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
“____” _____ 2026 р.

**ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА**
**за другим (магістерським) рівнем вищої освіти**
на тему
**“Дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування”**

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-25М
ОПП «Комп’ютерна інженерія»
спеціальності F7 «Комп’ютерна інженерія»
_____ Ісаєнко А.О.
«____» _____ 2026 р.

Керівник проекту
доктор технічних наук, професор
_____ Смірнов О.А.
«____» _____ 2026 р.

Нормоконтроль проекту
_____ Коваленко А.С.
«____» _____ 2026 р.

Рецензент _____
_____

Підписано КЕП
Центральноукраїнський національний технічний університет
Ісаєнко Артем Олександрович
Сертифікат: 18DF2C9B93CEA2040400000065254D00A4FD7B06
Дійсний з 25.11.2026 15:18:00 по 25.11.2027 15:17:59

**Центральноукраїнський національний технічний університет**
Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Рівень вищої освіти *магістр*
Галузь знань _F_____*"*Інформаційні технології*"*_____
Спеціальність F7 *"*Комп'ютерна інженерія*"*
Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 1 » вересня 2026 року

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

*Ісаєнку Артему Олександровичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування*

2. Керівник роботи *Смірнов Олексій Анатолійович, доктор технічних наук, професор*
( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 41-13 від 17.08.2026 року

3. Строк подання студентом роботи до захисту *2.12.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування*

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

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

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

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

7. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Охорона праці | Марченко К.М., к.т.н., доцент | 06.10.2026 | 16.11.2026 |
| | | | |

8. Дата видачі завдання « 1 » вересня 2026 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти | Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти | Примітка |
|---|---|---|---|
| 1. | Аналіз існуючих систем | 10.10.2026 р. | |
| 2. | Постановка задачі, оформлення ТЗ | 15.10.2026 р. | |
| 3. | Розробка моделі компонента | 20.10.2026 р. | |
| 4. | Розробка структур даних | 25.10.2026 р. | |
| 5. | Розробка алгоритмів системи | 30.10.2026 р. | |
| 6. | Програмування алгоритмів | 10.11.2026 р. | |
| 7. | Розрахунки з охорони праці та техніки безпеки | 15.11.2026 р. | |
| 8. | Оформлення ПЗ | 17.11.2026 р. | |
| 9. | Попередній захист роботи | 2.12.2026 р. | |
| | | | |

Дата видачі завдання
« 1 » вересня 2026 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 1 » вересня 2026 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

# АНОТАЦІЯ

**Ісаєнко А.О. Дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування. F7 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2026.**

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Метою розробки є дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Об'єктом дослідження є процес захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Предметом дослідження є методи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Методи дослідження базуються на методах теорії побудови комп'ютерних мереж, теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

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 Builder C++.

**Ключові слова**: комп’ютерна інженерія, захист веб-ресурсів, пасивний аналіз, мережевий трафік, опортуністичне шифрування

# ABSTRACT

**Isaienko A.O. Research and Software Implementation of a System for Protecting Web Resources Against Passive Network Traffic Analysis Using Automated Opportunistic Encryption Methods. F7 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.**

In this final qualification thesis for the second (master's) level of higher education, software has been developed for a system designed to protect web resources against passive network traffic analysis using automated opportunistic encryption methods.

The objective of this development is the research and software implementation of a system for protecting web resources against passive network traffic analysis using automated opportunistic encryption methods.

The object of the research is the process of protecting web resources against passive network traffic analysis using automated opportunistic encryption methods.

The subject of the research consists of methods for protecting web resources against passive network traffic analysis using automated opportunistic encryption methods.

The research methodology is based on principles from computer network design theory, information security theory, mathematical statistics, and software development.

The outcome of the work is a software implementation of a system for protecting web resources against passive network traffic analysis using automated opportunistic encryption methods.

During the development of the software model, an analysis of existing hardware and software tools was conducted. All components of the developed software are described in full detail.

A user-friendly interface has been developed. Instructions for using the software tools are provided.

The program can be used on PCs running the Windows 10/11 operating system.

The program was developed in the C++ Builder environment.

**Keywords:** computer engineering, web resource protection, passive analysis, network traffic, opportunistic encryption

## ВІДОМІСТЬ ПРОЕКТУ

| | | | | | |
|---|---|---|---|---|---|
| | | Текстові документи | | | |
| | | | | | |
| 1 | А4 | Пояснювальна записка | 98 | – | |
| 2 | А4 | Робоча програма | 35 | – | Додаток А |
| | | | | | |
| | | Графічні документи | | | |
| | | | | | |
| 3 | А4 | Наукова новизна | 1 | – | |
| | | | | | |
| 4 | А4 | Структурна схема | 1 | – | |
| | | | | | |
| 5 | А4 | Функціональна схема роботи системи | 1 | – | |
| | | | | | |
| 6 | А4 | Діаграма взаємодії процесів | 1 | – | |
| | | | | | |
| 7 | А4 | Блок-схема основної програми | 1 | – | |
| 8 | А4 | Блок-схема роботи підпрограми | 1 | – | |

**ЗМІСТ**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ .... 10
ВСТУП..... 11
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ ..... 14
1.1 Призначення системи..... 14
1.2 Область застосування ..... 15
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ ..... 19
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти..... 19
2.2 Обґрунтування вибору засобів для побудови системи та мови програмування ..... 33
2.3 Розгорнута постановка завдання ..... 34
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ ..... 36
3.1 Опис функціонування системи ..... 36
3.2 Розробка структурної схеми..... 42
3.3 Розробка функціональної схеми ..... 47
3.4 Розробка діаграми процесів ..... 50
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ .... 52
4.1 Розробка блок-схем та опис алгоритмів функціонування системи ..... 52
5 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ..... 73
6 НАУКОВА НОВИЗНА ..... 79

7 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ ..... 80
7.1 Вступ ..... 80
7.2 Характеристика умов праці програміста ..... 81
7.3 Розробка заходів з умов поліпшення охорони праці ..... 86
7.4 Розрахункова частина ..... 87
7.5 Висновки до розділу ..... 88
8 ОСНОВНІ ВИСНОВКИ ..... 89
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ..... 91

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

| | | |
|---|---|---|
| AH | – | автентифікуючий заголовок |
| CA | – | сертифікаційне співтовариство |
| DES | – | Data Encription Standard |
| DoS | – | атака "Відмова в обслуговуванні" |
| DOI | – | область інтерпретації |
| ESP | – | Інкапсуляція зашифрованих даних |
| HTTPS | – | зашифрований http |
| IAB | – | координаційна рада мережі Internet |
| IDS | – | система, яка автоматизує процес перегляду подій |
| IETF | – | проблемна група проектування Internet |
| IKE | – | протокол обміну ключами за замовчуванням для ISAKMP |
| IKMP | – | протоколу керування ключами прикладного рівня |
| IPsec | – | комплект протоколів захисту інформації по IP |
| ISAKMP | – | механізми узгодження атрибутів використовуваних протоколів |
| ISP | – | постачальник послуг Internet |
| MAC | – | коди на перевірку цілісності |
| MD5 | – | дайджест повідомлення |
| Oakley | – | сесійні ключі на комп'ютери мережі Інтернет |
| PFS | – | ідеальна пряма безпека |
| PRF | – | псевдовипадкова функція |
| SA | – | Security Association |
| SKIP | – | команда підготовки наступної команди |
| SPI | – | індекс параметрів безпеки |
| SPD | – | база даних політики безпеки |
| TLS | – | протокол захищених сокетів |
| TCP | – | транспортний протокол |

# ВСТУП

**Актуальність теми**. Опортуністичне шифрування дозволяє браузерам отримувати доступ до HTTP URI через зашифрований канал TLS. Воно не замінює HTTPS, але забезпечує додатковий захист для вразливих запитів.

Використовуйте HTTPS, коли потрібне як надійне шифрування, так і автентифікація. Опортуністичне шифрування HTTP забезпечує засоби ввімкнення TLS, коли це необхідно для інших протоколів, таких як HTTP/2. Воно не забезпечує тих самих показників безпеки, як HTTPS (зелений значок замка в адресному рядку більшості браузерів).

Загрози безпеці мережі стосуються ризиків і загроз, які можуть поставити під загрозу конфіденційність, цілісність і доступність даних та інформації в мережі організації.

Атака може спричинити простої системи та блокувати операції й бізнес-процеси, що призведе до втрати продуктивності та розкриття конфіденційних даних.

Окрім самої атаки, організація зазвичай стикається з довгостроковими наслідками, такими як втрата бізнесу та довіри клієнтів, а також репутаційна шкода. Людські, технічні та фінансові ресурси, необхідні для розслідування інциденту та відновлення систем, є неабиякою вагомою складністю – Forrester прогнозує, що кіберзлочинність, як очікується, коштуватиме 12 трильйонів доларів у 2026 році.

Коротше кажучи, профілактика краща за лікування, оскільки мережева безпека допомагає забезпечити безперервність бізнесу та запобігти фінансовим втратам. Нездатність захистити дані може мати ще більші наслідки, якщо врахувати штрафи та судові позови.

Загрози надходять з багатьох джерел: комп’ютери, сервери, програмні додатки, API тощо. Захист від зловмисників включає низку найкращих практик та інструментів для контролю та покращення безпеки робочого простору, які ми

розглянемо трохи пізніше.

Ці атаки можуть бути зловмисними спробами проникнення в мережу організації, і вони можуть бути:

– Пасивно, шляхом моніторингу або крадіжки інформації без її компрометації

– Активний, шляхом компрометації мережевого обладнання, серверів або робочих станцій, а також викрадання та/або зміни даних або порушення діяльності.

**Мета й завдання дослідження.** Метою роботи є дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

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

– Огляд існуючих систем захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– Дослідження системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– Програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Об’єктом дослідження* є процес захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Предметом дослідження* є методи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Методи дослідження* базуються на методах теорії побудови комп’ютерних мереж, теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

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

– Удосконалено метод захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– Розроблено вітчизняний продукт захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування, який має більш широкі можливості, на відміну від існуючих аналогів.

**Практична цінність отриманих результатів** полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

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

Робота апробована на LVIII науково-технічній конференції здобувачів вищої освіти «Наука-виробництву», основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №17.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

# 1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

## 1.1 Призначення системи

Перш ніж детальніше розглядати типи атак, які можуть поставити під загрозу безпеку мережі, важливо пам’ятати, що вразливості безпеки мережі становлять серйозну загрозу для робочого простору.

**Вразливості, що впливають на безпеку мережі**

Зловмисники можуть використовувати вразливість для отримання несанкціонованого доступу, запуску атак та поширення в мережу організації. Як зазначено вище, сегментація мережі є однією з найкращих практик для запобігання наслідкам атаки, оскільки така сегментація допомагає обмежити горизонтальне переміщення зловмисників по інформаційній системі після того, як їм вдається проникнути в неї.

Вразливості можуть знаходитися в програмному забезпеченні робочих станцій або серверів, мережевому обладнанні або зовнішніх пристроях. Також слід враховувати тіньові IT, оскільки персональні пристрої або інструменти, як правило, менш безпечні, ніж кінцеві точки інформаційних систем.

Точніше, вектори атак можуть містити застаріле програмне забезпечення, вразливі веб-сервіси, слабкі політики паролів, неправильні налаштування конфігурації (інструменти керування ідентифікацією, брандмауер тощо) або незахищені API, і можуть призвести до атак віддаленого виконання коду ( RCE ) та SQL-ін'єкцій. Вони також можуть виходити від самих користувачів через фішинг та соціальну інженерію.

Будь-яка з цих вразливостей є точкою входу для зловмисників для отримання несанкціонованого доступу до даних і систем. Саме тому оновлення та виправлення мають вирішальне значення для запобігання використанню вразливостей зловмисниками та мінімізації поверхні атаки.

## 1.2 Область застосування

### Типи атак, що порушують безпеку мережі

На мережу можуть бути спрямовані різні види атак, зокрема шкідливі програми (віруси, черв'яки, трояни, шпигунські програми тощо), які можуть зберігатися в системі для пошкодження або крадіжки даних, зараження пристроїв, шпигунства, порушення роботи або, зрештою, для вимоги викупу. Шкідливе програмне забезпечення може бути спрямоване на робочі станції та сервери.

Вони можуть поширюватися через використання вразливостей, заражених веб-сайтів, фішингових електронних листів, USB-пристроїв, зламаного програмного забезпечення тощо.

Вони часто утворюють платформу для зловмисників, де вони можуть виконувати шкідливі дії та переміщатися в скомпрометованому середовищі.

Давайте тепер заглибимося в різні загрози безпеці мережі.

### Програми-вимагачі

Програми-вимагачі можуть атакувати системи, мережі, робочі станції та сервери. Зловмисники можуть проникнути в мережу організації після викрадення облікових даних, щоб отримати доступ до конфіденційних пристроїв і даних. Потім зловмисники можуть спробувати викрасти дані, зробити їх недоступними, погрожувати їх витоком або навіть знищити, якщо жертва не сплатить викуп – або кілька викупів.

Інструменти захисту: обізнаність, надійні паролі та багатофакторна автентифікація (MFA), сегментація мережі для обмеження латерального переміщення зловмисника по мережі, брандмауери для моніторингу трафіку та блокування несанкціонованого доступу, а також інструменти виявлення вторгнень (такі як EDR) на основі поведінкового аналізу.

### Фішинг

Фішинг – це метод початкового доступу, що використовує соціальну інженерію, щоб спонукати користувача відкрити файл, вкладений до

електронного листа, або натиснути на посилання, яке встановить шкідливе програмне забезпечення на його пристрій. Двері широко відчинені, навіть якщо користувач вважає, що виконав цілком звичайну дію: зловмисник може отримати доступ до облікових даних або будь-якої конфіденційної інформації шляхом горизонтального переміщення та ескалації привілеїв. Фішинг може бути використаний як перший крок зловмисників до компрометації мережі та може призвести до складних кіберзлочинів, кібердиверсій або дій, пов’язаних із розширеною постійною загрозою (APT).

Електронні листи та кампанії, згенеровані штучним інтелектом, дозволяють охопити більшу кількість жертв за допомогою персоналізованих повідомлень, тепер використовуючи хмарні сервіси, які створюють враження легітимного повідомлення, щоб обійти системи фільтрації електронної пошти.

Інструменти захисту: антивірус для виявлення та видалення шкідливого програмного забезпечення, надійні паролі та багатофакторна автентифікація (MFA), рішення для фільтрації електронної пошти, навчання та підвищення обізнаності щодо виявлення підозрілих листів.

**Атаки на ланцюги поставок**

Зловмисники можуть атакувати не лише організацію, але й її постачальників. Вони можуть виявляти та використовувати вразливості в сторонніх сервісах для доступу до мережі організації.

Організація повинна не лише забезпечувати безпеку власної мережі, але й гарантувати, що її постачальники впроваджують відповідні заходи та інструменти виявлення та усунення недоліків.

Інструменти захисту: відстеження відповідності постачальників вимогам та моніторинг зовнішніх рішень, інструменти виявлення вторгнень на основі поведінкового аналізу, такі як EDR.

**Розподілені атаки типу «відмова в обслуговуванні» (DDoS)**

DDoS-атаки полягають у недоступності мережевих сервісів через перевантаження систем надмірним трафіком або запитами на з'єднання.

Вони можуть бути спрямовані на пропускну здатність, обчислювальну потужність сервера або ресурси прикладного рівня. DDoS-атаки зазвичай покладаються на ботнети для об'єднання пропускної здатності тисяч (або до мільйонів) окремо скомпрометованих комп'ютерів та перевантаження мережі цільових організацій.

Інструменти захисту: анти-DDoS (в ідеалі налаштований на межі мережі), балансувальники навантаження, інструменти моніторингу мережевого трафіку, такі як брандмауери, для виявлення будь-якої підозрілої кількості трафіку або запитів.

**Атаки типу «людина посередині»**

Людина посередині (MITM) – це дуже широкий тип атаки, який можна використовувати для прихованої крадіжки даних або облікових даних. Зловмисники можуть перехоплювати дані жертв, контролюючи та змінюючи їхні мережеві комунікації.

Розміщуючи себе «посередині», зловмисники можуть виступати в ролі цільового сервісу, заманюючи користувачів у відчуття безпеки. Їхні атаки можуть відбуватися через компрометовані маршрутизатори або публічні мережі Wi-Fi за допомогою сніферів пакетів, вони можуть перенаправляти трафік до власних систем або динамічно перехоплювати спроби автентифікації та захоплювати сеанси користувачів.

Інструменти захисту: VPN, шифрування даних, сучасні протоколи безпеки та шифрування.

**Внутрішні загрози**

Загрози також можуть виходити зсередини! Працівники або підрядники можуть використовувати свій доступ до інформаційної системи для порушення безпеки.

Конфіденційна інформація може бути розкрита або викрадена навмисно (крадіжка даних) чи помилково (натискання на шкідливе посилання або файл у фішинговому електронному листі). Оскільки ці загрози походять зсередини організації, їх особливо важко виявити за допомогою інструментів безпеки.

Метод нульової довіри – або, принаймні, «менталітет нульової довіри» – допоможе захистити мережу від таких інцидентів.

Інструменти захисту: підвищення обізнаності та навчання співробітників, сегментація та моніторинг мережі, посилені політики контролю доступу та інструмент керування привілейованим доступом (PAM), функція контролю зовнішніх пристроїв та виявлення кінцевих точок за поведінкою (EDR).

**Криптоджекінг**

Криптоджекінг передбачає використання комп'ютерної системи для майнінгу криптовалют, часто через вебсайти, проти волі користувача або без його відома.

Після того, як зловмисник проникне в мережу для криптоджекінгу, він може розширити свою атаку. Навіть якщо зловмисники «лише» розгорнуть програмне забезпечення для криптомайнінгу, ніщо не заважає їм змінювати напрямок та здійснювати інші дії в майбутньому!

З цієї причини важливо підготуватися до такого типу атаки.

Інструменти захисту: EDR, сегментація мережі для запобігання горизонтальному переміщенню, посилення захисту програмного забезпечення для запобігання встановленню невідомих або ненадійних програм.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

# 2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Безпека веб-застосунків, перш за все, спрямована на вирішення таких широких категорій ризиків:

**Витік даних та незахищений зв'язок**

Витік даних відбувається, коли конфіденційна інформація, така як персональні дані або платіжні реквізити, не захищена належним чином під час зберігання або передачі. Незахищені канали зв'язку дозволяють зловмисникам легко перехоплювати або змінювати інформацію, що особливо проблематично для веб-застосунків, які часто обмінюються даними з користувачами та іншими системами. Невикористання надійних протоколів шифрування, таких як TLS, або зберігання даних у відкритому тексті наражає організації на ризики витоків даних та недотримання вимог.

Зловмисники можуть використовувати недостатній захист даних, перехоплюючи трафік через незахищені з’єднання, отримуючи доступ до неналежним чином захищених баз даних або використовуючи вразливості в API. Щоб протидіяти цим ризикам, додатки повинні забезпечити передачу всіх даних, що обмінюються між браузером і сервером, через зашифровані канали, а також безпечне зберігання конфіденційної інформації за допомогою надійних криптографічних методів. Регулярна оцінка потоків даних і сховища необхідна для перевірки належного контролю безпеки.

**Ін'єкційні атаки**

Атаки типу ін'єкцій, такі як SQL-ін'єкції та ін'єкції команд, залишаються значною загрозою для веб-застосунків. Зловмисники використовують вразливості, надсилаючи ненадійні вхідні дані безпосередньо до інтерпретатора

застосунку або бази даних, що може призвести до несанкціонованого доступу до даних, втрати даних або навіть компрометації системи. Ці атаки можливі, коли застосунки не можуть належним чином перевірити та очистити дані, надані користувачем, або використовують небезпечні конструкції коду, такі як конкатенація рядків у запитах до бази даних.

Щоб запобігти атакам ін'єкцій, розробники повинні використовувати параметризовані запити, підготовлені оператори та інші безпечні методи кодування, які відокремлюють виконуваний код від вхідних даних, наданих користувачем. Перевірка вхідних даних та належна обробка помилок також є життєво важливими для зниження ризику. Регулярні перевірки коду та автоматизоване тестування безпеки повинні бути невід'ємною частиною робочого процесу розробки для виявлення та усунення вразливостей ін'єкцій перед розгортанням.

**Атаки шкідливих скриптів**

Міжсайтовий скриптинг (XSS) відбувається, коли зловмисник впроваджує шкідливі скрипти на веб-сторінку, яку переглядають інші користувачі, що дозволяє йому красти файли cookie сеансу, видавати себе за інших користувачів або перенаправляти трафік. Міжсайтова підробка запитів (CSRF) обманом змушує браузери користувачів виконувати небажані дії у веб-застосунку, в якому користувач автентифікований. Обидві атаки використовують слабкі місця в обробці вводу та управлінні сеансами.

Щоб пом'якшити атаки XSS, програми повинні очищувати та ігнорувати введені користувачем дані перед відображенням контенту на веб-сторінках, а також використовувати заголовки політики безпеки контенту (CSP) для обмеження виконання скриптів. Запобігання CSRF вимагає використання токенів перевірки (таких як токени CSRF) та гарантування того, що конфіденційні дії не можуть бути виконані без явної згоди. Підхід з глибоким захистом ускладнює для зловмисників використання цих вразливостей, захищаючи як користувачів, так і веб-програми.

**Неправильні конфігурації**

Неправильні конфігурації залишаються недооцінюваною, але критичною проблемою безпеки веб-додатків. До поширених помилок належать облікові дані за замовчуванням, непотрібні відкриті порти, докладні повідомлення про помилки або неправильні дозволи хмарного сховища. Ці прогалини надають зловмисникам точки входу для доступу до систем, підвищення привілеїв або витоку даних. Компоненти додатків, такі як веб-сервери, фреймворки та бази даних, потребують певних заходів захисту, щоб мінімізувати їхню поверхню атаки. Надійний процес керування конфігурацією, автоматизовані перевірки конфігурації та постійна валідація є важливими для запобігання неправильним конфігураціям. Регулярні перевірки налаштувань програм та інфраструктури в поєднанні з автоматизованими сценаріями розгортання та тестами безпеки допомагають підтримувати узгодженість та захищати від випадкових витоків. Оновлення документації та використання безпечних налаштувань за замовчуванням ще більше знижують ризик порушень, пов'язаних з неправильною конфігурацією.

**Ризики третіх сторін та ланцюгів поставок**

Сучасні веб-додатки покладаються на сторонні бібліотеки, API та платформи для пришвидшення розробки та додавання функцій. Однак незахищені або скомпрометовані залежності можуть призвести до серйозних вразливостей. Атаки на ланцюги поставок спрямовані на ці сторонні компоненти, компрометуючи додаток, навіть якщо власний код безпечний. Гучні інциденти демонструють, що одна непатчена бібліотека може поставити під загрозу цілі організації. Щоб зменшити ці ризики, організації повинні вести інвентаризацію всіх залежностей сторонніх розробників та контролювати їх на наявність вразливостей. Автоматизовані інструменти можуть допомогти виявити застарілі або високоризикові компоненти. Встановлення політик оновлення та перевірки залежностей, а також моніторинг програмних специфікацій матеріалів (SBOM), значно підвищує стійкість до атак ланцюга поставок та зменшує потенційні вектори атак.

**Найкращі практики безпеки веб-застосунків**

**Безпечна розробка та кодування**

**1. Перевірте та очистіть усі дані користувача**

Перевірка та очищення введених користувачем даних є важливими для запобігання атакам, що використовують ненадійні дані, такі як SQL-ін'єкції, XSS або віддалене виконання коду. Перевірка гарантує, що вхідні дані відповідають очікуваним форматам, типам і довжинам, відфільтровуючи небезпечні або неправильно сформовані значення перед обробкою. Очищення видаляє або кодує потенційно шкідливі символи, що робить безпечним відображення введених даних або їх зберігання в базах даних. Поєднання обох методів усуває значну кількість відомих вразливостей.

Покладатися лише на перевірку на стороні клієнта недостатньо, оскільки зловмисники можуть обійти перевірки на основі браузера. Перевірка та очищення вхідних даних завжди повинні відбуватися на стороні сервера, поблизу точки обробки даних. Розробникам слід використовувати білі списки для прийнятного вхідного даних, використовувати вбудовані функції фреймворків та уникати створення власної логіки перевірки, коли доступні перевірені бібліотеки. Регулярні перевірки безпеки та автоматизовані тести допомагають забезпечити безпечну обробку вхідних даних протягом усього життєвого циклу програми.

**2. Використовуйте параметризовані запити**

Параметризовані запити, також відомі як підготовлені оператори, відокремлюють SQL-код від даних, введених користувачем. Така конструкція запобігає впровадженню зловмисниками шкідливих SQL-команд, що може призвести до витоків даних або маніпуляцій. Параметризовані запити вказують механізму баз даних обробляти введені користувачем дані як дані, а не виконуваний код, що зводить нанівець спроби вставки додаткових команд через поля форми або URL-адреси. Майже всі сучасні мови програмування та бібліотеки баз даних підтримують параметризовані запити. Впровадження параметризованих запитів – це простий, але потужний захист від атак ін'єкцій.

Необхідно послідовне використання в усіх точках доступу до даних, оскільки навіть один вразливий запит створює вектор атаки. Розробникам також слід уникати конкатенації рядків або прямої інтерполяції вводу в команди SQL за будь-яких обставин. Регулярні перевірки коду та автоматизовані тести, орієнтовані на безпеку, забезпечують використання параметризованих запитів і допомагають усунути небезпечні практики обробки даних.

**3. Впроваджуйте безпечну автентифікацію та авторизацію**

Автентифікація підтверджує особу користувача, тоді як авторизація визначає доступ до ресурсів у веб-застосунку. Впровадження безпечної автентифікації передбачає застосування надійних політик паролів, підтримку багатофакторної автентифікації та зберігання облікових даних за допомогою криптографічно захищених хеш-функцій (таких як bcrypt або Argon2). Популярні варіанти, такі як OAuth, OpenID Connect та SAML, забезпечують безпечні потоки автентифікації як для користувачів, так і для служб.

Авторизація повинна відповідати принципу найменших привілеїв, надаючи користувачам лише ті дозволи, які їм потрібні для виконання завдань. Контроль доступу на основі ролей (RBAC) та контроль доступу на основі атрибутів (ABAC) допомагають керувати дозволами у великих масштабах. Розподіл обов'язків в архітектурі програми гарантує, що користувачі не можуть підвищувати привілеї або отримувати доступ до даних поза межами їхньої передбачуваної сфери. Регулярні перегляди та оновлення логіки авторизації є важливими, оскільки функції програми та ролі користувачів розвиваються.

**4. Безпечно обробляйте помилки**

Неправильна обробка помилок може ненавмисно розкрити конфіденційну інформацію про внутрішню роботу веб-програми, шляхи до файлів або версії програмного забезпечення – підказки, корисні для зловмисників. Безпечна обробка помилок передбачає надання загальних повідомлень кінцевим користувачам під час реєстрації детальної технічної інформації для адміністраторів. Відповіді на помилки ніколи не повинні розкривати трасування стека, SQL-запити або дані конфігурації системи у виробничому середовищі.

Для забезпечення безпечного керування помилками, програми повинні централізовано обробляти винятки, безпечно реєструвати помилки та вимикати детальні режими налагодження поза межами розробки. Журнал повинен фіксувати достатньо деталей для судово-медичного аналізу, не розкриваючи конфіденційні дані користувачів. Розробники повинні регулярно тестувати відповіді на помилки, щоб переконатися, що жодна внутрішня інформація не розголошується ненавмисно. Безпечна обробка помилок допомагає захистити програми від розвідки та цілеспрямованого використання.

**5. Уникайте жорстко закодованих облікових даних**

Жорстке кодування облікових даних, таких як паролі бази даних або ключі API, у вихідному коді або файлах конфігурації створює значний ризик для безпеки. Зловмисники можуть витягти ці секрети з репозиторіїв контролю версій, програмних пакетів або бінарних файлів програм, що призводить до несанкціонованого доступу та потенційних витоків даних. Натомість, облікові дані слід зберігати в зашифрованих змінних середовища або керувати ними за допомогою захищених інструментів сховища, розроблених спеціально для керування секретами.

Регулярні аудити повинні перевіряти вихідний код та розгортання на наявність випадкового розкриття облікових даних. Безперервна інтеграція та конвеєри розгортання можуть забезпечити дотримання політик, які блокують зміни, що містять секрети. Розробникам також слід регулярно змінювати облікові дані та дотримуватися принципу найменших привілеїв, щоб мінімізувати вплив скомпрометованого облікового запису. Уникнення жорстко закодованих облікових даних знижує ризик отримання доступу до критично важливих систем як внутрішніми, так і зовнішніми загрозами.

**Інфраструктура та розгортання**

**6. Використовуйте HTTPS із шифруванням TLS**

Уся комунікація між веб-програмами, користувачами та API повинна використовувати HTTPS, захищений протоколом Transport Layer Security (TLS), щоб запобігти перехопленню та несанкціонованому втручанню. HTTPS гарантує,

що конфіденційні дані, такі як облікові дані для входу, платіжна інформація та токени сеансу, не можуть бути прослухані або змінені під час передачі. Навіть внутрішні адміністративні інтерфейси та кінцеві точки API вимагають HTTPS для блокування локальних або внутрішніх загроз.

Впровадження HTTPS передбачає отримання та налаштування цифрових сертифікатів, забезпечення перенаправлення всього HTTP-трафіку на HTTPS та вимкнення незахищених протоколів або наборів шифрів. Такі інструменти, як HTTP Strict Transport Security (HSTS), забезпечують додатковий захист, забезпечуючи використання лише HTTPS-з'єднань. Регулярне сканування вразливостей має оцінювати дійсність сертифіката та надійність криптографічних налаштувань. Використання HTTPS з TLS є фундаментальною вимогою для будь-якої веб-програми, яка обробляє конфіденційні дані.

**7. Оновлюйте та виправляйте програмне забезпечення**

Платформи веб-застосунків, фреймворки, бази даних та серверні операційні системи періодично отримують оновлення безпеки та виправлення від постачальників. Застаріле програмне забезпечення є основною причиною вразливостей, оскільки зловмисники активно шукають та використовують відомі недоліки за допомогою доступних експлойтів, що підтверджують їхню придатність. Своєчасні оновлення мінімізують ризики, блокуючи добре задокументовані вектори атак.

Організації повинні підтримувати чіткий процес управління виправленнями, який інвентаризує всі компоненти програмного забезпечення та контролює відповідні потоки рекомендацій щодо безпеки. Автоматизована перевірка оновлень та поетапне розгортання допомагають мінімізувати час простою та збої в роботі. Найважливіше те, що залежності розробки та образи контейнерів так само важливі для оновлення, як і виробнича інфраструктура. Підтримка програмного забезпечення в актуальному стані є базовим, але життєво важливим захистом від компрометації.

**8. Безпечні API**

API формують основу сучасних веб-застосунків, часто розкриваючи

значну бізнес-логіку або конфіденційні дані. Захист API вимагає надійної автентифікації (наприклад, OAuth 2.0 або ключів API), перевірки вхідних даних та шифрування трафіку. Обмеження швидкості, квоти та ведення журналу допомагають запобігти зловживанням, атакам методом грубої сили та інцидентам відмови в обслуговуванні. Крім того, шлюзи API можуть забезпечувати узгоджені політики безпеки в усіх мікросервісах.

Розробникам слід перевіряти всі API на наявність непублічних кінцевих точок, витоків інформації про помилки або надмірних областей дозволів. Використовуйте інструменти специфікації та тестування, такі як OpenAPI, Swagger та Postman, для документування та перевірки поведінки кінцевих точок. Автоматизоване сканування та моніторинг трафіку API може виявити аномальні моделі використання або нові загрози. Захист API знижує ризик горизонтального переміщення та ескалації привілеїв у розподілених архітектурах.

**9. Забезпечити дотримання принципу найменших привілеїв (PoLP)**

Принцип найменших привілеїв диктує, що користувачі, процеси та компоненти повинні мати лише ті дозволи, які необхідні для виконання їхніх завдань. Зменшення обсягу дозволів обмежує радіус вибуху у разі компрометації, що значно ускладнює горизонтальне переміщення або ескалацію привілеїв для зловмисників. Застосування найменших привілеїв є критично важливим для облікових записів сервісів, середовищ виконання контейнерів та розгорнутих мікросервісів.

Політики безпеки та засоби контролю доступу необхідно регулярно переглядати та оновлювати, щоб запобігти поширенню привілеїв у міру зміни ролей або додавання функцій. Автоматизовані інструменти можуть перевіряти дозволи аж до рівня ресурсів і позначати надмірні права. Забезпечення мінімальних привілеїв не лише покращує безпеку, але й підтримує ініціативи щодо дотримання вимог, обмежуючи непотрібний доступ до регульованих даних.

**10. Шифрування даних у стані спокою**

Шифрування даних у стані спокою гарантує, що навіть у разі фізичного

витоку пристроїв зберігання даних, резервних копій або баз даних, конфіденційна інформація залишається захищеною та нерозшифровуваною для неавторизованих користувачів. Залежно від нормативних вимог та архітектури системи можна використовувати рішення для шифрування диска, шифрування на рівні файлів або криптографії на рівні програм. Ключі шифрування слід зберігати та керувати ними безпечно, окремо від серверів програм або систем зберігання даних. Впровадження надійного шифрування для баз даних, постійних сховищ та хмарних сховищ об'єктів є важливим для захисту від витоків даних. Регулярно перевіряйте конфігурації шифрування та методи керування ключами, щоб реагувати на нові загрози та дотримуватися стандартів, що розвиваються. Шифрування даних у стані спокою є невід'ємною вимогою для будь-якої веб-програми, яка керує особистими, фінансовими або конфіденційними даними.

**Безперервна безпека та моніторинг**

**11. Інтеграція моніторингу та реагування на основі штучного інтелекту для вебзастосунків**

Системи моніторингу на основі штучного інтелекту використовують машинне навчання та поведінковий аналіз для виявлення аномалій, загроз та незвичайних закономірностей в активності веб-додатків. Ці рішення адаптуються до методів атак, що розвиваються, що дозволяє швидко виявляти вразливості нульового дня, викрадення облікових записів та складні сценарії зловживань, які традиційні системи на основі правил можуть пропустити. Інтеграція моніторингу на основі штучного інтелекту дозволяє організаціям швидко реагувати на інциденти та скорочувати час перебування зловмисників. Розгортання передбачає підключення інструментів на базі штучного інтелекту до потоків журналів, подій програм і датчиків мережевого трафіку, де вони постійно навчаються та позначають підозрілу поведінку. Автоматизовані механізми реагування можуть ізолювати загрози, запускати сповіщення або усувати певні класи атак без втручання людини. Регулярне налаштування алгоритмів та інтеграція з процесами реагування на інциденти максимізують цінність безпеки веб-застосунків на базі штучного інтелекту.

**12. Регулярно проводите тестування безпеки**

Регулярне тестування безпеки, включаючи тестування на проникнення, сканування вразливостей та аналіз коду, є життєво важливим для виявлення та виправлення слабких місць, перш ніж зловмисники їх скористаються. Автоматизовані інструменти можуть постійно сканувати на наявність відомих проблем, тоді як ручні оцінки, що проводяться етичними хакерами, виявляють логічні недоліки та вразливості бізнес-логіки поза досяжністю автоматизованих сканерів. Як статичне, так і динамічне тестування відіграють певну роль протягом усього життєвого циклу розробки.

Впровадження стратегії тестування безпеки включає планове сканування всіх середовищ, перевірку стороннього коду та регулярні тренування червоної команди. Результати слід документувати, визначати пріоритети та усувати відповідно до ризику. Включення тестування до конвеєра CI/CD дозволяє командам виявляти вразливості раніше, зменшуючи витрати на усунення та покращуючи загальний стан безпеки додатків.

**13. Впроваджуйте брандмауер веб-застосунків (WAF)**

Брандмауер веб-застосунку (WAF) перехоплює та фільтрує трафік між користувачами та веб-застосунком, блокуючи шкідливі запити, такі як SQL-ін'єкції, XSS та автоматизовані атаки ботів. WAF працюють, використовуючи сигнатури для відомих загроз, виявлення аномалій та налаштовувані політики, що адаптуються до вимог конкретного застосунку. Вони є практичним рівнем захисту, який може значно знизити рівень успішного використання шкідливих програм.

Розгортання WAF включає налаштування наборів правил для мінімізації хибнопозитивних та хибнонегативних результатів, інтеграцію його з безперервним моніторингом безпеки та оновлення його з урахуванням найновіших даних про загрози. Хоча WAF не замінює безпечне кодування, він забезпечує важливий захист для застарілих програм або вразливостей нульового дня. Реєстрація та оповіщення про тригери правил WAF допомагають у виявленні інцидентів та реагуванні на них.

**14. Моніторинг та реєстрація активності**

Моніторинг та ведення журналу є основою будь-якої програми виявлення та реагування на інциденти. Комплексне ведення журналу фіксує спроби автентифікації, зміни конфігурації, адміністративні дії та помилкові стани. Централізовані платформи ведення журналу дозволяють проводити аналіз у режимі реального часу, довгострокове зберігання та швидку кореляцію під час судово-медичних розслідувань. Журнали повинні бути захищеними від несанкціонованого доступу та доступними лише для уповноваженого персоналу.

Ефективний моніторинг передбачає налаштування автоматичних сповіщень про незвичайну поведінку, надмірну кількість невдалих входів або підозрілі моделі доступу. Регулярно перевіряйте журнали на наявність порушень політик або ранніх ознак компрометації. Поєднання моніторингу з іншими методами безпеки дозволяє організаціям швидко виявляти порушення та реагувати на них, зменшуючи вплив атаки та час відновлення.

**15. Безпечне керування залежностями**

Безпечне керування залежностями є критично важливим, оскільки більшість веб-проектів залежать від великих бібліотек та зовнішніх модулів. Застарілі або вразливі залежності можуть призвести до серйозних експлойтів, навіть якщо код програми безпечний. Автоматизовані інструменти, такі як аналіз складу програмного забезпечення (SCA), відстежують, оцінюють та сповіщають про відомі вразливості в залежностях проєкту.

Розробники повинні підтримувати актуальний перелік усіх сторонніх компонентів та постійно контролювати їх на наявність проблем. Застосовуйте політики, що обмежують несанкціоновані або застарілі бібліотеки, та впроваджуйте процеси перевірки та оновлення залежностей як частину регулярних циклів розробки. Сканування контейнерів, інтеграція перевірок залежностей у конвеєри CI/CD та перевірка походження компонентів з відкритим кодом допомагають захистити ширший ланцюжок постачання веб-додатків.

**Безпека веб-застосунків за допомогою Radware**

Оскільки сучасні програми стикаються з дедалі складнішими атаками,

Radware пропонує єдиний набір можливостей, які покращують видимість, забезпечують безпечну поведінку програм і запобігають використанню на кожному рівні.

**Хмарний сервіс WAF**

Хмарний сервіс WAF є основним елементом портфоліо безпеки веб-додатків Radware. Він забезпечує розширений захист від 10 найкращих ризиків OWASP, включаючи атаки ін'єкцій, порушення контролю доступу та міжсайтовий скриптинг. Використовуючи адаптивне навчання, сервіс автоматично профілює легітимний трафік та створює позитивні моделі безпеки, адаптовані до кожної програми. Він також забезпечує віртуальне встановлення патчів, захищаючи вразливі програми від експлуатації, поки команди розробників розгортають постійні виправлення. Хмарний WAF перевіряє заголовки, корисні навантаження, сесії та сигнали ідентифікації користувачів, щоб блокувати шкідливу активність, не порушуючи роботу легітимних користувачів.

**Служба захисту хмарних застосунків (WAF + API + керування ботами)**

Служба захисту хмарних застосунків (Cloud Application Protection Service) забезпечує єдиний захист для сучасних веб-застосунків та API. Вона поєднує можливості WAF із безпекою API та керуванням ботами, що дозволяє організаціям захищати веб-, мобільні та мікросервісні архітектури в рамках єдиної платформи. Служба включає перевірку схеми, адаптивне обмеження швидкості, захист від додавання облікових даних та поведінковий аналіз для виявлення автоматизованих загроз та аномального споживання API. Її консолідована видимість допомагає організаціям застосовувати узгоджені політики в розподілених та хмарних середовищах.

**Менеджер ботів**

Bot Manager розрізняє хороших ботів (пошукові системи, партнерські інтеграції) та шкідливу автоматизацію, що використовується для парсингу, спроб грубої сили, захоплення облікового запису або розвідки. Він використовує

відбитки пристроїв, поведінкові випробування, класифікацію на основі намірів та машинне навчання для виявлення складних ботів, які імітують взаємодію людей. Фільтруючи шкідливу автоматизацію на периферії, Bot Manager захищає програми від зниження продуктивності, зловживання обліковими даними та атак бізнес-логіки, які в іншому випадку обійшли б традиційні засоби контролю безпеки.

**Захист від DDoS-атак у веб-просторі**

Захист від веб-DDoS-атак захищає програми від масштабних HTTP-флудів та L7 DDoS-атак, які перевантажують сервери запитами, що виглядають легітимними. На відміну від статичних засобів контролю швидкості, це рішення застосовує поведінковий аналіз без сигнатур для виявлення аномальних закономірностей в ентропії URL-адрес, темпі виконання запитів та взаємодії з користувачами. Воно генерує адаптивні політики пом'якшення наслідків у режимі реального часу, які блокують трафік атак, зберігаючи при цьому нормальну поведінку користувачів. Це стає все більш важливим для захисту сучасних програм від веб-флудів великого обсягу, які обходять звичайні засоби захисту інфраструктури від DDoS-атак.

**DefensePro (Фонд безпеки застосунків мережевого рівня)**

DefensePro забезпечує поведінковий захист від DDoS-атак, запобігання вторгненням та виявлення аномалій протоколу на межі мережі. Хоча це переважно рішення для DDoS-атак L3/L4, воно відіграє вирішальну роль у захисті середовищ веб-застосунків, блокуючи об'ємні атаки вище за течією, неправильне використання протоколів та розвідку, які часто передують вторгненням на рівні додатків. Його виявлення в режимі реального часу без сигнатур допомагає забезпечити доступність та ефективність захисту на рівні додатків (WAF, запобігання ботам, безпека API) під час складних багатовекторних кампаній.

**Хмарний сервіс захисту від DDoS-атак**

Служба захисту від DDoS-атак у хмарі поглинає масштабні об'ємні та багатовекторні атаки, перш ніж вони досягнуть інфраструктури джерела

додатків. Вона забезпечує автоматичне перенаправлення, високопродуктивне очищення та поведінкове виявлення для протидії динамічним кампаніям, спрямованим як на мережевий, так і на прикладний рівні. Стабілізуючи пропускну здатність, сеанси та мережу вище за течією під час атак, служба гарантує, що веб-додатки залишаються доступними, тоді як WAF та засоби контролю безпеки додатків працюють належним чином.

**Підписки на розвідку загроз**

Підписки на розвідку загроз надають кураторську інформацію про репутацію IP-адрес, індикатори ботнетів та профілювання акторів-злочинців, отримані від експертів Radware з глобальної телеметрії атак та ERT. Ця інформація покращує захист від WAF, управління ботами та DDoS-атак, превентивно блокуючи шкідливі джерела та виявляючи закономірності, пов'язані з відомими наборами експлойтів, ботнетами та інструментами розвідки. Це допомагає організаціям посилити контроль на рівні додатків за допомогою контекстно-залежного застосування.

**Аналітика хмарної мережі (керування видимістю та експозицією)**

Хмарна мережева аналітика забезпечує глибоке розуміння трафіку додатків, поведінки користувачів та потенційних неправильних конфігурацій, які розкривають веб-ресурси. Вона співвідносить дані потоків у гібридних та хмарних середовищах, виявляючи аномальні шаблони доступу, несанкціоновані кінцеві точки або активність API з високим рівнем ризику. Ця телеметрія допомагає командам безпеки перевіряти політики, виявляти загрози на ранніх стадіях та підтримувати узгоджений рівень безпеки додатків у розподілених середовищах.

**HarfangLab**

Кожен інструмент та дія кібербезпеки являють собою рівень багатогранної стратегії захисту від кіберзагроз. Окрім інструментів, правила IT-гігієни є важливими для ефективного кіберзахисту, зокрема завдяки суворим політикам безпеки, відповідним конфігураціям програмного забезпечення та посиленню захисту.

Майте на увазі, що жоден інструмент не здатний виявити всі можливі види атак : брандмауери не можуть виявити вразливості нульового дня або спроби фішингу; антивіруси базуються на виявленні сигнатур і не можуть виявити невідоме шкідливе програмне забезпечення або підозрілу поведінку; а дані, що зберігаються в хмарі, не захищені від порушень.

Ось чому важливо точно знати, який інструмент захищає від якої загрози, щоб створити ефективний захист кібербезпеки та отримати повну прозорість над інформаційною системою.

Платформа HarfangLab включає рішення для забезпечення повного захисту вашої інформаційної системи, від управління поверхнею атак до усунення загроз, включаючи вразливості та управління тіньовими IT, EPP та EDR.

І цей платформно-орієнтований підхід – чудовий спосіб вирішення проблем, з якими стикаються всі зацікавлені сторони, включаючи директорів з інформаційної безпеки, генеральних директорів та директорів з інформаційних технологій, щоб забезпечити безперервність обслуговування, уникнути фінансових втрат і зберегти довіру клієнтів і користувачів!

## 2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для розробки високопродуктивних компонентів системи захисту веб-ресурсів від пасивного аналізу трафіку та реалізації механізмів автоматизованого опортуністичного шифрування було обрано мову програмування **C++**.

C++ – це компільована, статично типізована мова загального призначення, яка поєднує властивості як високорівневих (об’єктно-орієнтоване та узагальнене програмування), так і низькорівневих мов. Наявність потужної стандартної бібліотеки, що включає ефективні контейнери, алгоритми, засоби введення-виведення та вбудовану підтримку багатопоточності, робить C++ оптимальним вибором для створення швидких мережевих застосунків, проксі-

серверів, модулів перехоплення/фільтрації трафіку та криптографічних шлюзів.

Завдяки своїй продуктивності та відсутності зайвих накладних витрат під час виконання («не платиш за те, що не використовуєш»), C++ широко застосовується в областях, критичних до швидкодії та затримок: при розробці операційних систем, мережевих драйверів, високопродуктивних серверів та систем безпеки. Сумісність синтаксису з мовою C дозволяє напряму взаємодіяти з низькорівневими мережевими інтерфейсами (наприклад, сокетами, стеками TCP/IP, бібліотеками OpenSSL тощо), що є критично важливим для аналізу мережевих пакетів і динамічного керування шифруванням трафіку «на льоту».

Підтримка в C++ парадигм об’єктно-орієнтованого програмування (інкапсуляція, спадкування, поліморфізм) та узагальненого програмування (шаблони, STL) дозволяє побудувати модульну, легко розширювану архітектуру захисної системи. Розробник отримує зручний інструментарій для створення гнучких структур даних, оптимізованих черг пакетів, а також безпечного керування пам'яттю, що мінімізує затримки (latency) при обробці великих обсягів вхідних запитів і захисті їх від атак пасивного прослуховування мережі (sniffing).

Починаючи з сучасних стандартів мови (C++11 і вище), розширені можливості багатопоточності, атомарних операцій та розумних покажчиків (smart pointers) дозволяють надійно реалізувати паралельну обробку потоків даних, асинхронне шифрування та ефективне використання ресурсів багатоядерних серверних платформ.

### 2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

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

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

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

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

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

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

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

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

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

# 3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

## 3.1 Опис функціонування системи

Організаціям потрібна мережева безпека для захисту критично важливих даних, програм та операцій від загроз мережевій безпеці, що постійно змінюються. Оскільки мережі розширюються через хмару, віддалених користувачів та третіх сторін, рішення для мережевої безпеки допомагають контролювати доступ, забезпечувати моніторинг безпеки мережі, зменшувати горизонтальне переміщення через сегментацію мережі та підтримувати ефективне управління мережевою безпекою, щоб мінімізувати час простою, втрату даних та перебої в роботі.

Сучасні мережі містять усе, від чого залежить організація. Бізнес-дані, інформація про клієнтів, програми та внутрішні комунікації переміщуються мережею. Якщо ця мережа скомпрометована, зловмисники не просто крадуть дані. Вони порушують операції, рухаються нестандартно та часто залишаються прихованими протягом тривалого часу.

Мережева безпека існує для того, щоб запобігти цьому. Це багаторівневий підхід, який контролює доступ, відстежує активність, захищає дані та обмежує вплив атак, коли щось прослизає.

Мережева безпека – це поєднання технологій, політик та процесів, що використовуються для захисту мережі від несанкціонованого доступу, неправильного використання, перебоїв та втрати даних.

На практичному рівні мережева безпека зосереджена на трьох основних цілях:

– Захистіть доступ, щоб підключатися могли лише довірені користувачі та пристрої.

– Слідкуйте за тим, що відбувається в мережі.

– Обмежте шкоду, швидко стримуючи загрози.

Ось чому ефективні рішення мережевої безпеки ніколи не будуються навколо одного інструменту. Вони покладаються на кілька елементів керування, що працюють разом.

Кожен елемент керування мережевою безпекою стосується різної частини ландшафту загроз. У наступній таблиці основні типи розподілено за їхньою основною функцією та роллю безпеки, яку вони відіграють, перш ніж ми детально розглянемо кожен елемент керування.

За останнє десятиліття загрози мережевій безпеці значно змінилися. Зловмисники більше не покладаються на гучні експлойти чи очевидне шкідливе програмне забезпечення. Вони використовують викрадені облікові дані, легітимні інструменти та зашифрований трафік, щоб непомітно злитися з натовпом.

Це робить мережеву безпеку критично важливою з кількох причин:

– Мережі тепер розподілені між хмарою, віддаленими користувачами та третіми сторонами.

– Одна скомпрометована кінцева точка може розкрити все середовище.

– Простої та втрата даних мають прямий фінансовий та репутаційний вплив.

Надійне управління мережевою безпекою знижує ризики, підвищує стійкість і надає командам безпеки необхідну прозорість для швидкого реагування.

**Різні типи мережевої безпеки**

Кожен тип мережевої безпеки відіграє певну роль. Марковані списки допомагають виділити мету, але пояснення показує, як кожен елемент керування фактично захищає мережу.

**1. Контроль доступу до мережі (NAC)**

Контроль доступу до мережі визначає, хто і що може підключатися до мережі.

– **Перевірка пристроїв та користувачів.** NAC перевіряє ідентифікаційні дані пристрою, облікові дані користувача та рівень безпеки перед наданням

доступу. Це допомагає блокувати невідомі або несумісні пристрої.

– **Зменшення поверхні атаки.** Обмежуючи доступ у точці входу, NAC запобігає перетворенню скомпрометованих кінцевих точок на легкий шлях до мережі.

NAC особливо важливий у середовищах з віддаленою роботою, підрядниками та некерованими пристроями.

**2. Політики IT-безпеки**

Політики IT-безпеки визначають, як застосовується та забезпечується безпека мережі.

– **Чіткі правила доступу та використання.** Політики визначають, хто може отримати доступ до чого, за яких умов та звідки.

– **Узгодженість під час інцидентів.** Коли трапляється інцидент, політики керують діями реагування та запобігають плутанині або затримкам.

Без чітких правил навіть передові рішення мережевої безпеки застосовуються непослідовно.

**3. Безпека застосунків**

Безпека програм захищає програмне забезпечення, яке працює в мережі.

– **Захист від атак на рівні додатків.** Це включає захист веб-додатків та API, на які зловмисники часто націлені в першу чергу.

– **Зменшення внутрішнього впливу.** Вразливий додаток може надати зловмисникам легітимний доступ до внутрішніх систем. Безпека додатків допомагає усунути цю прогалину.

Оскільки все більше бізнес-логіки переміщується в додатки, цей рівень стає дедалі важливішим.

**4. Управління вразливостями**

Управління вразливостями зосереджене на виявленні та усуненні слабких місць, перш ніж зловмисники їх скористаються.

– **Видимість уразливих систем**. Регулярне сканування допомагає командам зрозуміти, які системи вразливі та чому.

– **Пріоритизація на основі ризиків.** Не всі вразливості однакові.

Управління вразливостями допомагає командам зосередитися на тому, що найважливіше.

Це перетворює мережеву безпеку з реактивної на проактивну.

**5. Тестування на проникнення в мережу**

Тестування на проникнення оцінює, наскільки добре засоби контролю безпеки мережі витримують атаки.

– **Моделювання атаки в реальному світі.** Тести пера намагаються обійти елементи керування так, як це зробив би справжній зловмисник.

– **Перевірка існуючих захисних механізмів.** Результати показують, чи дійсно брандмауери, сегментація та моніторинг працюють разом.

Тестування на проникнення виявляє прогалини, які автоматизовані інструменти часто пропускають.

**6. Запобігання втраті даних (DLP)**

Запобігання втраті даних захищає конфіденційні дані під час їх переміщення мережею.

– **Інструменти DLP для моніторингу переміщення даних** відстежують файли, електронні листи та завантаження, щоб запобігти несанкціонованому обміну.

– **Зменшення ризиків внутрішнього проникнення та випадкового проникнення.** Багато витоків даних трапляються помилково. DLP допомагає запобігти цим інцидентам.

DLP має вирішальне значення для організацій, що працюють з регульованою або конфіденційною інформацією.

**7. Антивірусне програмне забезпечення**

Антивірусне програмне забезпечення забезпечує базовий захист від відомих загроз.

– **Виявлення поширених шкідливих програм.** Виявлення на основі сигнатур зупиняє великий обсяг малозабезпечених атак.

– **Зменшення фонового шуму.** Блокування поширених загроз дозволяє командам зосередитися на більш складних загрозах мережевій безпеці.

Антивіруса самого по собі недостатньо, але він залишається необхідним шаром.

**8. Виявлення та реагування на кінцеві точки (EDR)**

EDR розширює безпеку кінцевих точок за межі базового виявлення шкідливого програмного забезпечення.

– **Моніторинг на основі поведінки.** EDR шукає підозрілі дії, а не відомі сигнатури.

– **Можливості розслідування та реагування.** Команди безпеки можуть аналізувати інциденти та вживати заходів безпосередньо з кінцевої точки.

EDR є критично важливим, оскільки кінцеві точки часто є першою точкою компрометації.

**9. Безпека електронної пошти**

Безпека електронної пошти захищає користувачів від одного з найпоширеніших векторів атак.

– **Захист від фішингу та шкідливих програм.** Інструменти захисту електронної пошти аналізують повідомлення на наявність шкідливих намірів.

– **Зменшення початкового ризику.** Менше успішних фішингових атак означає менше зловмисників у мережі.

Надійний захист електронної пошти знижує загальний ризик для безпеки мережі.

**10. Безпека бездротового зв'язку**

Безпека бездротового зв'язку захищає мережі, що працюють за межами фізичних кордонів.

– **Безпечна автентифікація та шифрування.** Ці елементи керування запобігають несанкціонованому доступу до мереж Wi-Fi.

– **Виявлення несанкціонованих точок доступу.** Моніторинг допомагає виявляти неавторизовані бездротові пристрої.

Без бездротового захисту зловмисники можуть повністю обійти фізичний контроль.

**11. Системи виявлення та запобігання вторгненням (IDS/IPS)**

IDS та IPS відстежують мережевий трафік на наявність шкідливої активності.

– **Виявлення підозрілих закономірностей.** Система виявлення вторгнень сповіщає команди про потенційні атаки або зловживання.

– **Активне блокування загроз.** IPS може зупинити шкідливий трафік, перш ніж він завдасть шкоди.

Ці системи додають глибини моніторингу безпеки мережі.

**12. Сегментація мережі**

Сегментація мережі поділяє мережу на контрольовані зони.

– **Обмежений бічний рух.** Сегментація перешкоджає нападникам вільно пересуватися всередині.

– **Зменшений радіус вибуху.** Навіть якщо один сегмент пошкоджений, інші залишаються захищеними.

Сегментація є одним із найефективніших способів зменшення впливу порушень.

**13. Управління інформацією та подіями безпеки (SIEM)**

SIEM централізує дані безпеки по всій мережі.

– **Кореляція журналів та подій.** SIEM поєднує дані з кількох джерел для виявлення закономірностей.

– **Швидше виявлення та розслідування.** Централізована видимість покращує час реагування та точність.

SIEM є наріжним каменем сучасного управління мережевою безпекою.

**14. Веб-безпека**

Веб-безпека захищає користувачів від веб-загроз.

– **Блокування шкідливих веб-сайтів** за допомогою фільтрації URL-адрес зменшує ризики, пов'язані з відомими загрозами.

– **Захист від небезпечних завантажень.** Веб-інспекція допомагає зупинити поширення шкідливого програмного забезпечення.

Веб-безпека доповнює захист електронної пошти та кінцевих точок.

**15. Багатофакторна автентифікація (MFA)**

Багатостороння автентифікація (MFA) посилює безпеку на основі ідентифікації.

– **Захист від крадіжки облікових даних.** Навіть якщо паролі скомпрометовано, багатофакторна автентифікація блокує доступ.

– **Зниження ризику захоплення облікового запису.** Багатофакторна автентифікація (MFA) вирішує одну з найпоширеніших загроз безпеці мережі.

Це один з найпростіших та найефективніших доступних засобів контролю.

**16. Віртуальна приватна мережа (VPN)**

VPN забезпечують безпечний віддалений доступ до мережі.

– **VPN із зашифрованим зв'язком** захищають дані під час їх передачі через загальнодоступні мережі.

– **Контрольоване віддалене підключення.** Доступ можна обмежити та контролювати.

VPN-мережі залишаються широко використовуваними, особливо постачальниками мережевої безпеки для малого бізнесу.

## 3.2 Розробка структурної схеми

Окремо кожен тип мережевої безпеки вирішує вузьку проблему. Разом вони визначають, чи може організація фактично зупинити атаку, чи просто задокументувати її після того, як шкода вже завдана.

Ключова ідея – багаторівнева оборона зі спільним контекстом. Атакувальні сили не рухаються по прямих лініях, тому оборонні споруди не можуть існувати ізольовано.

Ось як ці елементи керування працюють разом протягом типового життєвого циклу атаки.

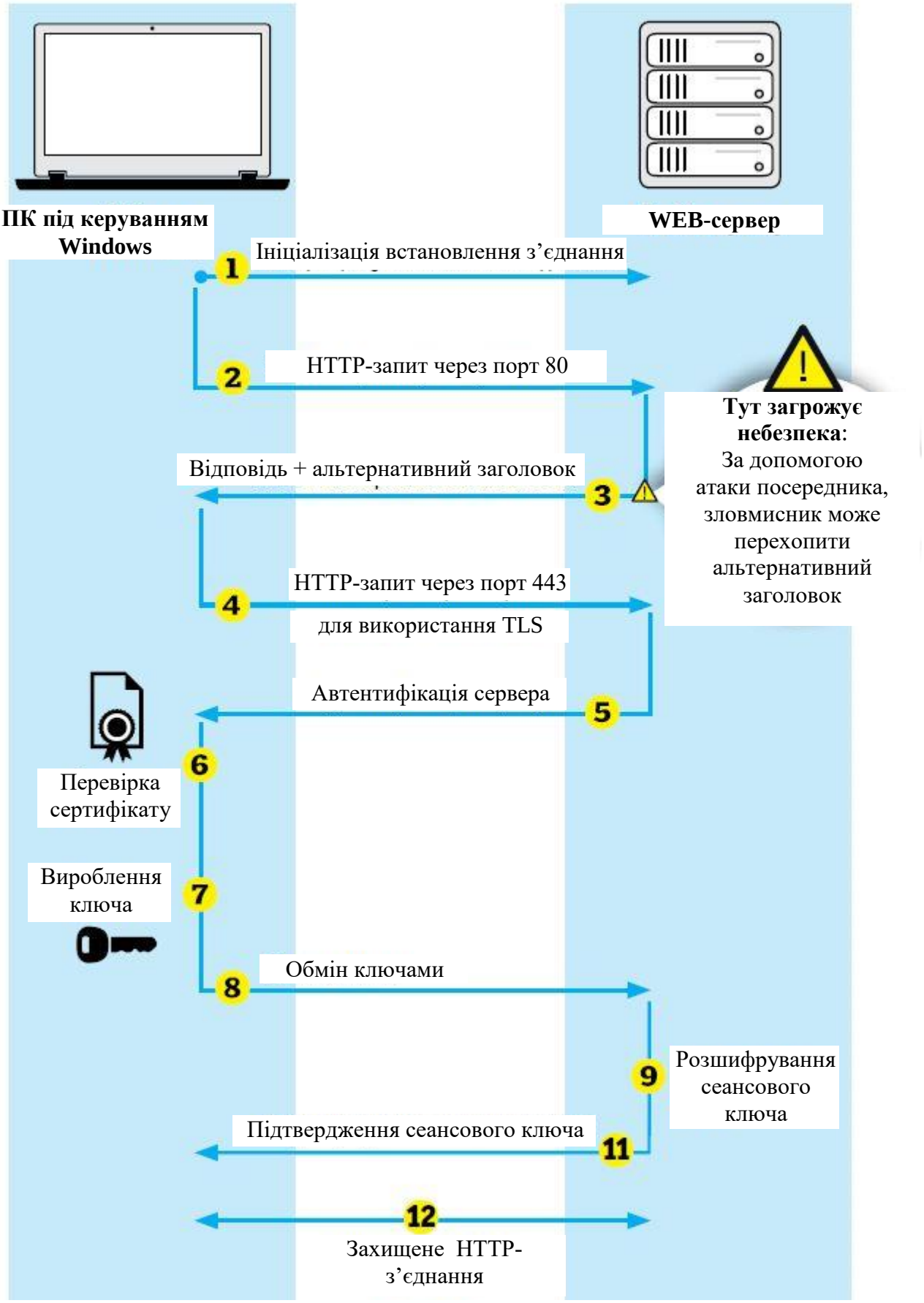


Рисунок 3.1 – Структурна схема системи

**1. Запобігання несанкціонованому доступу в точці входу**

Перший рівень зосереджений на зупинці зловмисників до того, як вони проникнуть всередину:

– Контроль доступу до мережі, багатофакторна автентифікація та політики IT-безпеки працюють разом, щоб контролювати, хто може підключатися та за яких умов.

– Безпека електронної пошти та веб-безпека зменшують ймовірність проникнення через фішинг, який досі є найпоширенішим методом початкового доступу.

– VPN та безпечні засоби керування віддаленим доступом захищають користувачів, які підключаються до мережі через ненадійні мережі.

Якщо ці рівні виконують свою роботу, багато атак ніколи не просуваються далі першого кроку. Але однієї лише профілактики ніколи не достатньо.

**2. Зменшення впливу, коли профілактика не допомагає**

Припустимо, що облікові дані викрадено або пристрій скомпрометовано. Саме тут важливе значення має стримування:

– Сегментація мережі обмежує подальші дії зловмисника, запобігаючи необмеженому горизонтальному переміщенню.

– Безпека застосунків гарантує, що навіть автентифікований доступ не призведе автоматично до контролю на системному рівні.

– Безпека бездротового зв'язку запобігає зловмисникам обходити внутрішні засоби контролю через незахищені точки доступу.

Цей шар не зупиняє компроміс. Він обмежує радіус вибуху. Ця відмінність має значення.

**3. Раннє виявлення підозрілої активності**

Щойно зловмисники опиняються всередині, швидкість виявлення стає вирішальним фактором між незначним інцидентом та серйозним порушенням:

– IDS/IPS виявляють підозрілі моделі трафіку та спроби експлуатації.

– Виявлення та реагування на кінцеві точки виявляє аномальну

поведінку на скомпрометованих пристроях.

– Моніторинг мережевої безпеки забезпечує видимість східно-західного трафіку, який пропускають інструменти периметра.

Ці інструменти генерують сигнали. Але самі по собі сигнали не є розумінням.

**4. Перетворення сигналів на контекст та дії**

Інструменти виявлення ефективні лише тоді, коли події пов'язані між собою:

– SIEM збирає журнали, сповіщення та телеметрію з усієї мережі, кінцевих точок, програм та рівнів доступу.

– Кореляція виявляє такі закономірності, як неправильне використання облікових даних, а потім горизонтальне переміщення та доступ до даних.

– Управління вразливостями додає контекст, показуючи, чи збігається підозріла активність з відомими слабкими місцями.

Саме тут мережева безпека переходить від орієнтованої на сповіщення до орієнтованої на розслідування.

**5. Обмеження розкриття даних та впливу на бізнес**

Навіть під час активного інциденту пошкодження все ще можна контролювати:

– Запобігання втраті даних допомагає запобігти виходу конфіденційних даних з мережі.

– Дії реагування EDR ізолюють скомпрометовані кінцеві точки.

– Контроль сегментації запобігає доступу зловмисників до цінних систем.

На цьому етапі мережева безпека вже не полягає у зупинці атаки. Йдеться про збереження безперервності бізнесу.

**NDR проти IDS/IPS: у чому різниця?**

Системи виявлення та реагування на мережу (NDR), системи виявлення вторгнень (IDS) та системи запобігання вторгненням (IPS) забезпечують безпеку на рівні мережі, але вони по-різному підходять до виявлення та реагування.

NDR зосереджується на постійному аналізі мережевого трафіку та поведінки для виявлення підозрілої активності, розслідування загроз та підтримки реагування. IDS в основному виявляє підозрілий трафік та сповіщає про нього, тоді як IPS може зробити додатковий крок блокування шкідливої активності.

IDS виявляє підозрілу активність та генерує сповіщення. IPS додає можливість блокувати виявлений трафік. NDR використовує ширший підхід, аналізуючи поведінку мережі та надаючи контекст, необхідний командам безпеки для розслідування загроз, включаючи активність, яка може не відповідати відомому підпису.

Для багатьох підприємств вибір не полягає в NDR чи IDS/IPS. Їх спільне використання може забезпечити запобігання, виявлення, розслідування та реагування на різних етапах атаки.

Мережеву безпеку часто неправильно розуміють як список продуктів, що купуються за списком. Брандмауери, антивірус, SIEM, EDR, VPN. Купіть достатньо інструментів, і мережа має бути безпечною.

Такий підхід більше не працює.

Сучасні мережі є динамічними, розподіленими та постійно змінюються. Зловмисники використовують прогалини між елементами керування, а не самі елементи керування. Ось чому ефективне управління мережевою безпекою зосереджується на тому,  як рівні взаємодіють, а не на тому, скільки їх існує.

Найбільш стійкі організації застосовують інший підхід:

– Вони агресивно контролюють доступ
– Вони вважають, що компроміс можливий
– Вони обмежують бічний рух за своєю конструкцією
– Вони пріоритезують видимість над обсягом
– Вони поєднують виявлення з реагуванням

Незалежно від того, чи ви оцінюєте рішення мережевої безпеки корпоративного рівня, чи співпрацюєте з постачальниками мережевої безпеки для малого бізнесу, мета однакова.

Побудуйте архітектуру мережевої безпеки, де запобігання, виявлення та реагування підсилюють одне одного. Не тому, що це добре виглядає на схемі, а тому, що коли щось піде не так, а це станеться, мережа зможе поглинути удар і продовжити роботу.

### 3.3 Розробка функціональної схеми

**Встановлення з’єднання (Handshake) та ініціалізація захищеного контексту**

Процес узгодження параметрів безпеки між клієнтом і сервером є критичним етапом для побудови захищеного каналу зв'язку. У сучасних веб-архітектурах цей етап спрямований на узгодження шифронаборів, методів автентифікації та генерацію спільних сеансових ключів, які формують криптографічний контекст. Обмін handshake-повідомленнями у протоколі TLS здійснюється поверх рівня записів, де кожен блок має спеціальний заголовок. У контексті протидії пасивному аналізу трафіку особливу увагу приділяють структурі та метаданим початкових повідомлень, оскільки системи глибокого аналізу пакетів (DPI) здатні ідентифікувати факт встановлення захищеного з'єднання саме за відкритими полями початкових пакетів (зокрема, *ClientHello*).

Повідомлення *ClientHello* містить низку ключових параметрів:

– максимальну версію підтримуваного протоколу;

– випадкову послідовність (*ClientRandom*);

– ідентифікатор сесії (*SessionID*) для можливого прискореного відновлення з'єднання;

– перелік підтримуваних шифронаборів (*Cipher Suites*);

– набір криптографічних розширень (зокрема, *SNI – Server Name Indication*).

Наявність відкритих полів на етапові *ClientHello* створює передумови для пасивного прослуховування мережі та класифікації трафіку сторонніми спостерігачами. Саме тому розробка систем автоматизованого опортуністичного

шифрування вимагає врахування особливостей маскування або мінімізації інформаційних витоків на етапі рукостискання. У відповідь на запит сервер формує повідомлення *ServerHello*, здійснює вибір оптимального шифронабору та передає власні параметри безпеки, включно з цифровими сертифікатами (*Certificate*) та даними для обміну за протоколом Діффі-Хеллмана (*ServerKeyExchange*). Це забезпечує як автентифікацію вузлів, так і реалізацію властивості секретності вперед (Perfect Forward Secrecy, PFS), що унеможливлює розшифрування перехопленого раніше трафіку в разі компрометації довгострокових ключів сервера.

**Використання сертифікатів та криптографічних примітивів**

Для забезпечення довіри та автентифікації у розробленій системі захисту веб-ресурсів можуть використовуватися різноманітні класи TLS-сертифікатів (зокрема від провідних центрів сертифікації, таких як *RapidTLS, GeoTrust, Thawte, VeriSign, Comodo, GlobalSign* та *DigiCert*).

Базовий криптографічний стеки протоколу включає:

– Алгоритми обміну ключами та перевірки підстилу: RSA, Diffie-Hellman, ECDH, SRP, PSK.

– Механізми автентифікації: RSA, DSA, ECDSA.

– Симетричні алгоритми шифрування для захисту даних усередині сесії: AES, Camellia, а також застарілі або альтернативні рішення (RC2, RC4, IDEA, DES, Triple DES).

– Геш-функції та механізми забезпечення цілісності: сімейство SHA, MD5 тощо.

**Транспортна безпека та інкапсуляція (IPsec та супутні протоколи)**

Окрім рівня додатків (TLS), для захисту мережевого рівня використовують механізми інкапсуляції та автентифікації трафіку за допомогою протоколів IPsec (зокрема заголовок *ESP* для забезпечення конфіденційності та заголовок *AH* для захисту цілісності й автентифікації пакетів). Для динамічного керування ключами безпеки та узгодження параметрів асоціацій безпеки (*Security Associations, SA*) задіюються протоколи типу *ISAKMP/Oakley* або сучасніші реалізації обміну ключами.




Рисунок 3.2 – Функціональна схема системі

**Засоби оптимізації та мінімізація накладних витрат**

З огляду на обчислювальну складність повного циклу асиметричного рукостискання, у системі передбачено використання скороченого формату *Handshake* (Session Resumption / Session Tickets). Це дозволяє суттєво зменшити мережеві затримки та знизити навантаження на процесор при обслуговуванні великої кількості паралельних клієнтських запитів.

Водночас протокол TLS має специфічні обмеження: він не приховує факт самого встановлення з'єднання та залишає відкритими певні службові метадані (наприклад, розміри пакетів чи типи повідомлень попереджень *Alert*). Це створює інформаційне поле для потенційного пасивного аналізу. Розроблена система проактивного та автоматизованого опортуністичного шифрування покликана мінімізувати подібні ризики шляхом динамічного адаптування параметрів захисту та ускладнення сигнатурного виявлення трафіку.

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

## 3.4 Розробка діаграми процесів

Діаграму процесів розробленої системи наведено на рисунку 3.3. Її детальний аналіз дозволяє простежити характер взаємодії внутрішніх компонентів та потоків даних у межах захищеного середовища. У рамках структурного проектування застосовується модель візуалізації потоків даних (Data Flow Diagram, DFD), яка описує обробку та трансформацію інформації в системі.

Традиційно процес проектування починається з побудови контекстної діаграми потоків даних, що визначає межі системи та її взаємодію із зовнішнім середовищем (клієнтськими запитами, зовнішніми вузлами та мережевими інтерфейсами). Надалі ця модель деталізується шляхом поетапної декомпозиції процесів обробки трафіку, генерації ключів та реалізації методів автоматизованого опортуністичного шифрування.

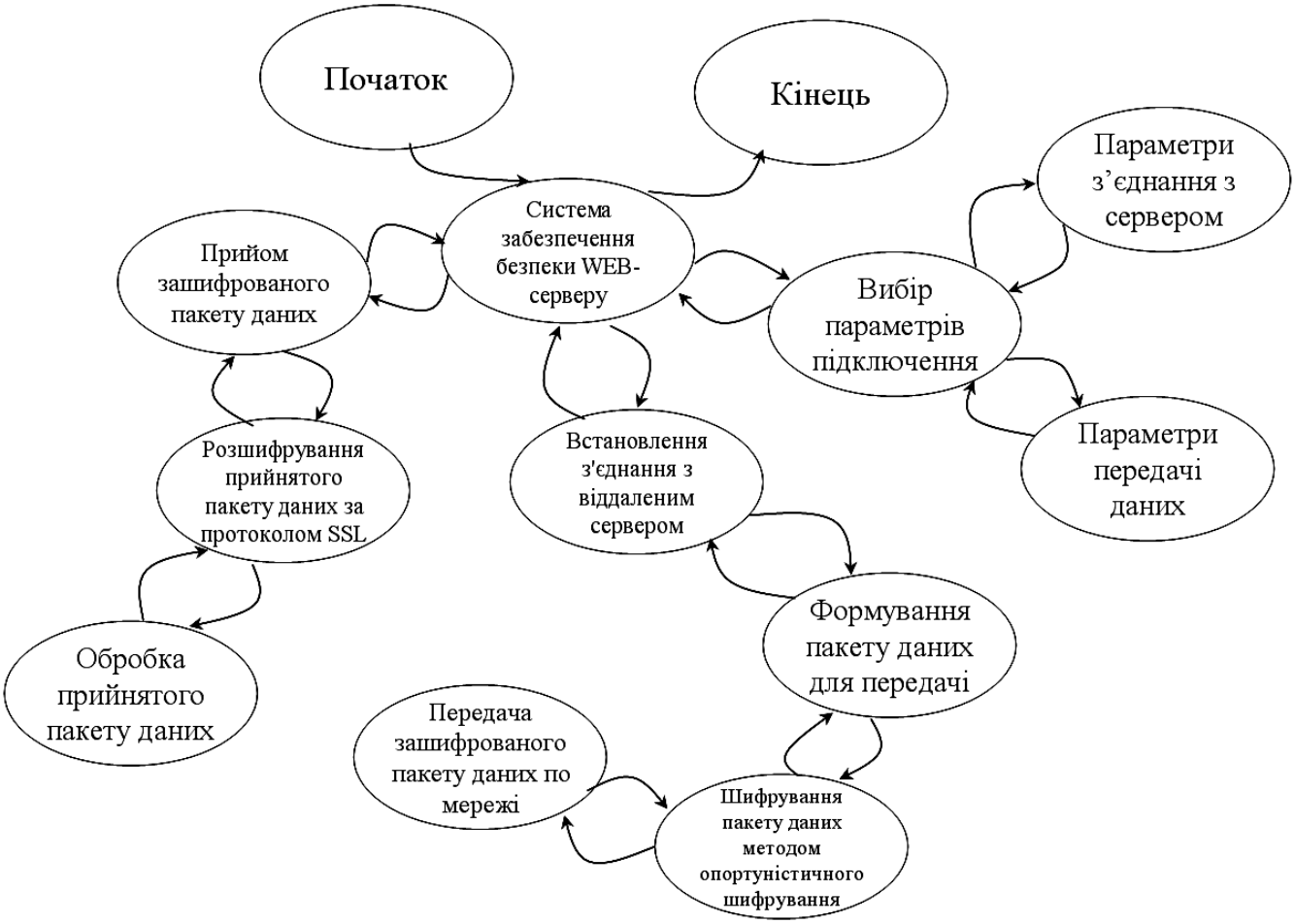


Рисунок 3.3 – Діаграма взаємодії процесів

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

Алгоритмічне забезпечення системи захисту вебресурсів від пасивного аналізу мережевого трафіку призначене для автоматизації використання криптографічно захищених з'єднань, контролю параметрів передавання інформації та запобігання розкриттю вмісту HTTP-запитів стороннім спостерігачам. Основою функціонування є застосування методів опортуністичного шифрування, які передбачають пріоритетне встановлення захищеного з'єднання за наявності відповідних можливостей у взаємодіючих компонентів. Для вебресурсів такий підхід реалізується через автоматизоване використання HTTPS, керування TLS-сертифікатами та перевірку можливості безпечного передавання даних. Важливою умовою є запобігання непомітному переходу до незашифрованого обміну під час роботи з конфіденційною інформацією.

Загальну послідовність функціонування програмного комплексу відображено на рисунку 4.1. Основний алгоритм передбачає визначення вебресурсу, перевірку його параметрів безпеки, встановлення доступності HTTPS та вибір допустимого режиму з'єднання. За наявності правильно налаштованого TLS виконується захищений обмін інформацією.

Якщо необхідні механізми відсутні, система визначає можливість автоматизованого налаштування шифрування або формує повідомлення про неможливість безпечного підключення. Подальша робота супроводжується контролем чинності сертифікатів, відповідності параметрів з'єднання встановленій політиці та реєстрацією результатів перевірки.

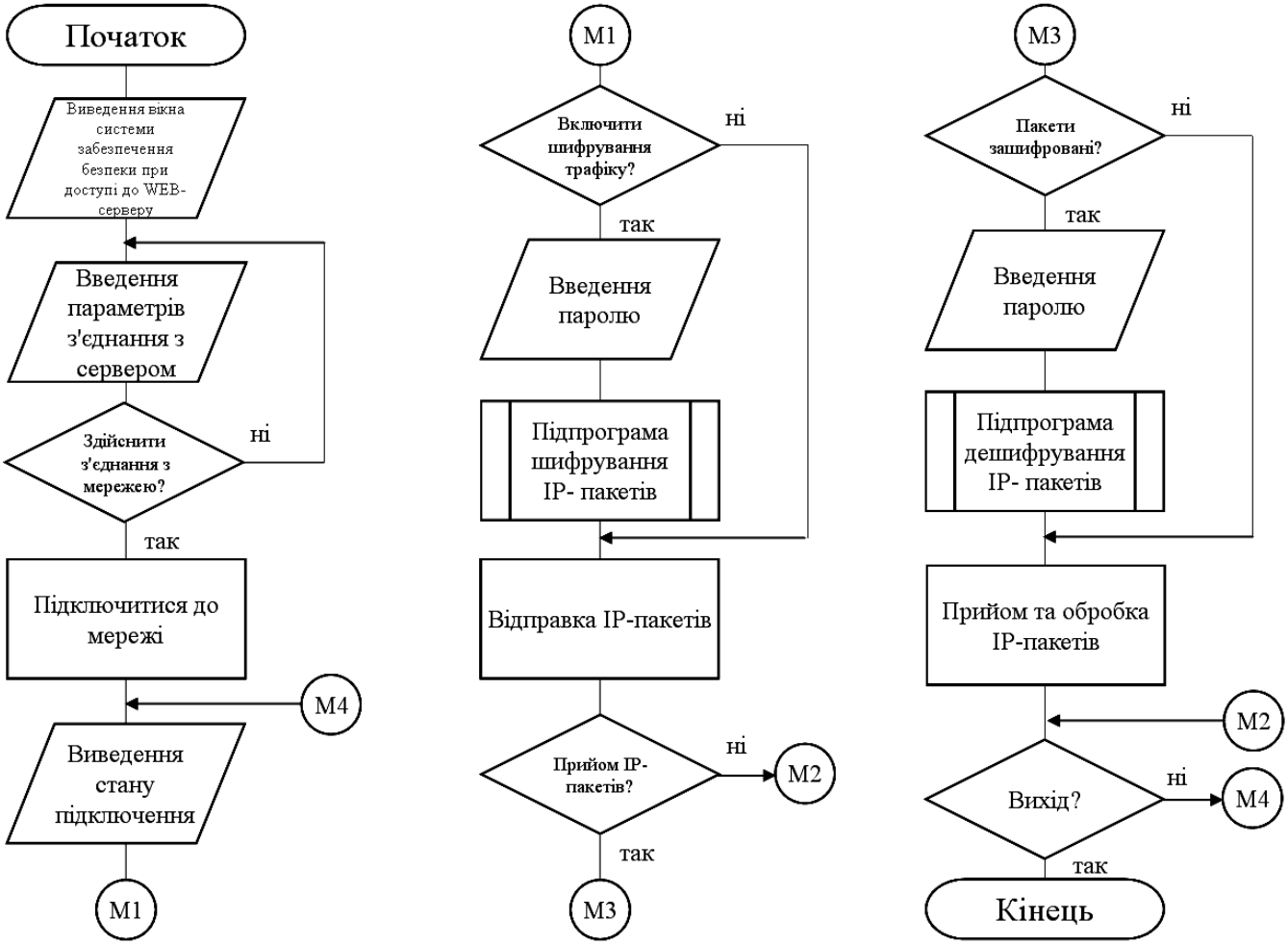


Рисунок 4.1 – Блок-схема основної програми

Функціональна архітектура системи може передбачати окремі компоненти аналізу вебресурсів, керування криптографічними параметрами, налаштування вебсервера та моніторингу захищених з'єднань. Компонент аналізу визначає можливість застосування HTTPS. Модуль керування сертифікатами забезпечує їх отримання та оновлення. Компонент конфігурування відповідає за підготовку захищених точок доступу до вебресурсів. Підсистема моніторингу перевіряє працездатність створеної конфігурації. Конкретне розміщення цих функцій залежить від архітектури програмної реалізації та наявних повноважень на керування серверною інфраструктурою.

Окремим функціональним елементом є підпрограма автоматизованого вибору захищеного режиму взаємодії. Її алгоритм представлено на рисунку 4.2.

Підпрограма отримує адресу вебресурсу, аналізує наявність підтримки TLS, перевіряє достовірність сертифіката та визначає відповідність з'єднання встановленим вимогам.

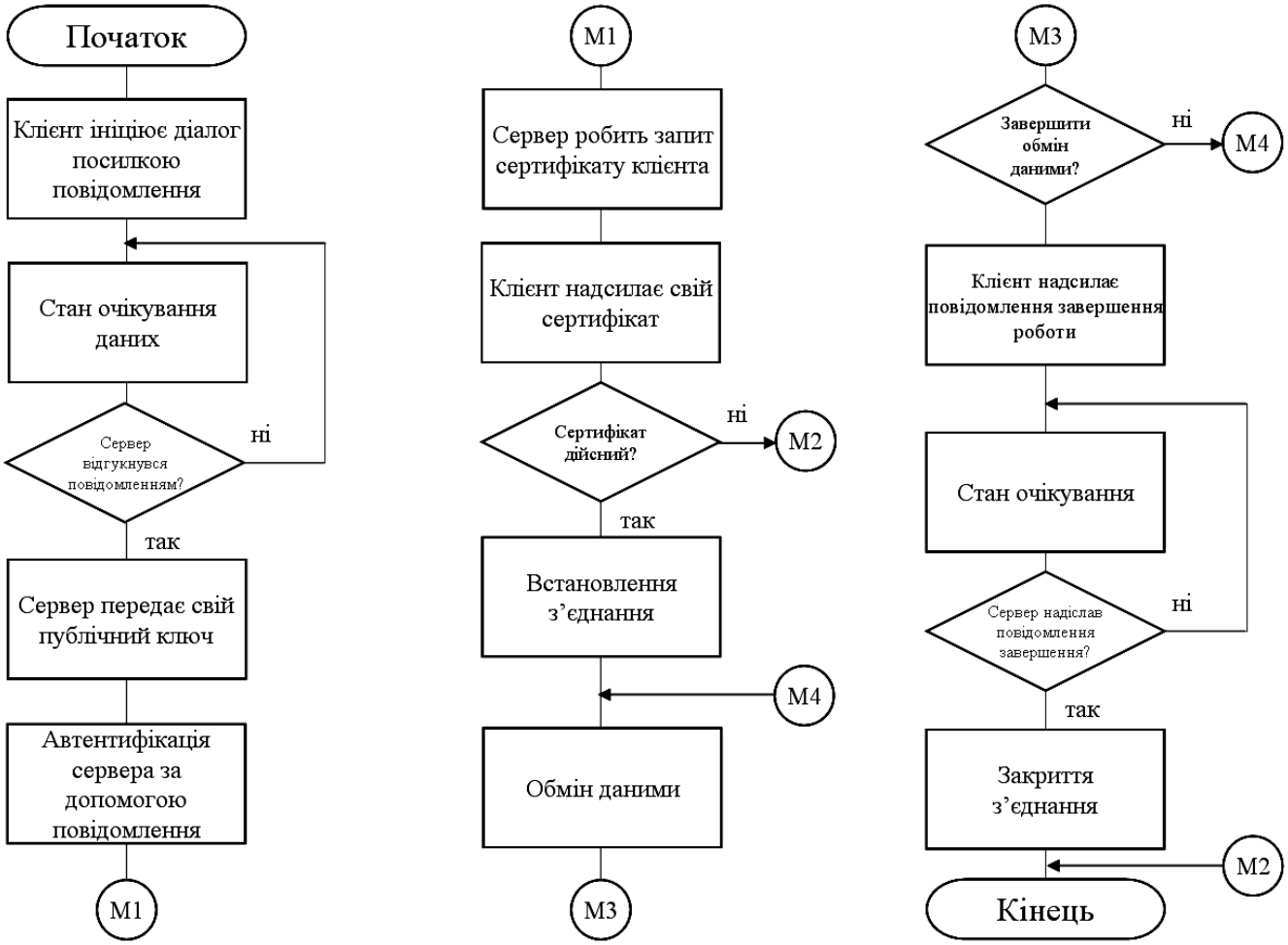


Рисунок 4.2 – Блок-схема роботи підпрограми

Якщо захищений канал можна використовувати, формується рішення про передавання даних через HTTPS. За відсутності необхідних умов виконання запиту обмежується відповідно до політики безпеки. Така організація дозволяє автоматизувати вибір режиму передавання без послаблення вимог до захисту конфіденційної інформації.

Під час проєктування алгоритмів необхідно враховувати особливості пасивного аналізу мережевого трафіку. Пасивний спостерігач отримує

інформацію з мережевих пакетів без безпосереднього втручання в їх передавання. За використання незашифрованого HTTP йому можуть бути доступні адреси запитуваних ресурсів, параметри запитів, заголовки, вміст відповідей та інша інформація прикладного рівня. Використання HTTPS дозволяє захистити цей вміст, але не приховує всі характеристики мережевого обміну. IP-адреси, обсяги переданих даних, часові характеристики та окремі відомості встановлення з'єднання можуть залишатися доступними для спостереження. Тому алгоритмічне забезпечення спрямоване насамперед на обмеження доступу до змісту переданої інформації та зменшення обсягу відкритих метаданих там, де це технічно можливо.

**Алгоритм визначення можливості захищеного доступу до вебресурсу**

Початковим етапом функціонування системи є аналіз параметрів вебресурсу та визначення можливості встановлення захищеного з'єднання. Вхідними даними алгоритму є доменне ім'я або URL, відомості про дозволені способи підключення та встановлена політика безпеки. Для вебресурсів, якими керує власник системи, додатково можуть використовуватися відомості про конфігурацію сервера та стан криптографічних сертифікатів.

Перед виконанням перевірки адреса нормалізується. Визначаються схема доступу, доменне ім'я, порт та шлях до ресурсу. Під час оброблення URL необхідно використовувати стандартний механізм синтаксичного аналізу адрес. Ручне виділення доменного імені шляхом простого пошуку символів може призводити до помилок у випадку нестандартних портів, IPv6-адрес або спеціальних символів.

Після перевірки адреси алгоритм визначає необхідний режим доступу. Якщо користувач указав HTTPS, система повинна встановлювати захищене з'єднання без спроб автоматичного переходу до HTTP після криптографічної помилки. Якщо початкова адреса використовує HTTP, перевіряється наявність правил, які дозволяють пріоритетне використання HTTPS.

Для аналізу доступності захищеного сервісу може виконуватися спроба встановлення TLS-з'єднання з відповідним вузлом. При цьому необхідно

перевіряти не лише наявність відповіді сервера, а й можливість завершення криптографічного узгодження з перевіркою його автентичності.

Якщо сервер повертає правильний сертифікат і підтримує допустиму версію TLS, ресурс визначається як придатний для захищеного доступу. Якщо сертифікат недійсний або не відповідає доменному імені, з'єднання не повинно позначатися як безпечне.

У випадку тимчасової мережевої недоступності алгоритм відокремлює транспортну помилку від відсутності підтримки шифрування. Це важливо, оскільки невдала спроба підключення не доводить, що вебсервер не підтримує HTTPS. Результатом перевірки є структурований запис про стан захищеного доступу та причину прийнятого рішення.

**Алгоритм автоматизованого переходу до HTTPS**

Пріоритетне використання HTTPS є одним з основних механізмів захисту вебтрафіку від пасивного перехоплення. Його алгоритмічна реалізація залежить від того, чи має система можливість змінювати серверну конфігурацію, чи працює лише на стороні клієнта.

Для керованих вебресурсів доцільно передбачити автоматичне перенаправлення HTTP-запитів на відповідні HTTPS-адреси. Після отримання запиту вебсервер формує відповідь із кодом перенаправлення та новою адресою ресурсу. Для постійного перенаправлення можуть використовуватися відповідні коди HTTP з урахуванням необхідності збереження методу запиту.

Важливо, що початковий HTTP-запит у такому сценарії передається незашифрованим. Тому перенаправлення не забезпечує повного захисту першого звернення. Пасивний спостерігач може отримати інформацію з початкового запиту, а активний порушник здатний намагатися перешкодити переходу до HTTPS.

Для зменшення відповідного ризику необхідно забезпечувати формування початкових посилань із використанням HTTPS. Додатково можуть застосовуватися браузерні механізми пріоритетного захищеного підключення та політика HTTP Strict Transport Security.

Якщо система реалізується у вигляді клієнтського компонента, вона може перевіряти можливість заміни HTTP-адреси на HTTPS до передавання запиту. Проте звичайний вебзастосунок не має повноважень довільно перехоплювати й змінювати всі мережеві запити браузера. Тому така функціональність потребує відповідного середовища виконання, наприклад спеціалізованого клієнтського застосунку або браузерного розширення з необхідними дозволами.

Алгоритм переходу повинен зберігати логічну структуру запитуваної адреси. Довільна зміна доменного імені, шляху або параметрів запиту може призводити до звернення до іншого ресурсу. Особливу увагу необхідно приділяти перевірці адрес перенаправлення, отриманих від сторонніх серверів.

Якщо HTTPS доступний і відповідає вимогам безпеки, подальша взаємодія виконується захищеним каналом. Якщо встановлення такого з'єднання неможливе, алгоритм застосовує визначену політику відмови. Для ресурсів, які обробляють облікові дані або іншу конфіденційну інформацію, повернення до відкритого HTTP не допускається.

**Алгоритм установлення та перевірки TLS-з'єднання**

Після визначення захищеної адреси виконується встановлення TLS-з'єднання. Його призначення полягає у формуванні криптографічно захищеного каналу між клієнтом і вебсервером. Для цього використовуються механізми протоколу Transport Layer Security.

На початковому етапі клієнт надсилає повідомлення, яке містить підтримувані параметри криптографічного узгодження. Сервер обирає сумісні параметри та передає необхідні відомості для підтвердження власної ідентичності. Подальша послідовність залежить від узгодженої версії TLS.

Для нових конфігурацій доцільно надавати перевагу TLS 1.3. Цей протокол передбачає механізми захищеного узгодження ключів і криптографічного захисту прикладного трафіку. Його принципи визначено у специфікації RFC 8446 та наступних нормативних оновленнях.

Важливим етапом є перевірка серверного сертифіката. Алгоритм повинен контролювати відповідність доменного імені, чинність сертифіката та можливість

побудови довіреного ланцюга сертифікації. Для перевірки імені використовуються відповідні ідентифікатори сертифіката, зокрема Subject Alternative Name.

Якщо сертифікат підписаний невідомим центром сертифікації або термін його чинності завершився, з'єднання не вважається достовірно автентифікованим. Аналогічне рішення приймається за невідповідності імені сервера.

Для забезпечення коректної роботи системи перевірка сертифікатів не повинна вимикатися з метою автоматичного обходу помилки. Інакше шифрування може застосовуватися до з'єднання з непідтвердженим сервером, що створює ризик підміни кінцевого вузла.

Після успішного завершення криптографічного узгодження формується захищений канал передавання HTTP-повідомлень. Прикладні дані шифруються засобами TLS, а їхня цілісність контролюється відповідними криптографічними механізмами.

Результатом алгоритму є встановлене з'єднання з підтвердженими параметрами безпеки або контрольована відмова із зазначенням причини.

**Алгоритм автоматизованого отримання TLS-сертифікатів**

Для вебресурсів, якими керує система, важливим завданням є автоматизація отримання криптографічних сертифікатів. Ручне встановлення сертифікатів ускладнює супроводження великої кількості доменів і створює ризик завершення терміну їхньої чинності без своєчасного оновлення.

Для вирішення цього завдання може застосовуватися протокол Automatic Certificate Management Environment (ACME). Його механізми визначені в RFC 8555. Протокол дозволяє автоматизувати підтвердження контролю над доменом, отримання сертифіката та окремі операції керування ним.

Вхідними даними алгоритму є доменне ім'я, параметри облікового запису ACME, конфігурація вебсервера та встановлена політика керування ключами. Перед початком операції перевіряється право системи змінювати конфігурацію відповідного домену.

На першому етапі визначається наявність чинного сертифіката. Якщо сертифікат відповідає домену та має достатній залишковий термін чинності, повторне отримання може бути відкладене. Якщо сертифікат відсутній або потребує оновлення, ініціюється відповідна процедура ACME.

Далі формується необхідний ключовий матеріал. Закритий ключ повинен створюватися за допомогою криптографічно стійкого генератора випадкових значень. Його збереження здійснюється із застосуванням обмежень доступу. Закритий ключ не повинен передаватися стороннім сервісам або записуватися у відкриті журнали.

Наступним етапом є формування запиту на сертифікацію. ACME-клієнт передає відповідні відомості центру сертифікації та отримує завдання для підтвердження контролю над доменом.

Для перевірки може використовуватися механізм HTTP-01. Він передбачає розміщення визначеного перевірочного вмісту за спеціальною HTTP-адресою. Центр сертифікації звертається до цього ресурсу та перевіряє відповідність отриманих даних.

Альтернативним способом є DNS-01, за якого підтвердження виконується через спеціальний DNS-запис. Такий варіант може застосовуватися за наявності доступу до керування DNS-зоною та підтримки відповідних операцій. Він також використовується для сценаріїв, у яких необхідно отримувати сертифікати для груп доменних імен.

Після успішного підтвердження контролю над доменом центр сертифікації може видати сертифікат. Отриманий результат перевіряється на відповідність запитаним іменам та очікуваним характеристикам. Лише після цього дозволяється його встановлення у конфігурацію вебсервера.

Якщо підтвердження домену не вдалося, алгоритм фіксує причину помилки. До можливих причин належать неправильна DNS-конфігурація, недоступність перевірочного ресурсу або відсутність необхідних повноважень. Повторні спроби повинні виконуватися з урахуванням обмежень центру сертифікації.

Результатом алгоритму є отриманий сертифікат, придатний для використання у захищеному вебз'єднанні, або повідомлення про неможливість завершення процедури.

**Алгоритм автоматичного оновлення сертифікатів**

Після встановлення сертифіката необхідно забезпечити контроль його життєвого циклу. Термін чинності сертифікатів є обмеженим, тому їх необхідно своєчасно замінювати. Для цього передбачається алгоритм періодичної перевірки та автоматичного оновлення.

Вхідними даними є реєстр доменів, відомості про чинні сертифікати та встановлена політика оновлення. Для кожного сертифіката визначається залишковий термін чинності. Якщо він перевищує встановлену межу, додаткові дії не виконуються.

У випадку наближення до завершення терміну чинності запускається процедура оновлення через відповідний центр сертифікації. Після успішного отримання нового сертифіката виконується перевірка його характеристик.

Перед заміною необхідно переконатися, що новий сертифікат відповідає конфігурації вебсервера та пов'язаний із правильним закритим ключем. Далі здійснюється підготовка оновленої конфігурації.

Заміна не повинна порушувати роботу інших доменів, які обслуговуються тим самим сервером. Тому доцільно використовувати попередню перевірку конфігурації та контрольоване застосування змін.

Після активації нового сертифіката виконується тестове TLS-підключення. Якщо сервер надає очікуваний сертифікат і з'єднання успішно встановлюється, операція вважається завершеною.

У разі помилки застосування конфігурації необхідно зберегти можливість повернення до попереднього працездатного стану, якщо відповідний сертифікат ще чинний. Якщо термін чинності попереднього сертифіката завершився, використання його як постійного резервного варіанта є недопустимим.

Результатом алгоритму є підтримання актуального стану сертифікатів та зменшення ризику переривання захищеного доступу через завершення їхнього терміну чинності.

**Алгоритм автоматизованого конфігурування вебсервера**

Після отримання сертифіката необхідно налаштувати серверний компонент, який забезпечує HTTPS-доступ. Як можливий варіант реалізації може використовуватися Nginx, Apache HTTP Server або інший вебсервер із підтримкою TLS.

Вхідними даними алгоритму є доменне ім'я, параметри мережевого доступу, шляхи до сертифіката та закритого ключа, а також встановлені вимоги до криптографічної конфігурації.

На першому етапі визначається відповідний віртуальний сервер. Якщо ресурс уже має HTTPS-конфігурацію, перевіряється її відповідність актуальним параметрам. Якщо захищений доступ відсутній, формується необхідна конфігурація.

Вона повинна містити параметри приймання TLS-з'єднань, посилання на сертифікат, дозволені версії протоколу та налаштування оброблення HTTP-запитів. Для вебресурсів, що працюють через зворотний проксі, необхідно також визначити адресу внутрішнього прикладного сервісу.

Особливу увагу слід приділити захисту з'єднання між проксі-сервером і внутрішнім застосунком. Завершення TLS на зовнішньому проксі не означає автоматичного шифрування подальшого обміну. Якщо внутрішній трафік проходить через мережу, доступну стороннім спостерігачам, необхідно передбачити відповідний захищений транспорт.

Після формування конфігурації виконується синтаксична перевірка. За наявності помилок нові параметри не застосовуються. Це дозволяє уникати порушення роботи сервера через некоректні директиви.

У разі успішної перевірки може виконуватися контрольоване перечитування конфігурації без повного завершення всіх поточних з'єднань, якщо така можливість підтримується вебсервером.

Після застосування змін виконується повторна перевірка HTTPS-доступу. Контролюється правильність сертифіката, відповідність доменного імені та доступність запитуваного ресурсу.

Результатом алгоритму є налаштована точка захищеного доступу до вебресурсу.

**Алгоритм застосування політики HTTP Strict Transport Security**

Для зменшення ризику повторного використання незахищеного HTTP може застосовуватися механізм HTTP Strict Transport Security (HSTS). Його призначення полягає в повідомленні браузеру про необхідність використовувати захищене з'єднання під час наступних звернень до відповідного вебресурсу.

Алгоритм передбачає перевірку готовності домену до постійного використання HTTPS. Перед активацією політики необхідно підтвердити доступність усіх критичних функцій вебресурсу через захищений канал.

Після успішної перевірки сервер налаштовується на передавання заголовка Strict-Transport-Security у захищених відповідях. Параметр max-age визначає тривалість дії політики у браузері. За необхідності може використовуватися директива includeSubDomains.

Використання includeSubDomains потребує попередньої перевірки піддоменів. Якщо частина з них не підтримує HTTPS, застосування відповідної політики може призвести до втрати доступності цих ресурсів.

Після отримання заголовка через коректне HTTPS-з'єднання браузер, який підтримує HSTS, зберігає відповідну політику. Подальші звернення з HTTP-адресами можуть автоматично перетворюватися на HTTPS до передавання запиту мережею.

Водночас HSTS не забезпечує повного захисту першого звернення, якщо браузер раніше не отримував відповідної політики. Для усунення цього обмеження існують механізми попереднього включення доменів до списків HSTS Preload. Їх використання потребує виконання встановлених вимог та окремого процесу включення домену.

Алгоритм повинен враховувати, що автоматичне ввімкнення HSTS для невідомого ресурсу без перевірки всієї доменної інфраструктури може створювати експлуатаційні проблеми. Тому зміна політики виконується лише після підтвердження готовності відповідного домену.

**Алгоритм контролю незахищених компонентів вебсторінки**

Навіть за успішного встановлення HTTPS-з'єднання вебсторінка може містити посилання на ресурси, які завантажуються через HTTP. До таких ресурсів належать зображення, стилі, сценарії та інші зовнішні компоненти. Передавання відповідних даних незахищеним каналом може розкривати їхній вміст сторонньому спостерігачу.

Для виявлення таких ситуацій передбачається алгоритм аналізу адрес залежних ресурсів. Вхідними даними є структура вебсторінки або її доступне представлення, а також відомості про використовувані зовнішні ресурси.

На першому етапі визначаються адреси вбудованих компонентів. Далі виконується перевірка схеми доступу до кожного ресурсу. Якщо використовується HTTPS, відповідна адреса може вважатися придатною для захищеного завантаження за умови успішної перевірки з'єднання.

Для HTTP-адрес перевіряється можливість доступу до того самого ресурсу через HTTPS. Якщо захищений варіант доступний і коректний, система може сформувати рекомендацію або виконати дозволене оновлення посилання.

Автоматична заміна повинна враховувати, що однаковий шлях на HTTP- та HTTPS-серверах не завжди повертає ідентичний вміст. Тому для критичних компонентів бажано перевіряти відповідність очікуваного ресурсу.

Браузери мають власні механізми обмеження mixed content. Частина незахищених ресурсів може автоматично переводитися на HTTPS, а інші блокуються. Проте поведінка залежить від типу ресурсу та реалізації браузера.

Як додатковий засіб можна використовувати Content Security Policy. Директива upgrade-insecure-requests дозволяє браузеру перетворювати відповідні незахищені адреси на HTTPS. Проте вона не створює підтримку TLS на сервері, який її не має.

Для взаємодії через WebSocket необхідно забезпечувати використання wss замість незахищеного ws у відповідних захищених сценаріях.

Результатом алгоритму є визначення незахищених залежностей та формування правил їхнього усунення або блокування.

**Алгоритм захисту DNS-запитів і зменшення розкриття метаданих**

Використання HTTPS забезпечує шифрування вмісту прикладного обміну, але не обов'язково приховує доменне ім'я вебресурсу від стороннього спостерігача. Під час звичайного DNS-запиту інформація про домен може передаватися відкритим каналом.

Для зменшення відповідного ризику доцільно розглядати використання захищеного DNS-транспорту. Як можливі технології застосовуються DNS over HTTPS (DoH) або DNS over TLS (DoT). Вони дозволяють шифрувати взаємодію між DNS-клієнтом і відповідним резолвером.

Алгоритм починається з перевірки доступного способу DNS-резолюції. Якщо середовище підтримує захищений DNS і його використання відповідає політиці системи, запити можуть передаватися відповідним каналом.

При цьому необхідно враховувати межі захисту. DoH і DoT приховують вміст DNS-запитів від пасивного спостерігача на ділянці між клієнтом і резолвером. Проте сам резолвер отримує відомості про запитувані домени. Крім того, інші характеристики мережевого обміну можуть дозволяти робити висновки про відвідувані ресурси.

Додатковим механізмом є використання DNS-записів HTTPS та SVCB. Вони можуть містити інформацію про параметри захищеного доступу, альтернативні кінцеві точки та підтримувані протоколи. Відповідні можливості визначено в RFC 9460.

У разі підтримки клієнтом такі записи можуть допомагати встановлювати захищений канал без зайвих проміжних звернень. Однак їхня наявність не замінює перевірку сертифіката сервера. Крім того, можливість отримання відповідної інформації залежить від підтримки DNS-клієнта та резолвера.

Для додаткового захисту доменного імені під час TLS-узгодження може розглядатися Encrypted ClientHello (ECH), якщо відповідні компоненти підтримують цю технологію. Її призначення полягає у приховуванні чутливої частини початкового повідомлення TLS.

Водночас використання ECH не гарантує повного приховування відомостей про вебресурс. IP-адреса сервера, часові характеристики та обсяги трафіку можуть залишатися доступними для аналізу.

Результатом алгоритму є визначення доступних додаткових механізмів захисту метаданих із урахуванням можливостей конкретної мережевої інфраструктури.

**Алгоритм прийняття рішення щодо опортуністичного шифрування**

Центральним завданням системи є автоматизований вибір захищеного режиму передавання. Для цього необхідно враховувати технічну доступність TLS та вимоги до конфіденційності конкретного ресурсу.

Поняття опортуністичної безпеки розглядається в RFC 7435. Воно передбачає використання доступних механізмів шифрування та автентифікації з урахуванням можливостей взаємодіючих сторін. Водночас опортуністичне шифрування без підтвердження автентичності не забезпечує захисту від активної підміни вузла.

Для вебресурсів, які обробляють облікові дані або іншу конфіденційну інформацію, доцільно використовувати суворішу політику. Вона передбачає обов'язкове встановлення автентифікованого HTTPS-з'єднання та заборону переходу до HTTP у разі помилки.

Вхідними даними алгоритму є тип ресурсу, його категорія захисту, результат перевірки HTTPS та встановлена політика оброблення. На першому етапі визначається допустимість незахищеного доступу.

Для захищених ресурсів використовується режим обов'язкового шифрування. Якщо TLS недоступний або не проходить перевірку, виконання запиту припиняється. Для публічних ресурсів без конфіденційного вмісту може

бути передбачена окрема політика сумісності, якщо така поведінка допускається вимогами системи.

Проте дозвіл на відкритий доступ не повинен автоматично поширюватися на форми автентифікації, персональні дані або інші чутливі операції. Необхідно також враховувати, що HTTP не забезпечує захисту цілісності вмісту від активного втручання.

Якщо підтримка HTTPS підтверджена, алгоритм формує рішення про використання захищеного каналу. Якщо відповідний ресурс належить до керованої інфраструктури та HTTPS ще не налаштований, може ініціюватися процедура автоматизованого отримання сертифіката й підготовки серверної конфігурації.

Для сторонніх ресурсів, якими система не керує, автоматичне налаштування серверного TLS без відповідних повноважень неможливе. У такому випадку система обмежується перевіркою доступності захищеного способу підключення та застосуванням клієнтської політики.

Результатом алгоритму є визначений режим доступу та обґрунтоване рішення щодо можливості передавання інформації.

**Алгоритм запобігання небезпечному зниженню рівня захисту**

Під час використання автоматизованого шифрування необхідно враховувати ризик downgrade-атак. Вони передбачають примусове використання слабшого режиму захисту або незашифрованого передавання замість доступного захищеного каналу.

На відміну від пасивного спостереження, така атака потребує активного впливу на мережеву взаємодію. Проте аналіз цього сценарію необхідний, оскільки можливість непомітного зниження рівня захисту безпосередньо впливає на стійкість опортуністичного підходу.

Алгоритм починається з перевірки історії захищеного доступу до ресурсу. Якщо для домену діє відповідна політика HSTS, використання незашифрованого HTTP не допускається.

Далі перевіряється результат TLS-узгодження. Якщо сервер повідомляє про неможливість використання допустимих криптографічних параметрів, система не повинна автоматично дозволяти застарілі або заборонені алгоритми лише для завершення підключення.

Окремо контролюються помилки сертифікації. Невідповідність доменного імені, недовірений ланцюг сертифікації або недійсний сертифікат не повинні автоматично ігноруватися.

Для системи автоматизованого налаштування вебсервера важливо перевіряти відсутність незахищених альтернативних точок доступу до конфіденційних ресурсів. Наприклад, прикладний сервіс не повинен залишатися доступним через відкритий мережевий порт в обхід налаштованого HTTPS-проксі, якщо це суперечить політиці безпеки.

Якщо виявлено спробу або ознаки зниження рівня захисту, алгоритм реєструє подію та застосовує відповідне обмеження. Для критичних ресурсів передавання інформації припиняється до відновлення допустимих умов.

Результатом є контроль відповідності фактичного режиму з'єднання встановленим криптографічним вимогам.

**Алгоритм контролю стану захищених вебз'єднань**

Після ввімкнення HTTPS необхідно забезпечити регулярну перевірку працездатності захищеного доступу. Для цього передбачається алгоритм моніторингу TLS-конфігурації та доступності вебресурсів.

Вхідними даними є перелік контрольованих доменів, параметри очікуваних з'єднань і встановлена періодичність перевірок. Для кожного ресурсу виконується тестове звернення з використанням HTTPS.

На першому етапі контролюється можливість встановлення TCP-з'єднання або відповідного транспортного каналу. Далі перевіряється успішність TLS-узгодження та достовірність сертифіката.

Після встановлення захищеного каналу може виконуватися обмежений HTTP-запит для перевірки доступності відповідного ресурсу. При цьому необхідно відокремлювати транспортні помилки від прикладних відповідей

сервера. Наприклад, код помилки прикладного рівня не завжди означає порушення TLS.

Окремо аналізується термін чинності сертифіката. За наближення до встановленого порога формується завдання на перевірку автоматичного оновлення. Якщо оновлення вже виконується, система повинна уникати запуску дубльованих процедур для того самого домену.

Для зменшення впливу короткочасних мережевих порушень доцільно використовувати повторні перевірки. Однак кількість спроб повинна бути обмеженою, щоб сам моніторинг не створював надмірного навантаження.

Результатом алгоритму є актуалізована інформація про стан захищеного доступу. Вона використовується для формування експлуатаційних сповіщень і запуску допустимих процедур відновлення.

**Алгоритм оброблення помилок і відновлення конфігурації**

Автоматизація криптографічного захисту потребує контрольованої обробки помилок. Вони можуть виникати під час отримання сертифіката, перевірки домену, встановлення TLS-з'єднання або застосування серверної конфігурації.

Для кожної операції необхідно визначати її початковий і кінцевий стани. Наприклад, процес налаштування HTTPS може перебувати у стані підготовки, отримання сертифіката, перевірки конфігурації або завершення. Якщо на певному етапі виникає помилка, система фіксує її причину та доступний результат попередніх дій.

Особливу увагу необхідно приділяти частково виконаним операціям. Наприклад, сертифікат може бути успішно отриманий, але налаштування вебсервера завершиться помилкою. У такому випадку повторне отримання сертифіката не завжди потрібне. Доцільніше повторити перевірку або застосування конфігурації.

Для запобігання конфліктним змінам необхідно обмежувати одночасне виконання операцій над одним доменом. Повторний запит із тим самим

ідентифікатором повинен повертати відомості про наявну операцію або її результат, а не створювати незалежну процедуру налаштування.

Перед застосуванням змін доцільно зберігати попередню працездатну конфігурацію. Якщо нова конфігурація не проходить перевірку, вона не повинна замінювати чинну.

Після успішного застосування параметрів необхідно провести перевірку фактичного HTTPS-доступу. Якщо сервер не надає очікуваного сертифіката або не обслуговує запити, може виконуватися повернення до попередніх коректних налаштувань.

Водночас повернення конфігурації не повинно відновлювати небезпечні параметри, які були заборонені політикою безпеки. За неможливості відновити захищений доступ система повинна повідомити про порушення та не створювати незашифрований альтернативний маршрут до конфіденційних ресурсів.

**Особливості програмної реалізації та інформаційної безпеки**

Практична реалізація розглянутих алгоритмів може передбачати керувальний сервіс, модуль ACME, компонент аналізу TLS та засоби взаємодії з конфігурацією вебсервера. Конкретні мови програмування, бібліотеки та способи розгортання визначаються прийнятою архітектурою.

Для виконання перевірок HTTPS можуть використовуватися програмні бібліотеки TLS, які підтримують перевірку сертифікатів і доступних параметрів з'єднання. Для автоматизованого керування сертифікатами доцільно використовувати сумісний ACME-клієнт. Як можливі інструменти розглядаються Certbot та інші програмні засоби з відповідною функціональністю.

Для серверного розгортання можуть застосовуватися Nginx або Apache HTTP Server. Вони забезпечують приймання HTTPS-з'єднань, використання сертифікатів і перенаправлення запитів до прикладних сервісів. Якщо реалізація базується на зворотному проксі, алгоритми конфігурування повинні враховувати його власні механізми перевірки та застосування налаштувань.

Важливо обмежувати права керувального компонента. Автоматизація оновлення сертифікатів не повинна вимагати необмеженого доступу до всіх

файлів операційної системи. Доцільно розмежовувати повноваження на керування доменами, сертифікатами та конфігурацією вебсервера.

Закриті криптографічні ключі повинні зберігатися із застосуванням обмежень доступу. Їхній вміст не допускається записувати до журналів або передавати через незахищені програмні інтерфейси.

Окремо необхідно контролювати використання користувацьких сеансів. Для вебресурсів, що працюють через HTTPS, доцільно встановлювати атрибут Secure для відповідних cookie. Додаткові атрибути HttpOnly та SameSite застосовуються залежно від призначення cookie та вимог прикладної безпеки.

Потрібно також враховувати, що захист від пасивного аналізу не означає повної анонімності користувача. Навіть за використання сучасних криптографічних механізмів зовнішній спостерігач може аналізувати інтенсивність обміну, розміри пакетів і часові закономірності. Усунення таких ознак потребує окремих методів захисту метаданих, які не забезпечуються самим фактом використання HTTPS.

**Алгоритм оцінювання ефективності захисту**

Для визначення практичної придатності запропонованих алгоритмів необхідно передбачити перевірку їхньої роботи за різних умов мережевої взаємодії. Основним завданням є встановлення здатності системи забезпечувати захищене передавання інформації та правильно реагувати на відсутність необхідних криптографічних умов.

На початковому етапі перевіряється доступність HTTPS для контрольованих вебресурсів. Для цього можуть використовуватися тестові сервери з правильною та навмисно некоректною TLS-конфігурацією.

Окремо перевіряються сценарії використання недійсного сертифіката, невідповідності доменного імені та завершення терміну чинності. Очікуваним результатом є відмова у встановленні з'єднання, яке не відповідає обов'язковим вимогам безпеки.

Для перевірки автоматизованого отримання сертифікатів доцільно використовувати контрольоване середовище ACME. У ньому досліджується

правильність виконання перевірки домену, оброблення помилок та застосування нової серверної конфігурації.

Під час перевірки захисту від пасивного спостереження можуть порівнюватися мережеві дампи HTTP- та HTTPS-сеансів. У першому випадку аналізується доступність незашифрованого прикладного вмісту. У другому визначається, які відомості залишаються видимими без доступу до сеансових криптографічних ключів.

Важливо розмежовувати вміст HTTP-повідомлень і мережеві метадані. Відсутність відкритого тексту запиту в пакеті не означає приховування IP-адреси сервера або часових характеристик з'єднання.

Для оцінювання додаткових витрат доцільно вимірювати тривалість встановлення з'єднання, затримку отримання відповіді та використання обчислювальних ресурсів. Порівняння повинно проводитися за однакових умов навантаження та мережевого середовища.

Окремо перевіряється поведінка системи під час недоступності ACME-сервера або помилки оновлення сертифіката. Необхідно встановити, чи зберігається чинна працездатна конфігурація та чи формується відповідне повідомлення про проблему.

Перевірка механізму запобігання downgrade-атакам може виконуватися шляхом моделювання блокування HTTPS або зміни параметрів початкового підключення. Для ресурсів із обов'язковим шифруванням очікуваною реакцією є припинення передавання конфіденційної інформації, а не перехід до HTTP.

Результати відповідних випробувань можуть використовуватися для визначення частки успішно захищених підключень, кількості відхилених небезпечних з'єднань та додаткових витрат на криптографічний захист. Кількісні висновки щодо ефективності можуть бути сформовані лише після отримання експериментальних даних.

Розглянуте алгоритмічне забезпечення визначає послідовність автоматизованого захисту вебресурсів, починаючи з перевірки доступності HTTPS і завершуючи контролем стану захищених з'єднань. Поєднання

опортуністичного вибору шифрованого каналу з автоматизованим керуванням сертифікатами, політиками TLS та механізмами запобігання небезпечному зниженню рівня захисту створює основу для обмеження можливостей пасивного аналізу прикладного мережевого трафіку. Практична ефективність запропонованих алгоритмів залежить від конфігурації вебсерверів, підтримки криптографічних механізмів клієнтським програмним забезпеченням і правильності застосування встановлених політик безпеки.

# 5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Програмне забезпечення системи захисту вебресурсів від пасивного аналізу мережевого трафіку призначене для автоматизованого налаштування криптографічно захищених з'єднань та контролю безпеки передавання інформації.

Його функціональність передбачає перевірку підтримки HTTPS, керування TLS-сертифікатами, налаштування параметрів вебсерверів і виявлення незахищених способів доступу до ресурсів.

Застосування методів опортуністичного шифрування дозволяє надавати перевагу захищеному передаванню даних відповідно до можливостей серверної інфраструктури. Головне вікно програми показане на рисунку 5.1.

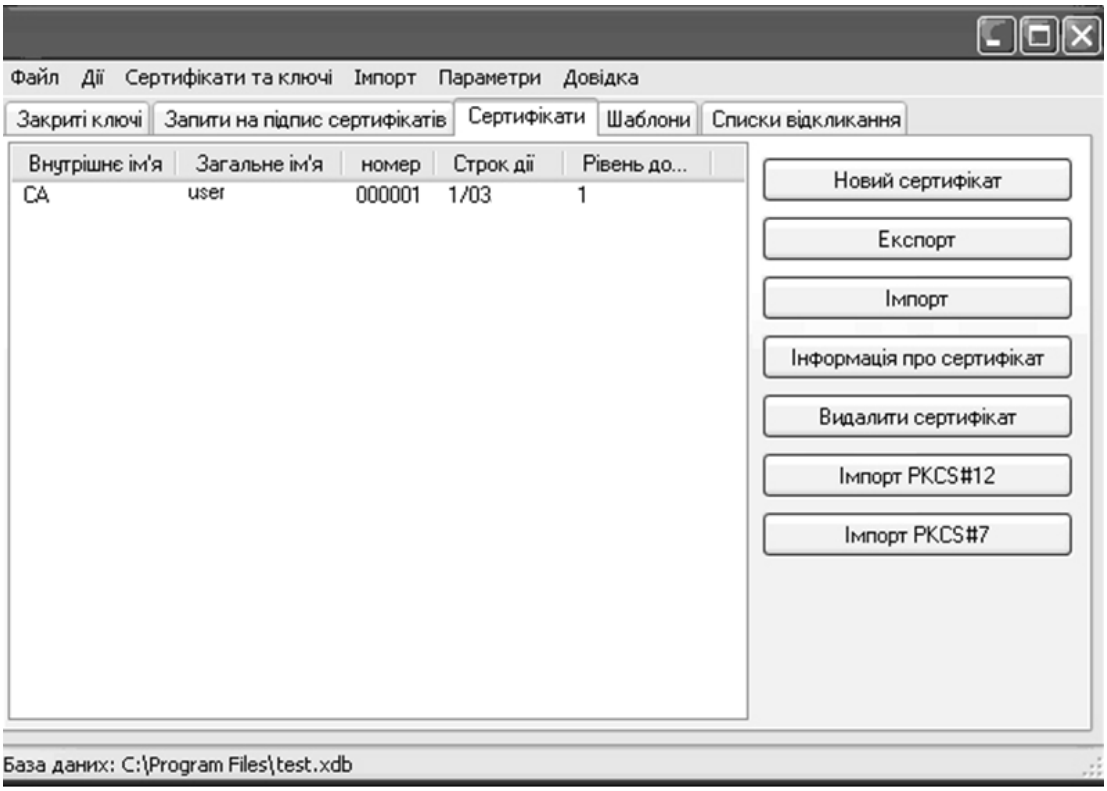


Рисунок 5.1 – Головне вікно ПЗ

Впровадження програмного комплексу передбачає підготовку серверного середовища, налаштування мережевої взаємодії та підключення засобів автоматизованого керування HTTPS. Система може використовуватися для адміністрування вебресурсів, до конфігурації яких користувач має відповідні повноваження. Для сторонніх ресурсів можливості обмежуються перевіркою доступності захищеного з'єднання та застосуванням допустимої клієнтської політики.

Як серверне середовище доцільно використовувати операційну систему Linux із підтримкою необхідних криптографічних бібліотек і мережевих служб. Для обслуговування HTTPS-запитів може застосовуватися Nginx або Apache HTTP Server. Вони забезпечують приймання TLS-з'єднань, використання сертифікатів і передавання запитів до прикладних сервісів. Конкретний вебсервер та спосіб інтеграції визначаються фактичною програмною реалізацією.

Для автоматизованого отримання сертифікатів може використовуватися протокол Automatic Certificate Management Environment (ACME). Одним із доступних програмних інструментів є Certbot. Він дозволяє виконувати процедури підтвердження контролю над доменом, отримання сертифікатів та їх оновлення. У разі застосування іншого ACME-клієнта послідовність підготовки визначається його підтримуваними функціями.

Апаратні вимоги залежать від кількості контрольованих вебресурсів, інтенсивності HTTPS-запитів та обсягу службової інформації. Сервер повинен мати достатні обчислювальні ресурси для виконання криптографічних операцій. Необхідно також забезпечити стабільне мережеве з'єднання, доступ до DNS-інфраструктури та можливість установлення з'єднань із відповідними центрами сертифікації.

Підготовка середовища починається з перевірки налаштувань доменних імен. Для кожного керованого вебресурсу необхідно переконатися, що DNS-записи відповідають фактичним мережевим адресам сервера. Далі перевіряється доступність необхідних мережевих портів. Для звичайного HTTPS

використовується TCP-порт 443, а для HTTP може застосовуватися порт 80. Його доступність необхідна, зокрема, для виконання перевірки ACME HTTP-01.

Після перевірки мережевих параметрів встановлюються необхідні програмні залежності та засоби керування сертифікатами. Для керувального компонента визначаються адреси вебсерверів, перелік доменів і права на зміну конфігурації. Важливо обмежити доступ до адміністративних інтерфейсів. Програмне забезпечення не повинно отримувати ширші системні повноваження, ніж необхідно для виконання його функцій.

Наступним етапом є налаштування взаємодії з ACME-сервісом. Для цього визначається відповідний центр сертифікації та спосіб підтвердження контролю над доменом. За використання HTTP-01 необхідно забезпечити доступність перевірочного ресурсу. Для DNS-01 потрібна можливість керування відповідними DNS-записами. Вибраний спосіб повинен відповідати характеристикам серверного середовища.

Після підтвердження контролю над доменом виконується отримання TLS-сертифіката. Закритий криптографічний ключ необхідно зберігати в захищеному каталозі з обмеженими правами доступу. Після отримання сертифіката перевіряються його чинність, відповідність доменним іменам і правильність зв'язку із закритим ключем. Лише після цього сертифікат допускається до використання у вебсервері.

Налаштування HTTPS передбачає визначення мережевої точки приймання запитів, допустимих версій TLS та параметрів сертифікації. Для нових конфігурацій доцільно надавати перевагу TLS 1.3. Можливість використання інших версій визначається вимогами сумісності та політикою безпеки. Застарілі й заборонені криптографічні алгоритми не повинні активуватися лише для забезпечення підключення несумісних клієнтів.

Перед застосуванням конфігурації виконується її синтаксична перевірка. Якщо виявлено помилки, нові налаштування не повинні замінювати працездатну конфігурацію. Після успішної перевірки здійснюється контрольоване

застосування параметрів вебсервера. Завершення цього етапу підтверджується тестовим TLS-підключенням із перевіркою сертифіката.

Окремо налаштовується автоматичне перенаправлення HTTP-запитів на HTTPS. При цьому необхідно враховувати, що початковий HTTP-запит передається без шифрування. Для зменшення відповідного ризику може застосовуватися HTTP Strict Transport Security (HSTS). Перед активацією політики слід перевірити готовність вебресурсу та його піддоменів до обов'язкового використання HTTPS. Автоматичне повернення до незахищеного HTTP для конфіденційних операцій не допускається.

Для запобігання завантаженню незахищених компонентів сторінок необхідно перевірити адреси зовнішніх ресурсів. За потреби можуть застосовуватися правила Content Security Policy, зокрема `upgrade-insecure-requests`. Проте цей механізм не забезпечує роботу HTTPS на сервері, який не підтримує TLS. Тому правильність завантаження залежних компонентів перевіряється після застосування відповідної політики.

Рекомендована послідовність запуску програмного комплексу передбачає початкову перевірку доступності DNS та вебсервера. Після цього запускається керувальний компонент, який отримує відомості про контрольовані домени. Далі перевіряється стан сертифікатів і наявність необхідних HTTPS-конфігурацій. За потреби запускаються процедури отримання або оновлення сертифікатів. Після завершення налаштування активується періодичний контроль захищеного доступу.

Для автоматичного оновлення сертифікатів визначається періодичність перевірки їхньої чинності. Якщо сертифікат потребує заміни, запускається відповідна процедура ACME. Після отримання нового сертифіката виконується перевірка та контрольоване застосування конфігурації. У разі помилки система повинна зберігати попередню працездатну конфігурацію, якщо вона відповідає чинним вимогам безпеки.

Первинна перевірка працездатності починається зі встановлення HTTPS-з'єднання з контрольованим ресурсом. Необхідно переконатися в успішності

TLS-узгодження, відповідності сертифіката доменному імені та доступності вебсторінки. Додатково перевіряється правильність перенаправлення HTTP-запитів і відсутність незахищених залежних ресурсів.

Для дослідження захисту від пасивного аналізу можуть використовуватися засоби захоплення мережевого трафіку, зокрема Wireshark або tcpdump. У контрольованому середовищі порівнюються HTTP- та HTTPS-сеанси. Перевіряється доступність вмісту запитів, заголовків і відповідей без використання сеансових ключів. Водночас необхідно враховувати, що IP-адреси, обсяги пакетів та часові характеристики залишаються доступними для спостереження.

Окремо перевіряється поведінка системи під час використання недійсного сертифіката, невідповідності доменного імені та недоступності HTTPS. Для ресурсів з обов'язковим шифруванням очікуваною реакцією є блокування незахищеного передавання. Додатково контролюється відсутність автоматичного вимкнення перевірки сертифіката або переходу до застарілих криптографічних параметрів.

Під час експлуатації можливі помилки DNS, недоступність ACME-сервісу та порушення конфігурації вебсервера. Для їх діагностики використовуються службові журнали керувального компонента, повідомлення ACME-клієнта та журнали вебсервера. Після усунення порушення необхідно повторно перевірити доступність HTTPS і відповідність сертифіката встановленим вимогам.

Перед введенням системи в регулярну експлуатацію перевіряється відновлення її роботи після перезапуску, збереження налаштувань та працездатність автоматичного оновлення сертифікатів. Додатково контролюються повноваження на зміну конфігурації та захист закритих ключів. Результати цих перевірок використовуються для оцінювання готовності програмного комплексу до практичного застосування.

Відомості про розробника програмного забезпечення необхідні для ідентифікації програмного продукту та його подальшого технічного

супроводження. Вони дозволяють установити зв'язок між створеною системою захисту вебресурсів і відповідною магістерською кваліфікаційною роботою.

На рисунку 5.2 зображено сформовану форму авторських даних розробленого програмного забезпечення.

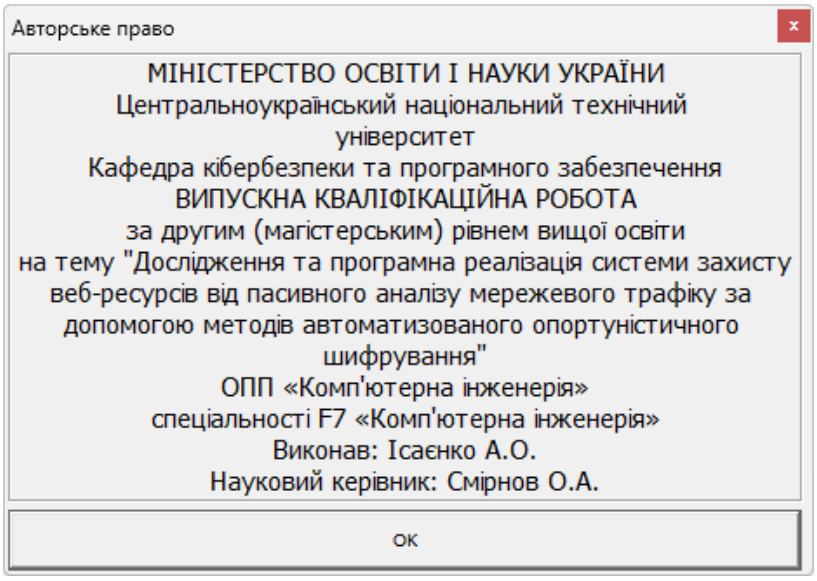


Рисунок 5.2 – Форма авторських даних

# 6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Метою розробки* є дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Об'єктом дослідження* є процес захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Предметом дослідження* є методи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

*Методи дослідження* базуються на методах теорії побудови комп'ютерних мереж, теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

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

– Удосконалено метод захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– Розроблено вітчизняний продукт захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування, який має більш широкі можливості, на відміну від існуючих аналогів.

# 7 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

## 7.1 Вступ

Сучасний розвиток технічного та технологічного стану виробництва передбачає постійну автоматизацію та оптимізацію виробничих процесів. Комп’ютер – невід’ємна складова сучасного життя. За допомогою обчислювальної техніки вирішують складні робочі задачі, ведуться наукові дослідження, створюються архітектурні креслення і твори мистецтва. Сьогодні, напевно, важко уявити компанію, господарська діяльність в якій здійснювалась би без використання комп’ютерної техніки.

Незважаючи на видиму безпеку та розвиток сучасних технологій, при роботі за комп'ютером є ряд чинників, які можуть вплинути на здоров'я людини. Через масовий характер робіт, що виконуються працівниками за допомогою комп’ютера, законодавством України чітко врегульовані норми та вимоги до використання комп’ютерної техніки на підприємстві й охорона праці при роботі за комп’ютером.

Законом України “Про охорону праці” [1] регламентуються загальні положення державної політики в галузі охорони праці, які конкретизуються нормативно-правовими актами про охорону праці, зокрема Вимогами щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями, затверджені наказом Мінсоцполітики від 14.02.2018 р. № 207, зареєстровані в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 [2].

Робота з комп'ютером характеризується значною розумовою напругою і нервово-емоційним навантаженням операторів, високою напруженістю зорової роботи і достатньо великим навантаженням на м'язи рук при роботі з клавіатурою ЕОМ.

У даному розділі магістерської роботи висвітлюються основні питання охорони праці працівників, робота яких пов’язана з роботою за комп’ютером, планування робочого приміщення, де працюють користувачі ПК; параметри мікроклімату, освітленість робочих місць та виробничих приміщень; шумові завади.

Правильна організація і раціональне устаткування робочого місця можливість ефективно і з якнайменшими витратами праці виконувати свої функції, плідно спілкуватися співробітниками і підлеглими, підтримувати високу працездатність і робочий настрій.

Велике значення має раціональна конструкція і розташовує елементів робочого місця, що важливе для підтримки оптимальної робочої пози людини-оператора, а також необхідно дотримувати правильний режим праці і відпочинку.

Що стосується питання охорони праці людини необхідно вирішувати на всіх стадіях трудового процесу незалежно від виду професійної діяльності.

Забезпечення безпечних і здорових умов праці в значній мірі залежить від правильної оцінки небезпечних, шкідливих виробничих факторів. Однакові по складності зміни в організмі людини можуть бути викликані різними причинами. Це можуть бути фактори виробничого середовища, надмірне фізичне і розумове навантаження, нервово-емоційна напруга, а також різне сполучення цих причин.

Робота працівників пов’язана з роботою за комп’ютером, тому актуальною є розгляд саме умов праці та стану охорони праці працівників які постійно працюють з комп’ютерною технікою.

Завдання даного розділу полягає у тому, щоб розробити якісний програмний продукт необхідно організувати безпеку на робочому місці програміста. Під час проектування безпеки робочому місці з ПК необхідно домагатися високої якості та надійності технічного забезпечення, але й створювати комфортні параметри довкілля для розробників.

## 7.2 Характеристика умов праці програміста

Приміщення в якому проводиться розробка і дослідження програмного продукту за завданням наведені в таблиці 7.1

Таблиця 7.1 – Характеристика умов праці

| Шкідливі та небезпечні фактори на робочому місці | Джерела утворення небезпек | Примітка (дані наведені для приміщення) |
|---|---|---|
| Електрична напруга вище 127В;<br>Шум;<br>Електромагнітні випромінювання;<br>Статична електрика;<br>Іонізація повітря;<br>Пожежна небезпека у приміщені;<br>Неякісне освітлення; | Кондиціонер<br>6 ПЕОП<br>Принтер<br><br><br><br>Папір<br>Світильники | Розміри приміщення (м)<br>Довжина – 9<br>Ширина – 6<br>Висота – 3<br>Кількість працюючих – 7 |

Згідно Державними санітарними правилами і нормами ДСанПіН 3.3.2.007-98 [3] може перебувати 7 працівників. Мінімальна площа приміщення на 1 людину повинна складати не менше 6 м$^2$. Висновок – за умовами завдання це виконується.

В приміщенні відсутні умови, які можуть створювати підвищену або особливо підвищену небезпеку, тому воно відноситься до класу звичайних приміщень згідно ПУЕ. Джерелом живлення є трифазна мережа напруги 380/220 В з глухо заземленою нейтралі, з частотою 50 Гц згідно За пожежо-вибухонебезпекою приміщення відноситься до класу В. В таблиці 7.2 наведена загальна характеристика приміщення щодо вибухо- пожежонебезпеки та важкістю робіт.

Температура повітря в приміщенні визначається температурою зовнішнього повітря і тепловою енергією, що виділяється всередині приміщення. Джерелами теплоти в даному приміщенні є люди, електроустаткування, а також освітлювальні прилади в темний час доби. Зовнішнім джерелом надлишкового тепла є сонячна радіація у світлий час доби. Робота, виконувана в даному приміщенні, відноситься до категорії І-а. Людиною в цьому випадку виділяється до 120 ккал теплової енергії в годину. Вологість

повітря в приміщенні визначається вологістю атмосферного і видихуваного людьми повітря, а також випарами з поверхні шкіри.

Таблиця 7.2 – Загальна характеристика приміщення щодо вибухопожежонебезпеки та важкістю робіт

| Характеристика приміщень за вибухопожежною категорією та класом зони | Загальна характеристика приміщення | Категорія за важкістю робіт згідно ГН 3.3.5-7.6.6.1 -2002 |
|---|---|---|
| В – пожежонебезпечне клас П – II | Звичайне без ознак хімічного забруднення та нормальної вологості і за санітарними нормами | 1а - до 139 Вт/м$^2$<br>1б - до 140-174 Вт/м$^2$<br>Клас умов праці – оптимальний |

У таблиці 7.3 приведені оптимальні значення параметрів мікроклімату для категорії ваги робіт І-а, а також фактичні значення цих параметрів у розглянутому приміщенні. У приміщеннях з використанням обчислювальної техніки рекомендується застосування тільки оптимальних значень показників мікроклімату, тобто таких, при яких людина почуває себе комфортно.

Таблиця 7.3 – Оптимальні і фактичні значення параметрів мікроклімату

| Пора року | Оптимальні для Іа | | | Фактичні | | |
|---|---|---|---|---|---|---|
| | Температура °С | Воло-гість,% | Швидкість повітря, м/ | Температура °С | Воло-гість% | Швидкість повітря, м/с |
| Холодна | 22-24 | 40-60 | 0,1 | 22-24 | 40-55 | 0,12 |
| Тепла | 23-25 | 50-70 | 0,1 | 24-25 | 50-65 | 0,9 |

Таким чином, показники мікроклімату в приміщенні, загалом, відповідають установленим нормам. В холодний період року використовується індивідуальне опалення, завдяки якому витримується температурний режим в приміщенні в залежності від температури повітря навколишнього середовища.

Для підтримки температури в літню пору встановлений кондиціонер, який має достатню потужність охолодження, а для підтримки вологості є зволожувач повітря.

Джерелами запиленості повітря в приміщенні є одяг людей і пил, що проникає з вулиці. З метою боротьби з пилом здійснюються регулярні вологі прибирання і провітрювання.

У приміщенні немає виділення шкідливих газів. Тому що в ньому не проводиться монтажних робіт, пайки чи інших робіт, при яких виділяються шкідливі гази.

Для нормалізації параметрів повітряного середовища також періодично здійснюється провітрювання приміщення і вологе прибирання. У всьому будинку діє встановлена загально обмінна витяжна вентиляція.

Особливістю роботи за дисплеєм ЕОМ є постійна й значна напруга функцій зорового аналізатора, обумовленого необхідністю розходження самосвітних об'єктів (символів, знаків тощо) при наявності відблисків на екрані, рядковій структурі екрана, мерехтіння зображення, недостатньою чіткістю об'єктів розходження. По характеру зорової роботи, робота відноситься до високої точності, розряд зорової роботи III, підрозряд Г.

Раціональне освітлення приміщення сприяє кращому виконанню виробничого завдання і забезпеченню комфорту при роботі. Для забезпечення нормального освітлення застосовуються природне, однобічне, бічне і штучне освітлення, а також комбіноване, нормуються зігдно ДБН В.2.5-28-2006 Природне і штучне освітлення [4]. Дані по нормах освітлення наведені в таблиці 7.4.

Таблиця 7.4 – Норми освітлення

| Мінімальний розмір об’єкта розрізнювання, мм | Фон | Контрас | Розряд, під розряд зорової роботи | Нормоване значення | | |
|---|---|---|---|---|---|---|
| | | | | Природне освітлення КПО,% | Штучне освітлення | |
| | | | | | $E_{мін.}$ лк | Тип ламп |
| Від 0,3 до 0,5 | Світлий | Середній | IIIг | 1,5 | 300 | Газоро зрядні |

За результатами виміру освітленості величина освітленості від системи загального штучного освітлення знаходиться у межах 320 лк, що відповідає вимогам, які пред'являються до даного приміщення.

Основними джерелами шуму на робочих місцях , обладнаних відео дисплейними терміналами, є принтер, сканер факс і обладнання для кондиціювання повітря, в самих відео дисплейних терміналів – вентилятори систем охолодження і трансформатори.

Згідно ДСанПіН 3.3.2.007-98 [3] допустимий еквівалентний рівень шуму для робочого місця програміста складає 50 дБА, Допустимі параметри рівнів звуку та звукового тиску представлені в таблиці 7.5.

Таблиця 7.2 – Рівні звукового тиску від різних джерел.

| Джерело шуму | Рівень шуму, дБА |
|---|---|
| Жорсткий диск | 45 |
| Вентилятор | 45 |
| Принтер | 55 |
| Сканер | 50 |

### 7.3 Розробка заходів з умов поліпшення охорони праці

У комплексі заходів, що сприяють удосконаленню організації праці програміста, збереженню здоров'я і підвищення працездатності, його велике значення має організація робочих місць.

Проводячи аналіз умов праці в розглянутому приміщенні, ми одержали наступні результати:

– обсяг приміщення, що приходиться на одному працюючого, відповідає нормативному значенню;

– показники мікроклімату відповідають нормативному значенню;

– акустичні умови роботи в нормі.

Важлива роль в ефективному забезпеченні праці належить моральному мікроклімату. Відносини працівників повинні ґрунтуватися на об'єктивності, доброзичливості, взаємодопомозі, глибокій повазі до кожного члена колективу, турботі про молодих співробітників.

Для забезпечення найбільш ефективного виконання обов'язків, плануючи розпорядок, слід дотримуватися таких принципів:

– для зняття втоми через кожні 1,5-2 год. робити перерви для відпочинку на 5-10 хв.;

– для усунення монотонності виконуваних робіт чергувати характер праці.

За умови неправильної організації праці та відпочинку, втома може нагромаджуватися щоденно й переходити в перевтому або захворювання. У зв'язку з цим режим праці та відпочинку користувачів ПК необхідно будувати з урахуванням працездатності, яка змінюється протягом доби.

Виходячи із наступного можна зробити висновок, що основною причиною втомлюваності та зниження працездатності працівника, який постійно працює за комп’ютером є психофізіологічний фактор, тому основною пропозицією правильна організація робочого місця з урахуванням ергономічних вимог, а також дотримання регламентованого режиму праці та відпочинку.

### 7.4 Розрахункова частина

Розглянемо приклад розрахунку ризику травмування людей, зайнятих певним видом діяльності (в розрахунку за рік), якщо середньорічна кількість осіб, що займаються цією діяльністю – 600 , а за останні 4 років травми одержали 9 осіб. Порівняти обчислений рівень ризику з нормованим (прийнятим на сьогоднішній день) у світовій практиці.

1. Визначаємо середньорічну кількість травмованих осіб, для цього ділимо кількість постраждалих на кількість років, за які сталися ці трагічні події:

$$N = К/Т \quad (7.1)$$

де К – кількість постраждалих; Т – період часу; підставляємо значення і отримуємо:

$$N = 9 / 4 = 2.25 \text{ осіб на рік.}$$

2. Знаходимо величину ризику за формулою

$$R = n / N , \quad (7.2)$$

де n – кількість подій, які відбулись з небажаними наслідками, N – загальна кількість подій, що може відбутися.

$$R = 2.25 / 600 = 0,00375 = 3,75 \cdot 10^{-3} .$$

Далі порівняємо отриманий результат з величиною прийнятого ризику, який визначений у світовій практиці і дорівнює $1 \cdot 10^{-6}$.

Для цього отриманий результат ділимо на прийнятий ризик:

$$П = 0,00375 / 0,000001 = 3750 \text{ разів.}$$

Таким чином, величина ризику становить $3.75 \cdot 10^{-3}$, а ризик травмування людей більший за прийнятий у 3750 разів. Крім того, можна розрахувати коефіцієнт частоти виникнення небезпечних ситуацій на 1000 чоловік, а для цього необхідно:

$$К = R \cdot 1000 \quad (7.3)$$

$$К = 0.00375 \cdot 1000 = 3.75.$$

Таким чином, коефіцієнт частоти дорівнює 3,75 людини на 1000 осіб.

## 7.5 Висновки до розділу

У даному розділі магістерської роботи розглянуті умови праці програміста, проведено аналіз санітарно-гігієнічних умов праці працівників, які зайняті роботою з комп’ютерною технікою. Наведено приклад розрахунку ризику травмування людей, зайнятих певним видом діяльності. Можна зробити наступний висновок, що шкідливі та небезпечні виробничі фактори існують практично на будь якому робочому місці. Тільки повна усвідомленість працівника про можливі небезпеки, що можуть підстерігати його на робочому місці та дотримання вимог нормативних актів о питань охорони праці та відповідних рекомендацій фахівців, дозволять значною мірою знизити негативний вплив шкідливих та небезпечних факторів при роботі з комп’ютером на організм людини.

# 8 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

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

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– Досліджена система захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

– На основі отриманих результатів досліджень створена програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності

предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Builder C++. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

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

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

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. Ісаєнко А.О. Дослідження та програмна реалізація системи захисту веб-ресурсів від пасивного аналізу мережевого трафіку за допомогою методів автоматизованого опортуністичного шифрування // Збірник праць молодих науковців ЦНТУ. – Вип. 17. – Кропивницький: ЦНТУ, 2027.

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

3. Odnoochko, D., Ilyenko, A., Ilyenko, S., Smirnov, O., Hulak, H. «Marker-based crypto-steganographic system for secure data transmission over open channels». *CEUR Workshop Proceedings*, 2026, 4278, pp. 102-117.

4. Богаченко, С., Смірнов, О., Буравченко, К., Смірнова, Т., Конопліцька-Слободенюк, О., Якименко, Н., Смірнов, С. «Обмеження ресурсів мобільних та edge-платформ при розробці програмного забезпечення для реалізації та оптимізації моделей машинного навчання». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2026, т. 2 № 34, С. 677-686.

5. Вінтенко Б.Ю., Смірнова Т.В., Миронець І.В., Смірнов О.А., Буравченко К.О. «Метод оцінки функціональної стійкості комп'ютерно-орієнтованих процедур системи підтримки оперативного персоналу АЕС». *Системи управління, навігації та зв'язку*, 2026, вип. 2(84), С. 62-69.

6. Смірнова Т.В., Усік П.С., Лисенко І.А., Буравченко К.О., Смірнов О.А. «Методологія підтримки технологічних процесів у критичній інфраструктурі з забезпеченням безпеки інформації на основі хмарних технологій». *Безпека інформації.* 2025. Том 31 № 1. С. 49-60.

7. Смірнова Т.В., Буравченко К.О., Добринчук О.А., Смірнов С.А., Якименко Н.М., Смірнов О.А. «Хмарна технологія моніторингу ключових індикаторів ефективності технологічних процесів у критичній інфраструктурі».

*Центральноукраїнський науковий вісник. Технічні науки.* 2025. № 12(43), ч. II. С. 36-54.

8. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *Кібербезпека: освіта, наука, техніка*. 2025. Том 2 № 30. С. 413-427.

9. Vintenko, B., Myronets, I., Smirnov, O., Smirnova, T. «Enhancing the Reliability of Input Data for Nuclear Power Plant Operator Support Systems». *International Conference on Sensors, Devices and Systems. SDaS 2025. Lecture Notes in Electrical Engineering,* 2026. vol 1617. pp 3-12/

10. Kuznetsov O., Frontoni E., Arnesano M., Smirnov O., Khruskov B. «Biometric Authentication in TinyML: Opportunities and Challenges». *Tiny Machine Learning: Design Principles and Applications,* 2026, pp. 587-633.

11. Kuznetsov O., Frontoni E., Kuznetsova Y., Chevardin V., Smirnov O. «Architectural foundations for adaptive security in edge computing systems». *Cybersecurity Defensive Walls in Edge Computing,* 2025. pp. 21-61.

12. Chulinda L., Smirnov O., Shapenko L., Ustynova I., Bohatiuk I., Kelyp S. «The role of innovation in ensuring the safety of international civil aviation». *Ceur Workshop Proceedings,* 2025, 4024, pp. 530–542.

13. Петрик В.М., Присяжнюк М.М., Аль-Файюмі Халед та ін. «Системи інформаційної зброї та технології інформаційної війни»: підручник / Петрик В.М., Присяжнюк М.М., Аль-Файюмі Халед, Жарков Я.М., Смірнов О.А, Буравченко К.О., Давидюк А.В., Кононович В.Г., Корчинский В.В., Кудирко В.М., Фесенко А.О.; за заг. ред. В.М. Петрика, М.М. Присяжнюка.– К.: Видавничий центр “Кафедра”, 2025.– 320 с.

14. Усік, П.С., Смірнова, Т.В., Буравченко, К.О., Смірнов, О.А., Улічев, О.С., Смірнов, С.А. «Дослідження технологій забезпечення кібербезпеки банківських систем з використанням штучного інтелекту». *Кібербезпека: освіта, наука, техніка*. 2025. Том 1 № 29. С.704–716, 2025

15. Kuznetsov, O., Frontoni, E., Kuznetsova, K., Arnesano, M., Smirnov, O. «A secure biometric authentication architecture for blockchain-driven cyber-physical systems». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 193–224.

16. Kuznetsov, O., Atzeni, G., Arnesano, M., Randieri, C., Smirnov, O. «Secure IoT-based smart wheelchair system: From implementation to security enhancement strategy». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 225–257.

17. Kuznetsov, O., Smirnov, O., Kuznetsova, T., Shaikhanova, A., Svatowsky, I. «Privacy-utility trade-offs in IoT networks: A comparative analysis of differential privacy mechanisms for sensor data aggregation». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 589–622.

18. Lakhno, V., Malyukov, V., Smirnov, O., Bebeshko, B., Chubaievskiy, V., Zhumadilova, M., Malyukova, I., Smirnov, S. «Multifactorial Model for Targeted Attacks Counteracting Within the Framework of a Multi-Step Quality Game with Fuzzy Information». *8th International Symposium on Intelligent Informatics, ISI 2023*, 2025. vol 389. pp 377-389. Springer, Singapore.

19. Kuznetsov O., Frontoni E., Kuznetsova Y., Smirnov O., Moskovchenko I. «Trust-Based Security Architecture for Edge Computing: A Simulation Study of Dynamic Trust Evolution and Attack Detection». *CEUR Workshop Proceedings*, 2024, 3909, pp. 227–241.

20. Kuznetsov, O., Frontoni, E., Kryvinska, N., Chevardin, V., Smirnov, O. «Wireless Network Encryption Stream Ciphers, Computational Modeling, and Security Analysis». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 379–402.

21. Kuznetsov, O., Frontoni, E., Kryvinska, N., Smirnov, O., Imoize, G.L. «Computational Modeling of Enhanced Spread Spectrum Codes for Asynchronous Wireless Communication». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 403–447.

22. Ткаченко, О., Ільєнко, А., Улічев, О., Мелешко, Є., Смірнов, О. «Правові засади поширення інформаційних впливів в соціальних мережах». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»,* 2024. № 2(26), С. 170–188.

23. Смірнова Т.В., Конопліцька-Слободенюк О.К., Буравченко К.О., Смірнов С.А., Кравчук О.В., Козірова Н.Л., Смірнов О.А. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка.* 2024. №4(24), С. 6-27.

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

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

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

27. Akhalaia, G., Iavich, M., Iashvili, G., Prysiazhnyy, D., Smirnova, T. «Secure Encrypted Connection on Georgian Website». *CEUR Workshop Proceedings*, 2023, 3550, pp. 313-320.

28. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56

29. 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.

30. 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.

31. 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.

32. Смірнова Т.В., Гнатюк С.О., Бердибаєв Р.Ш., Сидоренко В.М., Жигаревич О.К., «Система корелювання подій та управління інцидентами кібербезпеки на об'єктах критичної інфраструктури». *Кібербезпека: освіта, наука, техніка*, №3(19), 2023, С. 176-196.

33. Смірнов О.А. Козлов Я.О., Смірнова Т.В. «Дослідження застосування SIEM-систем для забезпечення кібербезпеки та захисту інформації». *ІІ 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.

34. 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.

35. 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.

36. 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.

37. 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,

38. 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.

39. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

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., 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

44. 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.

45. 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.

46. 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.*

47. 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.

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

49. 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.

50. 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.

51. 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.

52. 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.

53. 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.