToIntDef(Fetch(LTmp, '.'), -1);
  Result:=Result and (j > -1) and (j < 256);
  if NOT Result then begin
    Break;
  end;
 end;
 end;

// Це ім’я хоста
 function IsHostname(const S: String): Boolean;
 begin
 Result:=((FilterPos('.', S) = 0) or (S[1] <> '.')) and NOT IsValidIP(S);
 end;

// Це кореневий хост
 function IsTopDomain(const AStr: string): Boolean;
 Var
 i: Integer;
 S1,LTmp: String;
 begin
 i:=0;
 LTmp:=AnsiUpperCase(Trim(AStr));
 while FilterPos('.', LTmp) > 0 do begin
  S1:=LTmp;
  Fetch(LTmp, '.');
  i:=i + 1;
 end;
 Result:=((Length(LTmp) > 2) and (i = 1));
 if Length(LTmp) = 2 then begin  // Ім’я
  S1:=Fetch(S1, '.');
  if LTmp = 'UK' then begin
    if S1 = 'CO' then result:=i = 2;
    if S1 = 'COM' then result:=i = 2;
  end;
  if LTmp = 'TW' then begin
    if S1 = 'CO' then result:=i = 2;
    if S1 = 'COM' then result:=i = 2;
  end;
 end;
 end;

// Це хост
 function IsDomain(const S: String): Boolean;
 begin
 Result:=NOT IsHostname(S) and (FilterPos('.', S) > 0) and NOT IsTopDomain(S);
 end;
// Ім’я хоста
 function DomainName(const AHost: String): String;
 begin
 result:=Copy(AHost, FilterPos('.', AHost), Length(AHost));
 end;

 function IsFQDN(const S: String): Boolean;
 begin
 Result:=IsHostName(S) and IsDomain(DomainName(S));
 end;
// прогресс
 function ProcessPath(const ABasePath: string;
 const APath: string;
```

```
  const APathDelim: string = '/'): string;
  var
  i: Integer;
  LPreserveTrail: Boolean;
  LWork: string;
  begin
  if FilterPos(APathDelim, APath) = 1 then begin
   Result:=APath;
  end else begin
   Result:='';
   LPreserveTrail:=(Copy(APath, Length(APath), 1) = APathDelim) or
 (Length(APath) = 0);
   LWork:=ABasePath;
   if (Length(LWork) > 0) and (Copy(LWork, Length(LWork), 1) <> APathDelim)
 then begin
     LWork:=LWork + APathDelim;
   end;
   LWork:=LWork + APath;
   if Length(LWork) > 0 then begin
     i:=1;
     while i <= Length(LWork) do begin
       if LWork[i] = APathDelim then begin
         if i = 1 then begin
           Result:=APathDelim;
         end else if Copy(Result, Length(Result), 1) <> APathDelim then begin
           Result:=Result + LWork[i];
         end;
       end else if LWork[i] = '.' then begin
 if (Copy(Result, Length(Result), 1) = APathDelim) and (Copy(LWork, i, 2) =
 '..') then begin
           Delete(Result, Length(Result), 1);
  while (Length(Result) > 0) and (Copy(Result, Length(Result), 1) <>
        APathDelim) do begin
            Delete(Result, Length(Result), 1);
         end;
         Inc(i);
        end else begin
         Result:=Result + LWork[i];
        end;
       end else begin
         Result:=Result + LWork[i];
       end;
       Inc(i);
     end;
   end;
 if (Result <> APathDelim) and (Copy(Result, Length(Result), 1) = APathDelim)
     and (LPreserveTrail = False) then begin
      Delete(Result, Length(Result), 1);
    end;
  end;
  end;
 constructor TFLocalEvent.Create(const AInitialState: Boolean = False;
  const AManualReset: Boolean = False);
  begin
  inherited Create(nil, AManualReset, AInitialState, '');
  end;

function TFLocalEvent.WaitFor: TWaitResult;
  begin
  Result:=WaitFor(Infinite);
  end;
 // запит
 function iif(ATest: Boolean; const ATrue: Integer; const AFalse: Integer):
 Integer;
  begin
  if ATest then begin
    Result:=ATrue;
  end else begin
    Result:=AFalse;
```

```
 end;
 end;
// запит, стандартні параметри
function iif(ATest: Boolean; const ATrue: string; const AFalse: string):
string;
 begin
 if ATest then begin
   Result:=ATrue;
 end else begin
   Result:=AFalse;
 end;
 end;

 // запит, розширені параметри
 function iif(ATest: Boolean; const ATrue: Boolean; const AFalse: Boolean):
Boolean;
 begin
 if ATest then begin
   Result:=ATrue;
 end else begin
   Result:=AFalse;
 end;
 end;

 procedure TDStream.SetPointer(Ptr: Pointer; Size: Integer);
 Begin
 inherited SetPointer(Ptr, Size);
 Seek(0,0); Position:=0;
 End;


function TDStream.Write(const Buffer; Count: Integer): Longint;
 begin
 Result:=0;
 End;


function  AnsiPosFx_ (const ASubStr: AnsiString; AStr: PChar; L1: Cardinal;
AStartPos: Cardinal=0): Cardinal;
 var
 L2: Cardinal;
 ByteType : TMbcsByteType;
 Str, SubStr, CurResult: PChar;
 Begin
 Result:= 0;
 // не знайдено
 L1:=Length(AStr);
 L2:=Length(ASubStr);
 if (L2=0) or (L2>L1) then Exit;
 Str:=Pointer(AStr);
 SubStr:=Pointer(ASubStr);
 if AStartPos>0 then begin
   Str:=Str + AStartPos - 1;
   L1 :=L1 + 1 - AStartPos;
 end;
 if L1<=0 then EXIT;
 CurResult:=StrPos(Str, SubStr);
 while (CurResult <> nil) and ((L1 - Cardinal(CurResult - Str)) >= L2) do
begin
   ByteType:=StrByteType(Str, Integer(CurResult-Str));
   if (ByteType <> mbTrailByte) and
     (Windows.CompareString(LOCALE_USER_DEFAULT, 0, CurResult, L2, SubStr, L2)
= 2) then begin
     Result:=CurResult-Pointer(AStr)+1;
     Exit;
   end;
   if (ByteType = mbLeadByte) then Inc(Result);
   if (ByteType <> mbTrailByte) and
     (strncmp(CurResult, SubStr, L2) = 0) then begin
```

```
      Result:=CurResult-Pointer(AStr)+1;
      Exit;
    end;
    Inc(Result);
    CurResult:=StrPos(CurResult, SubStr);
  end;
  End;

function  AnsiPosFx(const ASubStr,AStr: AnsiString; AStartPos: Cardinal=0):
Cardinal;
  Begin
  Result:=AnsiPosFx_(ASubStr, Pointer(AStr), Length(AStr), AStartPos);
  End;
// позиція
  function  AnsiMemoryPos(const ASubStr: String; MemBuff: PChar; MemorySize:
Integer): Integer;
  Begin
  Result:=AnsiPosFx_(ASubStr, MemBuff, MemorySize, 0);
  End;
// пошук
  function  PosFx (const ASubStr,AStr: AnsiString; AStartPos: Cardinal):
Cardinal;
  var
  lpSubStr,lpS: PChar;
  LenSubStr,LenS: Integer;
  LChar: Char;
  Begin
  LenSubStr:=Length(ASubStr);
  LenS:=Length(AStr);
  if (LenSubStr=0) or (LenSubStr>LenS) then begin
    Result:=0;
    EXIT;
  end;
  lpSubStr:=Pointer(ASubStr);
  lpS:=Pointer(AStr);
  if AStartPos>0 then begin
    lpS:=lpS+AStartPos-1;
    LenS:=LenS+1-Integer(AStartPos);
  end;
  LChar :=lpSubStr[0];
  lpSubStr:=lpSubStr  +1;
  LenSubStr:=LenSubStr-1;
  LenS:=LenS-LenSubStr;
  if LenS<=0 then begin
    Result:=0;
    EXIT;
  end;
  while LenS>0 do begin
    if lpS^= LChar then begin
      inc(lpS);
      if CompareMem(lpS,lpSubStr,LenSubStr) then begin
        Result:=lpS-Pointer(AStr);//+1 already here
        EXIT;
      end;
    end
    else begin
      inc(lpS);
    end;
    dec(LenS);
  end;
  Result:=0;
  End;
// допоміжна
  function MakeMethod (DataSelf, Code: Pointer): TMethod;
  Begin
  Result.Data:=DataSelf;
  Result.Code:=Code;
  End;
```

```
Initialization

GStackClass:=TFStack;
ATempPath:=TempPath;
GStackClass:=TFStackWindows;
if LeadBytes = [] then begin
  FilterPos:=SBPos;
end else begin
  FilterPos:=AnsiPos;
end;

SetLength(FilterFalseBoolStrs, 1);
FilterFalseBoolStrs[Low(FilterFalseBoolStrs)]:='FALSE';
SetLength(FilterTrueBoolStrs, 1);
FilterTrueBoolStrs[Low(FilterTrueBoolStrs)]:='TRUE';
finalization
FreeAndNil(FFPorts);
end.
```

ОПИС ІНТЕРФЕЙСУ КОРИСТУВАЧА *.XUL
(USER-INTERFACE LANGUAGE)

```
<?xml version="1.0"?>
 <overlay id="FISHING" xmlns=" keymaster/gatekeeper/there.is.only.xul">
   <menupopup id="menu_ToolsPopup">
   <menuitem label="FISHING" position="1" />
   <menuitem label="About" position="2"/>
   </menupopup>
<?xml version="1.0"?>
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<?xml-stylesheet href="findfile.css" type="text/css"?>
<!ENTITY findWindow.title "Find Files">
<!ENTITY fileMenu.label "File">
<!ENTITY editMenu.label "Edit">
<!ENTITY fileMenu.accesskey "f">
<!ENTITY editMenu.accesskey "e">
<!ENTITY openCmd.label "Open Search...">
<!ENTITY saveCmd.label "Save Search...">
<!ENTITY closeCmd.label "Close">
<!ENTITY openCmd.accesskey "o">
<!ENTITY saveCmd.accesskey "s">
<!ENTITY closeCmd.accesskey "c">
<!ENTITY cutCmd.label "Cut">
<!ENTITY copyCmd.label "Copy">
<!ENTITY pasteCmd.label "Paste">
<!ENTITY cutCmd.accesskey "t">
<!ENTITY copyCmd.accesskey "c">
<!ENTITY pasteCmd.accesskey "p">
<!ENTITY cutCmd.commandkey "X">
<!ENTITY copyCmd.commandkey "C">
<!ENTITY pasteCmd.commandkey "V">
<!ENTITY openCmdToolbar.label "Open">
<!ENTITY saveCmdToolbar.label "Save">
<!ENTITY searchTab "Search">
<!ENTITY optionsTab "Options">
<!ENTITY findDescription "Введіть дані">
<!ENTITY findCriteria "Кріерії пошуку">
<!ENTITY type.name "Name">
<!ENTITY type.size "Size">
<!ENTITY type.date "Час модифікації ">
<!ENTITY mode.is "Is">
<!ENTITY mode.isnot "Is Not">
<!ENTITY casesensitive "Case Sensitive Search">
<!ENTITY matchfilename "Match Entire Filename">
<!ENTITY results.filename "Filename">
<!ENTITY results.location "Location">
<!ENTITY results.size "Size">
<!ENTITY bytes.before "">
<!ENTITY bytes.after "bytes">
<!ENTITY button.find "Find">
<!ENTITY button.cancel "Cancel">
<window
  id="findfile-window"
  title="&findWindow.title;"
  persist="screenX screenY width height"
  orient="horizontal"
<script src="findfile.js"/>
<popupset>
   <popup id="editpopup">
     <menuitem label="Cut" accesskey="&cutCmd.accesskey;"/>
     <menuitem label="Copy" accesskey="&copyCmd.accesskey;"/>
<menuitem label="Paste"  accesskey="&pasteCmd.accesskey;"disabled="true"/>
   </popup>
</popupset>
<keyset>
   <key id="cut_cmd" modifiers="accel" key="&cutCmd.commandkey;"/>
   <key id="copy_cmd" modifiers="accel" key="&copyCmd.commandkey;"/>
   <key id="paste_cmd" modifiers="accel" key="&pasteCmd.commandkey;"/>
```

```
   <key id="close_cmd" keycode="VK_ESCAPE" oncommand="window.close();"/>
</keyset>
<vbox flex="1">
 <toolbox>
  <menubar id="findfiles-menubar">
    <menu id="file-menu" label="&fileMenu.label;"
        accesskey="&fileMenu.accesskey;">
      <menupopup id="file-popup">
        <menuitem label="&openCmd.label;"
                  accesskey="&openCmd.accesskey;"/>
        <menuitem label="&saveCmd.label;"
                  accesskey="&saveCmd.accesskey;"/>
        <menuseparator/>
        <menuitem label="&closeCmd.label;"

accesskey="&closeCmd.accesskey;" key="close_cmd" oncommand="window.close();"/>
      </menupopup>
    </menu>
    <menu id="edit-menu" label="&editMenu.label;"
          accesskey="&editMenu.accesskey;">
      <menupopup id="edit-popup">
        <menuitem label="&cutCmd.label;"
                  accesskey="&cutCmd.accesskey;" key="cut_cmd"/>
        <menuitem label="&copyCmd.label;"
                  accesskey="&copyCmd.accesskey;" key="copy_cmd"/>
        <menuitem label="&pasteCmd.label;"
                  accesskey="&pasteCmd.accesskey;" key="paste_cmd"
disabled="true"/>
      </menupopup>
    </menu>
  </menubar>
  <toolbar id="findfiles-toolbar">
    <toolbarbutton id="opensearch" label="&openCmdToolbar.label;"/>
    <toolbarbutton id="savesearch" label="&saveCmdToolbar.label;"/>
  </toolbar>
 </toolbox>
 <tabbox>
  <tabs>
    <tab label="&searchTab;" selected="true"/>
    <tab label="&optionsTab;"/>
  </tabs>
  <tabpanels>
   <tabpanel id="searchpanel" orient="vertical" context="editpopup">
   <description>
     &findDescription;
   </description>
   <spacer class="titlespace"/>
   <groupbox orient="horizontal">
     <caption label="&findCriteria;"/>
     <menulist id="searchtype">
       <menupopup>
         <menuitem label="&type.name;"/>
         <menuitem label="&type.size;"/>
         <menuitem label="&type.date;"/>
       </menupopup>
     </menulist>
   <spacer class="springspace"/>
     <menulist id="searchmode">
       <menupopup>
         <menuitem label="&mode.is;"/>
         <menuitem label="&mode.isnot;"/>
       </menupopup>
     </menulist>
   <spacer class="springspace"/>
<menulist id="find-text" flex="1"
editable="true"
datasources="file:/// recents.rdf">
     <template>
       <menupopup>
```

```
        </menupopup>
      </template>
    </menulist>
    </groupbox>
   </tabpanel>
   <tabpanel id="optionspanel" orient="vertical">
      <checkbox id="casecheck" label="&casesensitive;"/>
      <checkbox id="wordscheck" label="&matchfilename;"/>
     </tabpanel>
  </tabpanels>
 </tabbox>
 <tree id="results" style="display: none;" flex="1">
   <treecols>
     <treecol id="name" label="&results.filename;" flex="1"/>
     <treecol id="location" label="&results.location;" flex="2"/>
     <treecol id="size" label="&results.size;" flex="1"/>
   </treecols>
   <treechildren>
     <treeitem>
       <treerow>
         <treecell label=" "/>
         <treecell label="/usr/local"/>
         <treecell label="&bytes.before;2520&bytes.after;"/>
       </treerow>
     </treeitem>
   </treechildren>
 </tree>
 <splitter id="splitbar" resizeafter="grow" style="display: none;"/>
 <spacer class="titlespace"/>
 <hbox>
<progressmeter id="progmeter" value="50%" style="display: none;"/>
   <spacer flex="1"/>
   <button id="find-button" label="&button.find;"
           oncommand="doFind()"/>
   <button id="cancel-button" label="&button.cancel;"
           oncommand="window.close();"/>
</hbox>
</vbox>
</window>
```

МОДУЛЬ ПОШУКУ ПЛАГИНІВ DELPHI.RDF
(RESOURCE DESCRIPTION FRAMEWORK)

```
<?xml version="1.0"?>
<RDF:RDF xmlns:RDF="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:chrome="/rdf/chrome#">
  <RDF:Seq about="urn:package:root">
    <RDF:li resource="urn:package:FISHING"/>
  </RDF:Seq>

  <RDF:Description about="urn:package:FISHING"
    chrome:extension="true" chrome:name="FISHING"/>

  <RDF:Seq about="urn:overlays">
    <RDF:li resource="chrome://browser/content/browser.xul"/>
  </RDF:Seq>

  <RDF:Seq about="chrome://browser/content/browser.xul">
<RDF:li>chrome://FISHING/content/FISHING-Overlay.xul</RDF:li>
  </RDF:Seq>
</RDF:RDF>
```

ФАЙЛ ПРОЕКТУ INSTALL.RDF (RESOURCE DESCRIPTION FRAMEWORK)

```
<?xml version="1.0"?>
<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"xmlns:em="">
<Description about="urn:install-manifest">
<em:creator> Davidov D.O. </em:creator>
<em:description>FISHING extention</em:description> <em:homepageURL>
</em:homepageURL>
   <em:id>{65b3130e-8513-41b6-8ea8-43dbd9cc0b12}</em:id>
   <em:name>FISHING</em:name>
   <em:version>1.0</em:version>
   <em:targetApplication>
   <Description>
   <em:id>{ec8030f7-c20a-464f-9b0e-13a3a9e97384}</em:id>
   <em:minVersion>1.0</em:minVersion>
   <em:maxVersion>1.0</em:maxVersion>
   </Description>
   </em:targetApplication>

     <em:file>
<Description about="urn:extension:file:FISHING.jar">
         <em:package>content/</em:package>
       </Description>
     </em:file>
     </Description>
</RDF>
```