Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARC

Виконав: студента 4 курсу, гр.1КІТС-21б
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем
(шифр і назва напряму підготовки, спеціальності)

Дяченко О.О
(прізвище та ініціали)

Керівник: к.т.н., доц., доцент каф. МБІС
Грицак А.А
(прізвище та ініціали)
«____» _____ 2025 р.

Рецензент: ст. викладач кафедри ОТ
Колесник І.С.
(прізвище та ініціали)
«____» _____ 2025 р.

Допущено до захисту
Голова секції УБ кафедри МБІС
д.т.н., проф. Яремчук Ю.С.
"____" _____ 2025р.

Вінниця ВНТУ – 2025

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти І-й (бакалаврський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

д.т.н., проф. Яремчук Ю.Є.
" 20 " березня 2025 р.

З А В Д А Н Н Я
на бакалаврську кваліфікаційну роботу студенту
Дяченко О.О.
(прізвище, ім'я, по-батькові)

1. Тема роботи Підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARC
Керівник роботи Грицак А.А. к.т.н, доц.кафедри МБІС
(прізвище, ім'я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " 20 " березня 2025року № 97
2. Термін подання студентом роботи за тиждень до захисту
3. Вихідні дані до роботи Електронні джерела, підручники та наукові статті по темі. Існуюче програмне забезпечення, яке стосується теми бакалаврської дипломної роботи.
4. Зміст текстової частини: Зміст текстової частини роботи охоплює аналіз фішингових загроз у сфері електронної пошти, дослідження вразливостей існуючих систем електронної комунікації та огляд сучасних механізмів автентифікації повідомлень — SPF, DKIM та DMARC. У роботі проведено порівняльний аналіз ефективності кожного з методів, запропоновано вдосконалену модель політик автентифікації для підвищення рівня захищеності поштових систем. Реалізовано програмний модуль перевірки вхідної пошти з журналюванням результатів перевірки. Виконано тестування функціональності, точності виявлення фішингових повідомлень та впливу оновлених політик на доставку легітимної пошти.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)

агальна схема обміну електронною поштою з автентифікацією SPF, DKIM, DMARC Структурна схема перевірки заголовків листа на сервері одержувача Порівняльна таблиця фішингових атак із та без автентифікаціїСхема взаємодії поштового клієнта, DNS-сервера та політик автентифікації Приклад правильно сформованих записів SPF, DKIM та DMARC у DNS

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	Грицак А.А. к.т.н., доц., доцент		
Розділ II	Грицак А.А. к.т.н., доц., доцент		
Розділ III	Грицак А.А. к.т.н., доц., доцент		

7.Дата видачі завдання 20 березня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	23.03.2025	30.03.2025	Виконано
2	Аналіз методів розв'язання задачі	1.04.2025	10.04.2025	Виконано
3	Постановка задач розробки програмного модулю	11.04.2025	20.04.2025	Виконано
4	Дослідження інформаційних джерел	21.04.2025	10.05.2025	Виконано
5	Аналіз варіантів роботи модуля	11.05.2025	15.05.2025	Виконано
6	Розробка алгоритму роботи	16.05.2025	20.05.2025	Виконано
7	Реалізація програмного модулю	21.05.2025	24.05.2025	Виконано
8	Тестування програмного модулю	24.05.2025	26.05.2025	Виконано
9	Оформлення матеріалів до захисту БДР	26.05.2025	31.05.2025	Виконано

Студент _____ Дяченко О.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Грицак А.А
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дана бакалаврська дипломна робота присвячена підвищенню захищеності електронної пошти від фішингових листів шляхом вдосконалення політик автентифікації на основі протоколів SPF, DKIM та DMARC. Метою дослідження є зниження ризику фішингових атак завдяки впровадженню ефективних механізмів перевірки достовірності відправника та контролю за поштовим трафіком. У роботі проаналізовано існуючі вектори атак, недоліки чинних протоколів автентифікації, а також розроблено програмний модуль, що здійснює перевірку поштових повідомлень та керування політиками безпеки. Для реалізації рішення використано сучасні технології та мови програмування.

Результати дослідження сприятимуть підвищенню кіберстійкості інформаційних систем, забезпеченню достовірності електронних листів та захисту користувачів від фішингових загроз.

Ключові слова: фішинг, електронна пошта, SPF, DKIM, DMARC, автентифікація, захист.

ABSTRACT

This bachelor’s thesis is devoted to enhancing the security of email communication against phishing attacks by improving authentication policies based on SPF, DKIM, and DMARC protocols. The aim of the research is to reduce the risk of phishing by implementing effective mechanisms for sender verification and email traffic control. The work analyzes current attack vectors and weaknesses of existing authentication protocols and presents the development of a software module that verifies incoming email messages and manages security policies. Modern technologies and programming tools were used to implement the proposed solution.

The results of this research contribute to increasing the cybersecurity resilience of information systems, ensuring the authenticity of email messages, and protecting users from phishing threats.

Keywords: phishing, email security, SPF, DKIM, DMARC, authentication, protection.

ЗМІСТ

АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ ДИПЛОМНОЇ РОБОТИ 8

1.1 Основні поняття електронної пошти, фішингу та методів атак..... 10

1.2 Класифікація фішингових атак та особливості їх виявлення..... 16

1.3 Протоколи автентифікації електронної пошти: загальний огляд SPF, DKIM, DMARC..... 26

1.4 Аналіз недоліків та вразливостей чинних механізмів автентифікації 34

1.5 Висновок до розділу 1 та постановка задачі 40

АЛГОРИТМ ВІДСЛІДКОВУВАННЯ НЕСАНКЦІОНОВАНИХ ДІЙ КОРИСТУВАЧІВ 42

2.1 Особливості формування політик SPF, DKIM та DMARC 42

2.2 Розробка комбінованої стратегії автентифікації з підвищеним рівнем захисту 44

2.3 Розробка алгоритму обробки поштових повідомлень з перевіркою SPF, DKIM та DMARC 46

2.4 Висновок до розділу 2 50

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО МОДУЛЮ 52

3.1 Обґрунтування вибору мови програмування та середовища розробки програмного модулю 52

3.2 Реалзація програмного модуля керування SPF, DKIM та DMARC. 56

3.3 Реалзація програмного модуля захисту 61

ВИСНОВКИ..... 68

ПЕРЕЛІК ПОСИЛАНЬ..... 69

Додаток А. Технічне завдання..... 75

Додаток Б. Лістинг програмного коду 77

Додаток В. Ілюстраційний матеріал..... 80

Додаток Г. Протокол перевірки на антиплагіат..... 86

ВСТУП

Актуальність. У сучасному цифровому середовищі електронна пошта залишається одним з основних каналів комунікації в особистому, корпоративному та урядовому секторах. Проте саме цей канал найчастіше стає об'єктом фішингових атак, що базуються на методах соціальної інженерії, спуфінгу, підміни адреси відправника, маскування гіперпосилань та впровадження шкідливих вкладень. Такі атаки можуть призвести до витоку конфіденційної інформації, компрометації облікових даних, фінансових втрат та зниження рівня довіри до організації.

У цьому контексті актуальним є створення ефективних механізмів автентифікації повідомлень, які б дозволяли виявляти та блокувати фішингові листи ще на етапі доставки. Використання протоколів SPF, DKIM та DMARC забезпечує перевірку джерела повідомлення, його цілісності та відповідності заявленому домену, проте окреме застосування цих протоколів має низку обмежень. Розробка програмного модуля, що реалізує вдосконалену стратегію перевірки з урахуванням всіх трьох механізмів, дозволить значно підвищити безпеку поштових систем.

Метою роботи є підвищення захищеності електронної пошти від фішингових повідомлень шляхом розробки програмного модуля автентифікації з використанням протоколів SPF, DKIM та DMARC у комплексі.

Для досягнення поставленої мети необхідно виконати наступні **завдання**:

–проаналізувати літературні джерела та статистику фішингових інцидентів;

–дослідити принципи роботи протоколів SPF, DKIM, DMARC, а також їх технічні особливості;

–виявити недоліки та вразливості існуючих механізмів автентифікації;

–розробити алгоритм обробки вхідних листів з перевіркою автентичності;

–реалізувати програмний модуль, що виконує перевірку повідомлень згідно з заданими політиками;

–протестувати працездатність модуля та оцінити його ефективність у виявленні фішингових повідомлень.

Об’єктом дослідження є методи автентифікації електронної пошти та захисту від фішингу.

Предметом дослідження є процес реалізації програмного модуля, що здійснює перевірку автентичності поштових повідомлень на основі SPF, DKIM та DMARC.

Наукова новизна роботи полягає у створенні інтегрованого рішення для захисту електронної пошти, яке поєднує три рівні автентифікації з адаптивною політикою обробки повідомлень, що не проходять перевірку.

Практична цінність полягає у можливості впровадження розробленого модуля в корпоративне середовище для забезпечення фільтрації небажаних листів, зменшення кількості фішингових атак та підвищення загального рівня інформаційної безпеки.

АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ ДИПЛОМНОЇ РОБОТИ

У цьому розділі буде проведено аналіз основних загроз для електронної пошти, зокрема фішингових атак, які становлять одну з найпоширеніших форм соціальної інженерії. Розглянуто принципи роботи та обмеження сучасних механізмів автентифікації електронної пошти — SPF, DKIM і DMARC. Проаналізовано типові вектори обходу цих механізмів, а також виявлено недоліки в їх конфігурації та впровадженні. Особливу увагу приділено розгляду існуючих рішень та політик безпеки у корпоративному середовищі.

На основі проведеного аналізу сформовано висновки та визначено завдання дипломної роботи щодо вдосконалення політик автентифікації для підвищення захищеності від фішингових листів.

1.1 Основні поняття електронної пошти, фішингу та методів атак

Електронна пошта (e-mail) є одним з основних способів цифрової комунікації, що використовується для передавання текстових повідомлень, файлів, гіперпосилань та інших даних між користувачами мережі. За даними статистики компанії Statista, щодня у світі надсилається понад 347 мільярдів електронних листів (дані за 2023 рік), що демонструє масштаб і критичну важливість цього каналу зв’язку.

Для реалізації процесу надсилання та отримання електронних листів використовуються спеціалізовані протоколи прикладного рівня моделі OSI. Кожен з них має власне призначення, переваги та технічні параметри. У таблиці 1.1 наведено основні характеристики поштових протоколів, які використовуються в більшості систем електронної пошти. [1]

Таблиця 1.1 – Основні поштові протоколи та їх технічні характеристики

Протокол	Призначення	Порт (типово)	Захищена версія
SMTP	Надсилання листів	25 / 587	SMTPS (465)
POP3	Отримання листів з видаленням	110	POP3S (995)
IMAP	Отримання листів зберігаючи копію	143	IMAPS (993)

Як видно з таблиці, кожен протокол має як стандартну, так і захищену реалізацію з використанням TLS/SSL. Найбільш вразливими є SMTP-сесії без шифрування, особливо при передачі між серверами. Саме через це електронна пошта без впровадження додаткових механізмів автентифікації та шифрування залишається мішенню для фішингових атак та перехоплення трафіку.

Сама система електронної пошти складається з таких функціональних блоків:

– клієнтський додаток (MUA — Mail User Agent): програма, з якої користувач створює, надсилає та отримує листи;

– поштовий сервер (MSA/ MTA): передає лист через SMTP до серверів одержувача;

– сервер зберігання листів (MDA): приймає вхідні повідомлення і зберігає їх для доступу через POP3 або IMAP.

Щоб краще зрозуміти послідовність функціонування електронної пошти, на рисунку 1.1 подано структурно-функціональну схему, яка ілюструє основні етапи маршруту повідомлення від відправника до одержувача. Схема також відображає роль кожного з компонентів системи у процесі доставки листа.

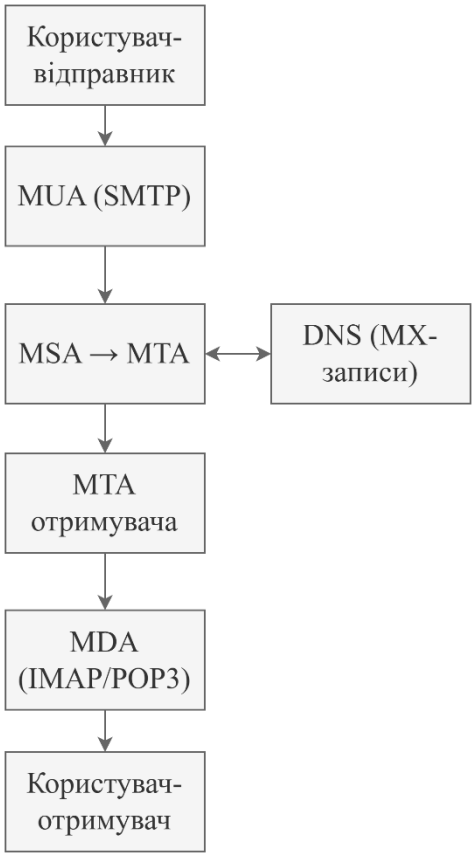


Рисунок 1.1 – Структурно-функціональна схема роботи електронної пошти

Схема демонструє, що лист проходить кілька логічних вузлів, де кожен етап є потенційним місцем атаки — особливо під час обміну між серверами. Саме тому впровадження політик автентифікації (SPF, DKIM, DMARC) є критично важливим для виявлення та блокування фальсифікованих повідомлень на ранніх етапах.[1]

Алгоритм доставки електронного листа:

– Користувач вводить текст повідомлення та натискає «Надіслати»;

– SMTP-протокол пересилає лист до поштового сервера;

– MTA визначає MX-запис домену отримувача через DNS;

– Повідомлення пересилається до MTA сервера-одержувача;

– Лист зберігається на сервері і доступний одержувачу через POP3 або IMAP.

Для аналітичного моделювання можемо представити доставлення листа як скінченну послідовність переходів через вузли мережі. Наприклад, загальна кількість хопів (переходів між серверами) можна описати формулою:

$$N_{hop} = \sum_{i=1}^{n} \delta_i \quad (1.1)$$

де:

N_{hop}— кількість мережевих переходів,

δ_i— затримка або вага кожного переходу,

n— загальна кількість вузлів маршруту.

Недоліки існуючої архітектури електронної пошти:

– Відсутність вбудованої автентифікації: базова специфікація SMTP не перевіряє, чи справді відправник є тим, за кого себе видає.

– Відкритість заголовків: зловмисники можуть підробляти поле From у заголовку листа, імітуючи довіреного відправника.

– Міжсерверна передача не завжди шифрована: хоча існує STARTTLS, воно не є обов'язковим і не гарантує end-to-end шифрування.

– Складність конфігурації: через погано налаштовані SPF або DKIM політики можна легко обійти захист.[2]

Незважаючи на розвинену архітектуру та формальні протоколи електронної пошти, описані раніше, більшість систем електронного листування залишається вразливою до атак, що експлуатують не технологічні, а людські слабкості. Саме цей вектор загроз лежить в основі фішингу — одного з найнебезпечніших інструментів соціальної інженерії у сфері кібербезпеки.

Фішинг — це вид атак, при якому зловмисник імітує надійне джерело (наприклад, банк, поштовий сервіс, внутрішній IT-відділ) з метою викрасти конфіденційну інформацію. На відміну від технічних вразливостей, фішинг не вимагає злому системи. Натомість він використовує довіру, неуважність або стрес користувача, щоб змусити його самостійно ввести пароль, підтвердити операцію або завантажити шкідливий файл.[2]

Основна мета фішингових атак:

– отримання облікових даних користувачів (логінів, паролів);

– викрадення платіжної інформації або кодів двофакторної автентифікації;

– встановлення шкідливого ПЗ для подальшого доступу до систем;

– розширення атаки на інші об’єкти в інфраструктурі (латеральне переміщення в мережі).

Фішингові повідомлення мають низку спільних рис, які дозволяють виділити їх навіть без аналізу вмісту:

– Схожість на легітимні листи (використання логотипів, стилю, підпису реальної організації);

– Підроблена адреса відправника або її маскування через дисплейне ім’я;

– Терміновість або тиск (“Ваш обліковий запис буде заблоковано через 24 години”);

– Заклики до дії (“перейдіть за посиланням”, “введіть свій пароль”, “оновіть дані”).

Наприклад, лист може містити таку фразу:

"Шановний користувачу! У вашому акаунті було зафіксовано підозрілу активність. Для захисту даних підтвердьте особу за цим посиланням..."

При цьому посилання веде на домен, дуже схожий на справжній — наприклад, mail-security-check[.]com замість mail.google.com.[3]

Успіх фішингу значною мірою визначається здатністю викликати довіру до відправника. Зловмисники ретельно підбирають мову, стиль і контекст листа, щоб він виглядав знайомим. У деяких випадках використовується інформація з відкритих джерел (Open Source Intelligence), щоб зробити звернення персоналізованим: вказати справжнє ім’я отримувача, його посаду або навіть назву компанії.

Особливо ефективною є атака у поєднанні з емоційним впливом — наприклад, страхом втрати доступу, фінансової шкоди або публічного викриття. Саме ці елементи соціального впливу значно підвищують ймовірність того, що користувач виконає дії, які йому пропонує зловмисник.

Залежно від методів впливу, технічної реалізації та цільової аудиторії, фішингові атаки можуть мати різні форми: від простих масових розсилок до

цілеспрямованих атак на конкретних осіб або організації. Вони можуть здійснюватися через підміну адреси відправника, використання шкідливих вкладень, QR-кодів, фальшивих веб-сайтів, або навіть через легітимні ланцюжки листування (clone phishing).[3]

Кожен з цих типів атак має свої особливості та рівень загрози. Наприклад, цілеспрямовані атаки на керівників вищої ланки (так званий whaling) можуть завдавати суттєвих фінансових збитків навіть у разі одноразового інциденту. Масові розсилки, навпаки, орієнтовані на кількість, а не якість жертв.

Фішингові атаки залишаються одним з наймасштабніших кіберзагроз у світі, що щороку набуває дедалі витонченіших форм. Попри розвиток технологій кіберзахисту, атаки на електронну пошту з використанням соціальної інженерії демонструють стійке зростання як за кількістю, так і за складністю.

Згідно зі звітом Verizon Data Breach Investigations Report (2023), фішинг є основним вектором проникнення у понад 36% усіх виявлених інцидентів кібербезпеки у корпоративному середовищі. У свою чергу, Proofpoint State of the Phish (2023) зазначає, що 84% організацій у світі зазнали хоча б однієї фішингової атаки протягом року.

Значну роль відіграє економічна шкода. За оцінками FBI IC3 (Internet Crime Complaint Center), у 2022 році збитки лише від атак типу Business Email Compromise (BEC) сягнули 2,7 мільярда доларів лише у США.[4]

Приклади інцидентів:

– 2020, Twitter — фішинг через панель адміністратора дозволив зловмисникам зламати акаунти відомих осіб (Ілон Маск, Барак Обама) та опублікувати шахрайські повідомлення.

– 2021, Colonial Pipeline — фішингове проникнення спричинило зупинку найбільшого нафтопроводу США і викликало паніку на ринку пального.

– 2022, Uber — співробітника змусили надати облікові дані через spear phishing, що призвело до масового витоку внутрішніх даних.

Такі приклади свідчать про недостатню ефективність традиційних засобів захисту, таких як антивірусне ПЗ або прості фільтри за ключовими словами. Це обумовлено двома ключовими причинами:

– Зовнішній вигляд фішингового листа часто неможливо відрізнити від легітимного;

– SMTP-протокол дозволяє надсилати повідомлення від імені будь-якого домену, якщо не впроваджено політики перевірки достовірності (SPF, DKIM, DMARC).

Отже, виникає критична потреба у впровадженні системної перевірки технічного походження повідомлення. Це завдання не може бути покладене лише на користувача або кінцевий поштовий клієнт — воно має вирішуватись на рівні поштових серверів та політик автентифікації.[4]

У наступному підпункті буде сформульовано висновки до підрозділу 1.1 та окреслено основні задачі для подальшого дослідження.

У цьому підрозділі було розглянуто базові принципи функціонування електронної пошти, сутність фішингових атак як прояву соціальної інженерії, їх типові характеристики та масштаби поширення. Аналіз показав, що традиційні методи захисту не здатні ефективно протидіяти фішингу через відсутність вбудованих механізмів перевірки достовірності відправника.

1.2 Класифікація фішингових атак та особливості їх виявлення

Фішингові атаки, засновані на спуфінгу адреси відправника, є одними з найпоширеніших через простоту реалізації та високу ймовірність успіху. Спуфінг (англ. spoofing) — це підміна технічної або візуальної інформації про відправника листа з метою змусити одержувача повірити в автентичність повідомлення.

У випадку електронної пошти, спуфінг полягає в імітації довіреного домену або конкретної адреси, яка зазвичай використовується в межах внутрішньої комунікації або в офіційних каналах. Зловмисник може видавати себе за керівника компанії, технічну підтримку, банківську установу чи навіть державний орган.[5]

Коренем проблеми є базовий протокол передачі пошти SMTP (Simple Mail Transfer Protocol), який спочатку розроблявся без урахування безпекових механізмів. SMTP дозволяє будь-якому клієнту (відправнику) вказати довільну адресу у полі MAIL FROM та From: у заголовку повідомлення. Таким чином, відсутність перевірки справжнього джерела створює ідеальні умови для підміни.

У разі, якщо сервер одержувача не впроваджує додаткові політики автентифікації (наприклад, SPF або DMARC), він просто приймає і доставляє лист незалежно від його походження.[5]

Щоб краще зрозуміти механізм проходження фішингового листа з підміною адреси, розглянемо типову схему (рис. 1.2) маршруту, за яким подібні повідомлення надходять до поштової скриньки жертви. Схема відображає відсутність перевірки автентичності на ключових етапах доставки електронного листа, що дозволяє зловмиснику видавати себе за довірену особу або організацію.

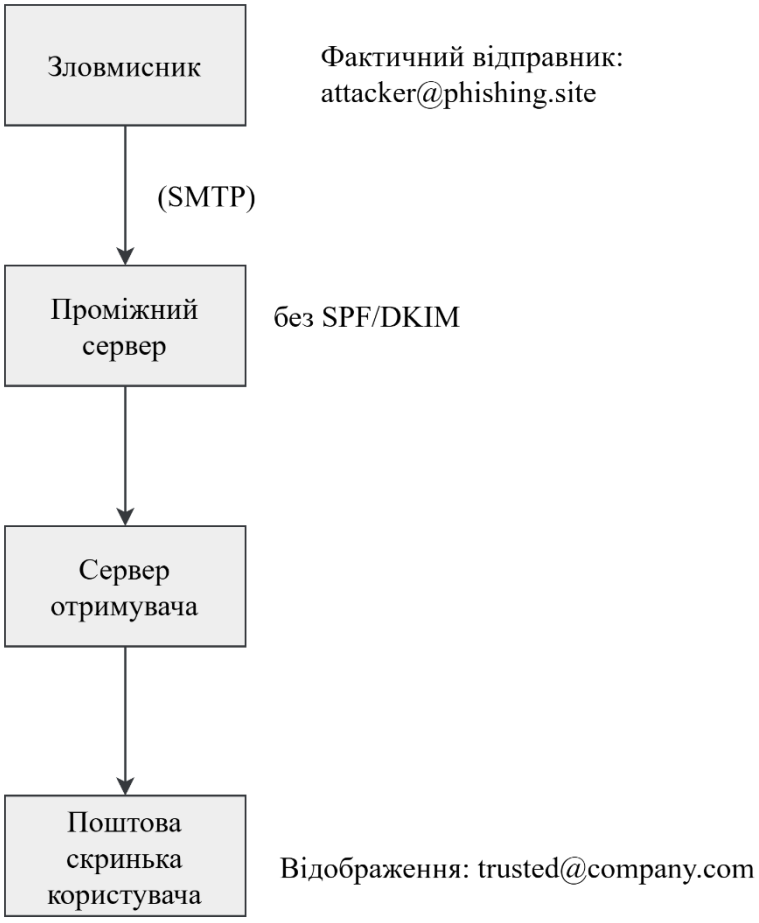


Рисунок 1.2 – Передача фішингового листа з підміною адреси

Схема демонструє критичну вразливість: сервер одержувача не перевіряє, чи дійсно відправник має право надсилати листи від імені вказаного домену. У випадку відсутності політик SPF, DKIM або DMARC сервер не має інструментів для автоматичної ідентифікації підробки, тому спуфінговане повідомлення безперешкодно потрапляє у вхідні. Це підтверджує необхідність впровадження автентифікаційних механізмів як стандарту безпеки для будь-якої організації, що використовує електронну пошту.[6]

Фактори, що сприяють успішності спуфінгу:

– Людський фактор: більшість користувачів не перевіряють детально технічні заголовки пошти, довіряючи лише полю From або імені відправника.

– Низький рівень впровадження SPF/DKIM/DMARC: за даними Cisco (2023), лише близько 70% доменів у глобальному трафіку мають активні SPF-записи, і ще менше — коректно налаштовані політики DMARC.

– Ідентичність візуального оформлення: використання логотипів, шаблонів та підписів оригінальних листів створює додаткову ілюзію легітимності.

Вплив спуфінгу на різні сценарії комунікації особливо помітний у трьох ключових контекстах: внутрішній корпоративній пошті, взаємодії з клієнтами та державному листуванні.

У межах внутрішньої корпоративної пошти спуфінг може призвести до фінансових шахрайств або несанкціонованого доступу до внутрішніх систем. Наприклад, працівник може отримати лист, який виглядає як наказ від керівництва про переказ коштів чи зміну реквізитів партнера.

У випадку комунікації з клієнтами, підміна адреси дозволяє зловмисникам викрасти персональні або платіжні дані, використовуючи авторитет компанії, якій клієнт довіряє. Такі атаки часто націлені на банківські установи, логістичні сервіси або служби підтримки.[6]

Не менш критичним є спуфінг у державному листуванні, де він може бути використаний для поширення дезінформації, впливу на громадську думку або

фішингу особистих даних громадян (наприклад, під виглядом "виборчих комісій", "держслужб" тощо).

У багатьох випадках навіть досвідчені співробітники можуть потрапити в пастку таких повідомлень — особливо, якщо вони оформлені як термінові вказівки або містять запити з фінансовим змістом. Це ще раз підтверджує, що візуальна достовірність у поєднанні з технічною уразливістю створює всі умови для успішної атаки.[7]

Спуфінг — не просто атака на технологію, а атака на довіру. І хоча цей метод здається простим, саме він найчастіше використовується для запуску складніших атак — таких як spear phishing, атаки із вкладеннями або інфікування мережі через користувача. Вирішення проблеми потребує впровадження протоколів перевірки достовірності відправника, які будуть розглянуті в наступних розділах.

Ще одним поширеним прийомом у фішингових атаках є маскування гіперпосилань — техніка, що полягає в підміні або приховуванні реального призначення посилання, аби змусити користувача перейти на підроблений або шкідливий ресурс.

Цей метод широко використовується в текстових повідомленнях, HTML-листах та навіть у PDF-документах, і базується на сприйнятті користувача: людина бачить одну адресу, але насправді натискає на зовсім іншу. Особливо небезпечно це в середовищах, де поштові клієнти приховують реальні URL-адреси або автоматично конвертують їх у клікабельний текст.[7]

Причини ефективності маскування:

–Переважна більшість користувачів не перевіряє адресу вручну, особливо на мобільних пристроях.

–HTML-розмітка дає змогу показувати текст, відмінний від цільової адреси.

–Скорочення посилань через сервіси типу bit.ly, tinyurl тощо ускладнює розпізнавання справжнього домену.

–Сучасні шкідливі посилання можуть бути обгорнуті у вигляд безпечного домену з подальшим редиректом.

Маскування посилань може реалізовуватись різними способами, залежно від технічного рівня атакувальника та захищеності поштової інфраструктури. У таблиці 1.2 нижче узагальнено основні методи, які застосовуються у фішингових кампаніях для приховування реального призначення гіперпосилань, а також зазначено потенційні ризики, що виникають при кожному з них.[8]

Таблиця 1.2 – Поширені методи маскування гіперпосилань та засоби обходу фільтрації

Метод маскування	Суть методу	Потенційна загроза
Текст ≠ URL	Текст посилання виглядає легітимно, але веде на інший сайт	Переходи на шкідливі ресурси без підозри
Скорочення посилань	Застосування сервісів типу bit.ly	Неможливо одразу визначити кінцеву адресу
Динамічні редиректи	Переадресація після відкриття "чистого" сайту	Обхід фільтрів за рахунок затримки переходу
Кодовані URL (Base64, Unicode)	URL у вигляді закодованого рядка	Ускладнює автоматичний аналіз у поштових шлюзах
Маскування в кнопках/зображеннях	Посилання приховане за елементами інтерфейсу	Не видно справжню адресу без перегляду коду

Як видно з таблиці 1.2, атакувальники використовують як базові, так і складні техніки маскування — від простої підміни тексту до кодування URL-адрес і використання сторонніх сервісів для редиректу. Ці методи часто комбінуються зі спуфінгом адреси, що ще більше посилює довіру до повідомлення. Наприклад, лист може виглядати так, ніби його надіслано з support@paypal.com, а кнопка містить

напис “Verify Now”, який веде на сайт pp-support-check.io, візуально схожий на справжній.[8]

Це значно ускладнює виявлення загроз автоматичними системами та вимагає не лише глибокого технічного аналізу повідомлень, але й підвищення обізнаності користувачів, які є останнім бар’єром перед переходом за шкідливим посиланням.

Попри розвиток антифішингових механізмів, багато фільтрів не аналізують вміст JavaScript-редиректів або кодованих URL-адрес на глибокому рівні, особливо якщо посилання динамічно генерується на стороні сервера. Це дозволяє атакувальникам адаптувати маскування під конкретні поштові системи й обійти фільтрацію.

Особливо вразливі системи із застарілим антифішинговим модулем, відсутністю контент-аналізу або мінімальним порогом для trigger-спрацювань. Наприклад, у 2023 році компанія Avanan (Check Point Group) повідомила, що понад 25% фішингових кампаній успішно обійшли стандартні фільтри Microsoft 365, використовуючи саме маскування посилань.[9]

Маскування гіперпосилань — це приклад простого, але надзвичайно ефективного методу соціальної інженерії, що активно використовується в реальних атаках. Його успішність обумовлена як людською довірою до візуальних елементів, так і технічними обмеженнями фільтраційних механізмів. Подальші підрозділи розкриють, як у поєднанні з вкладеннями або персоналізацією такі методи формують комплексну загрозу корпоративній безпеці.

Окрім маскування посилань, ще одним поширеним каналом фішингових атак є вкладення, що містять шкідливий код. У цьому випадку користувачеві пропонується відкрити нібито легітимний файл, наприклад, рахунок, контракт або службовий документ, який насправді активує небезпечний скрипт або вбудований макрос.

Цей тип атаки особливо небезпечний у корпоративному середовищі, де працівники щодня працюють із великою кількістю вкладених документів, що створює передумови для автоматичного або неусвідомленого відкриття заражених файлів.

Шкідливі вкладення можуть мати різні формати, але найбільш часто використовуються:

– Документи з макросами: .docx, .xlsm — використовують вбудовані сценарії VBA (Visual Basic for Applications), які активуються після відкриття.

– PDF-файли зі скриптами — можуть містити JavaScript-код, що запускається при відкритті.

– Архіви (.zip, .rar) — дозволяють обійти фільтри, приховуючи шкідливий файл всередині.

– Виконувані файли (.exe, .js) — активне програмне забезпечення, яке одразу запускається (частіше — в атаках за межами корпоративного середовища).

Розподіл типів вкладень у фішингових атаках демонструє, що зловмисники переважно орієнтуються на ті формати, які часто зустрічаються в офіційній та діловій комунікації, що підвищує ймовірність відкриття без підозри. На графіку нижче подано частотність використання різних форматів за результатами аналізу понад 250 тисяч інцидентів за 2023 рік.[9]

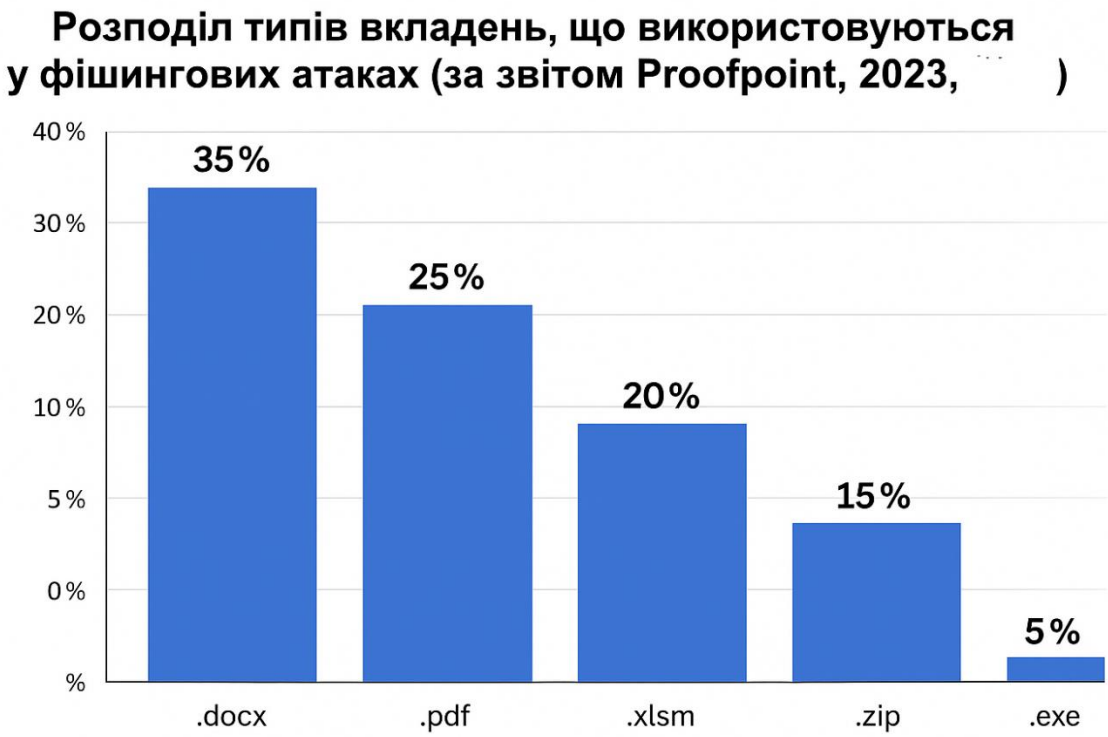


Рисунок 1.3 - Розподіл типів вкладень, що використовуються у фішингових атаках

Як видно з діаграми, найбільшу частку становлять документи Microsoft Office, зокрема .docx і .xlsm, які легко маскуються під службові документи. Високий відсоток PDF-файлів також свідчить про активне використання соціального фактору довіри до “перевірених” форматів. Використання архівів і виконуваних файлів є менш поширеним, але залишається ефективним в умовах низької обізнаності користувачів.

Активація загрози з вкладення відбувається лише у випадку, якщо виконуються дві умови: по-перше, користувач відкриває файл вручну, по-друге, у документі увімкнено макроси або інші активні елементи. Відтак можна подати це у вигляді логічної формули, яка моделює ймовірність успішного запуску коду.[10]

Процес запуску шкідливого коду із вкладення залежить від кількох умов — зокрема, налаштувань безпеки системи, дозволу на макроси, і дій користувача. Це можна формалізувати через умовну логіку:

$$R_{\text{exec}} = \begin{cases} 1, & \text{якщо } \text{macro\_enabled} = \text{TRUE} \wedge \text{user\_clicked} = \text{TRUE} \\ 0, & \text{інакше} \end{cases} \tag{1.2}$$

Де R_{exec} — результат виконання шкідливого коду.

Ця модель демонструє, що хоча вкладення саме по собі не є активною загрозою, ризик виникає саме на етапі взаємодії з користувачем. Тому цей тип атаки активно поєднується зі спуфінгом адреси відправника або психологічним тиском у листі, щоб змусити жертву довіритись вмісту документа.

Атаки через вкладення залишаються одним із найнебезпечніших векторів проникнення, оскільки вони напряму взаємодіють з операційною системою користувача. Надійний захист має включати не лише фільтрацію вкладень на рівні поштового шлюзу, а й обмеження виконання активного контенту в офісних програмах, навчання працівників, а також системи автоматичного виявлення шкідливої поведінки файлів (sandbox-аналіз).[10]

У міру того, як цифрові технології стають все більш мобільними та інтегрованими в повсякденне життя, фішинг теж еволюціонує, виходячи за межі класичних електронних листів. Сучасні форми атак дедалі частіше націлюються на

альтернативні канали комунікації, які важче контролювати за допомогою традиційних засобів кіберзахисту.

QR-фішинг (quishing):

Цей тип фішингу базується на використанні QR-кодів, які містять зашифровані шкідливі посилання. Користувач отримує нібито легітимне повідомлення (наприклад, від служби доставки чи банку), у якому пропонується просканувати QR-код для підтвердження транзакції або перегляду оновлення. Оскільки кінцева адреса при скануванні не відображається у зрозумілому вигляді, користувач переходить за нею без звичного візуального контролю.[11]

Особливу небезпеку quishing становить у фізичних просторах: фальшиві QR-коди можуть бути розміщені у публічних місцях, на афішах, чеках або навіть поверх справжніх наклейок у ресторанах та поштоматах.

Голосовий фішинг (vishing):

У випадку vishing-а зловмисник використовує телефонний дзвінок або голосове повідомлення, видаючи себе за представника банку, технічної підтримки чи держустанови. Часто такі дзвінки супроводжуються імітацією номеру телефону або автоінформатором для створення враження офіційності.

Головна мета — виманити конфіденційну інформацію, наприклад, код з SMS, реквізити картки або дані акаунта. Цей вид атаки особливо небезпечний для людей похилого віку, а також у стресових ситуаціях, коли людина не встигає критично оцінити ситуацію.

Smishing (SMS-фішинг):

Smishing використовує текстові повідомлення на мобільні телефони як канал доставки фішингового вмісту. У повідомленні зазвичай вказується термінова причина для переходу за посиланням: "Ваша посилка не може бути доставлена", "Ваша карта буде заблокована", "Оновіть акаунт через додаток". Посилання веде на підроблений вебсайт, де користувач вводить свої дані.[11]

Smishing складно виявити, оскільки:

– повідомлення приходить з короткого номеру, який виглядає офіційно;

– на мобільних пристроях важче розпізнати фальшиву URL-адресу;

– в деяких випадках зловмисники вже мають часткові дані про жертву з попередніх витоків.

Сучасні фішингові методи активно використовуються у мобільному середовищі, де відсутні звичні механізми перевірки — як-от попередній перегляд URL, аналіз заголовків листів або корпоративні поштові фільтри. За даними Lookout Security Report (2023), понад 52% фішингових атак у світі здійснюються не через email, а через мобільні повідомлення та дзвінки.[12]

Це свідчить про необхідність розширення захисних механізмів за межі традиційної поштової інфраструктури та впровадження багаторівневої автентифікації, зокрема для операцій у банківських додатках, сервісах доставки та авторизації через мобільні номери.

Сучасні фішингові атаки виходять за межі електронної пошти, активно використовуючи непоштові вектори впливу, такі як голос, SMS або QR. Це підвищує складність виявлення, збільшує площу атаки й потребує нового рівня кібергігієни та проактивної обізнаності користувачів. Такі форми атак особливо ефективні у мобільному середовищі, де користувачі приймають рішення швидко та часто без належної перевірки.

Фішингові атаки охоплюють широкий спектр технік — від спуфінгу адреси відправника до маскування гіперпосилань, шкідливих вкладень і цілеспрямованих атак на конкретних осіб. Окрему загрозу становлять сучасні методи через альтернативні канали: QR-коди, голосові дзвінки та SMS.[12]

Ефективність цих атак пояснюється поєднанням технічної уразливості поштових протоколів та впливом на людську поведінку. Більшість атак залишаються успішними через відсутність перевірки автентичності відправника, недостатню обізнаність користувачів і слабку інтеграцію захисних механізмів у мобільному середовищі.

1.3 Протоколи автентифікації електронної пошти: загальний огляд SPF, DKIM, DMARC

Електронна пошта залишається одним з основних каналів ділової та приватної комунікації, однак з моменту створення протоколу SMTP (Simple Mail Transfer Protocol) безпекові механізми в ньому не були передбачені. Історично SMTP був розроблений для відкритого, довіреного середовища на початку 1980-х років, коли питання автентифікації відправника не вважалося критичним.

Це створило фундаментальну вразливість: будь-хто, маючи доступ до SMTP-сервера, може вказати довільну адресу у полі "From:", не порушуючи жодного стандарту. Саме ця властивість лежить в основі фішингу, спуфінгу та атак соціальної інженерії, які масово використовуються зловмисникам[13]

Основні проблеми SMTP без автентифікації:

– Неможливо перевірити, чи має домен відправника право надсилати листа з вказаної IP-адреси.

– Користувач бачить лише поле "From", яке легко підробити.

– Поштові шлюзи не мають надійного способу визначити легітимність джерела повідомлення.

– Forwarding та пересилання листів часто змінюють шляхи доставки, що ще більше ускладнює виявлення шахрайства.

У результаті більшість фішингових атак є технічно допустимими з точки зору SMTP і блокуються лише завдяки евристичному аналізу або вручну, що є неефективним у масштабі.

Для усунення цієї системної прогалини було розроблено три взаємодоповнюючі протоколи:

– SPF (Sender Policy Framework) — перевіряє, чи дозволено конкретному серверу надсилати листи від імені домену.

– DKIM (DomainKeys Identified Mail) — забезпечує криптографічну перевірку цілісності листа та його походження.

– DMARC (Domain-based Message Authentication, Reporting and Conformance) — встановлює політики обробки листів, які не проходять перевірку SPF або DKIM.

Ці протоколи працюють разом і надають можливість поштовим серверам:

– Автоматично відхиляти або маркувати фішингові листи.[13]

– Створювати звітність про спроби підробки домену.

– Контролювати довіру до домену відправника через DNS-записи.

Автентифікація електронної пошти є не додатковою опцією, а необхідним стандартом для захисту від фішингових атак. Без SPF, DKIM та DMARC жодна поштово-орієнтована організація не може гарантувати безпеку своїх користувачів, партнерів і репутації власного домену.

SPF (Sender Policy Framework) — це механізм автентифікації електронної пошти, який дозволяє власникам доменів визначити перелік серверів, яким дозволено надсилати пошту від імені цього домену. Принцип дії SPF полягає у порівнянні IP-адреси сервера-відправника з дозволеним списком, що зберігається у вигляді DNS-запису типу TXT.

Це перший рівень захисту, який дозволяє виявляти спуфінг ще на етапі передачі повідомлення між серверами.

Принцип роботи SPF полягає в наступному:

– Власник домену публікує у DNS спеціальний TXT-запис, який містить перелік дозволених IP-адрес для надсилання пошти.

– Коли поштовий сервер отримує лист, він виконує DNS-запит і перевіряє, чи IP-адреса відправника відповідає дозволеним значенням.

– На основі результату приймається рішення: прийняти лист, відхилити або позначити як підозрілий.[14]

Щоб краще зрозуміти, як саме відбувається перевірка SPF у процесі доставки листа, доцільно розглянути послідовність дій між поштовим сервером одержувача, DNS-запитами та логікою прийняття рішення. Схема (рис. 1.4) нижче ілюструє базовий механізм, за яким сервер отримувача перевіряє дозволені IP-адреси на основі SPF-запису домену відправника.

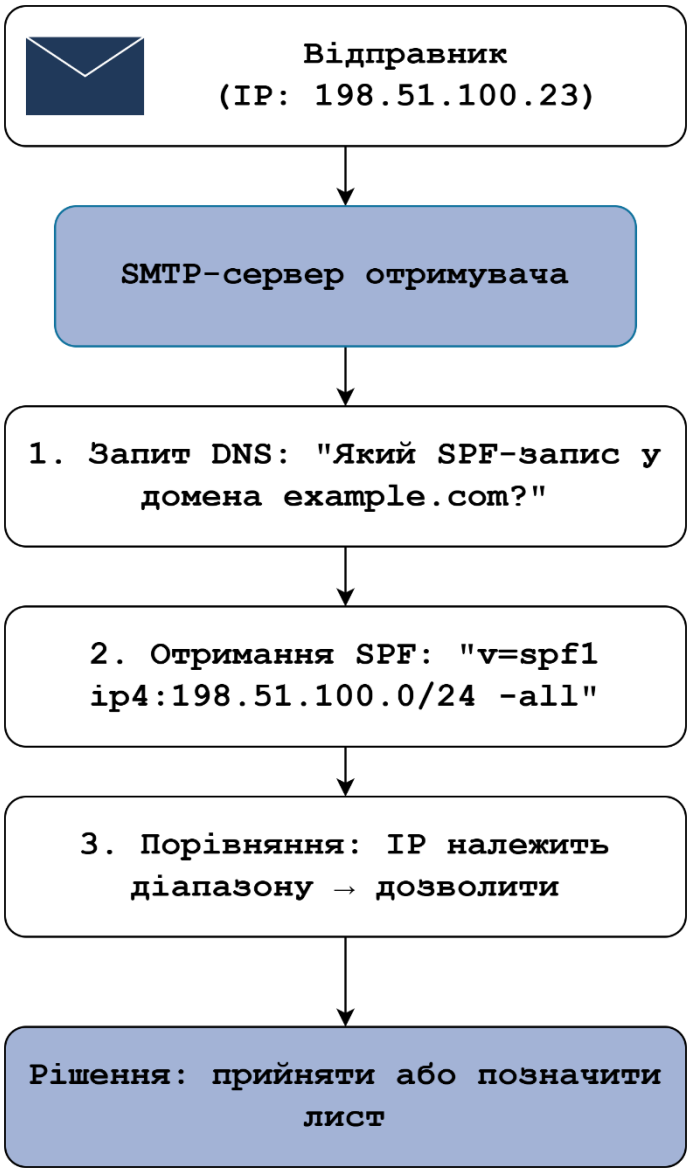


Рисунок 1.4 – Механізм перевірки SPF під час доставки електронного листа

Як видно зі схеми, SPF-запис діє як механізм допуску: він не підтверджує сам факт того, що лист справді походить від вказаної особи, але дозволяє серверу-одержувачу відсіяти всі IP-адреси, які явно не мають права надсилати листи від імені домену. Це робить SPF ефективним бар'єром проти базових форм спуфінгу, однак у поєднанні з DKIM та DMARC його ефективність зростає у рази.[14]

Хоча SPF ефективно захищає від прямого спуфінгу, він має низку обмежень:

– Не перевіряє вміст листа — лише IP-адресу відправника.

– Не працює з пересланими листами (forwarding), якщо пересилаючий сервер не входить у дозволений список.

– Не контролює піддомени, якщо вони не мають окремих SPF-записів.

– Легко обійти за допомогою підміни display name, коли адреса відображається знайомо, але домен — інший.

SPF є важливою складовою інфраструктури перевірки відправника в електронній пошті. Його застосування дозволяє зменшити кількість спуфінгованих листів і покращити загальний рівень довіри до поштових систем. Водночас, він не є самодостатнім інструментом і потребує доповнення криптографічним підписом DKIM, що буде розглянуто в наступному підпункті.

DKIM (DomainKeys Identified Mail) — це протокол, який забезпечує криптографічну перевірку справжності та цілісності електронного листа. На відміну від SPF, який перевіряє лише джерело доставки (IP-адресу), DKIM дозволяє перевірити, чи було повідомлення змінено під час передачі, і чи дійсно воно було надіслане з авторизованого домену.[15]

У DKIM кожне вихідне повідомлення підписується за допомогою приватного ключа, що належить відправнику. У той же час відкритий ключ публікується у DNS-зоні домену. Коли лист доходить до сервера-одержувача, він перевіряє криптографічний підпис, використовуючи відкритий ключ.

Це дозволяє:

– підтвердити, що лист справді був сформований на стороні авторизованого домену;

– переконатися, що вміст повідомлення (заголовки, текст, вкладення) не змінювався під час передачі.

Основні етапи перевірки DKIM:

– Відправник формує цифровий підпис частини листа (найчастіше — заголовків і тіла).

– Підпис додається до листа у вигляді заголовка DKIM-Signature.

– Сервер-одержувач отримує лист і зчитує публічний ключ із DNS.

– Система перевіряє відповідність підпису та контрольної суми.[15]

У DKIM криптографічна автентифікація базується на підписуванні частини вмісту електронного листа — зазвичай це вибрані заголовки та тіло повідомлення. Підпис формується на основі хеш-функції, а потім шифрується приватним ключем,

що належить домену відправника. На боці одержувача система верифікує підпис, розшифровуючи його за допомогою відкритого ключа, отриманого з DNS. Це дає змогу визначити, чи було повідомлення змінене під час доставки, і чи справді воно сформоване авторизованим джерелом.

Суть процесу можна формалізувати наступним чином:

$$\text{Підпис} = \text{Sign}_{\text{priv}}(H(\text{Headers} \parallel \text{Body}))$$

$$\text{Перевірка: Verify}_{\text{pub}}(\text{Підпис}) = H(\text{Headers} \parallel \text{Body}) \quad (1.3)$$

де:

H — хеш-функція (наприклад, SHA-256),

$\text{Sign}_{\text{priv}}$ — підписання приватним ключем домену,

$\text{Verify}_{\text{pub}}$ — перевірка підпису за допомогою відкритого ключа.[16]

Формула демонструє, що цілісність повідомлення забезпечується шляхом обчислення хешу з комбінації критичних елементів листа, які не повинні змінюватися. Важливо, що результат перевірки залежить не тільки від правильності підпису, але й від того, чи був лист модифікований на будь-якому етапі маршрутизації. Якщо навіть один символ змінено — перевірка не буде успішною.

Це робить DKIM особливо ефективним засобом виявлення спроб прихованої модифікації повідомлення — як на рівні вмісту, так і на рівні структури заголовків.

DKIM є ключовим елементом поштової автентифікації, який гарантує достовірність і незмінність вмісту повідомлення. На відміну від SPF, DKIM не залежить від IP-адреси, а працює на рівні цифрового підпису. Проте ефективність DKIM досягається лише у поєднанні з політикою перевірки — DMARC, яку буде розглянуто далі.[16]

DMARC — це політичний рівень автентифікації поштових повідомлень, що базується на результатах перевірок SPF та DKIM, і дозволяє власнику домену визначити, як саме сервери-одержувачі повинні поводитися з листами, які не проходять перевірку автентичності.

DMARC також підтримує механізм звітності, який дозволяє доменам-володільцям отримувати інформацію про спроби відправки фішингових або несанкціонованих повідомлень від їх імені.

Мета DMARC:

– Об’єднати SPF і DKIM у єдину політику обробки невідповідностей.

– Забезпечити механізм керування листами, що не проходять перевірку (приймати, поміщати в карантин або відхиляти).

– Надати засіб моніторингу через регулярні агреговані звіти (RUA) або звіт про інциденти (RUF).[17]

Сервер одержувача, отримавши електронний лист, виконує перевірку автентичності за двома механізмами — SPF і DKIM. Якщо хоча б один із них успішно спрацьовує та підтверджує відповідність домену, вказаного у полі From:, то перевірка DMARC вважається успішною.

У випадку, коли жоден з механізмів не підтвердив справжність повідомлення або виявлено невідповідність між доменами, сервер-одержувач діє відповідно до політики, зазначеної у DMARC-записі домену відправника. Політика може передбачати одне з трьох рішень: прийняти лист без втручання (none), надіслати його в карантин або позначити як спам (quarantine), або повністю відхилити повідомлення (reject).[18]

Переваги DMARC:

– Гарантує зв’язок між заголовком From: і реальним доменом, використовуючи механізм alignment.

– Дозволяє гнучко впроваджувати політику: від спостереження до жорсткого блокування.

– Підтримує регулярне звітування про використання домену третіми сторонами.

Приклад логіки перевірки:

– Якщо SPF успішний, але домен у результаті SPF ≠ домен у From: → DMARC не проходить.

– Якщо DKIM успішний, але підписаний інший домен → DMARC не проходить.

– Якщо будь-яка перевірка успішна і домен вирівняний → DMARC проходить.

DMARC є критично важливим компонентом безпеки електронної пошти, оскільки замикає систему автентифікації, забезпечуючи політичне трактування результатів SPF та DKIM. Він дозволяє власникам доменів контролювати використання своєї адреси, зменшувати кількість фішингових інцидентів та отримувати аналітику для подальшого вдосконалення політики безпеки. У сукупності з SPF та DKIM, DMARC формує сучасний триєдиний стандарт захисту поштової інфраструктури.[19]

Для системного впровадження автентифікації в електронній пошті необхідно розуміти функціональні відмінності між протоколами SPF, DKIM та DMARC. Кожен із них перевіряє різні параметри повідомлення на різних етапах доставки. У таблиці 1.4 наведено порівняння їхніх технічних характеристик, формату записів, механізмів перевірки та взаємодії з поштовою інфраструктурою.[20]

Таблиця 1.3 – Технічне порівняння протоколів SPF, DKIM і DMARC

Характеристика	SPF	DKIM	DMARC
Тип перевірки	IP-адреса сервера	Хеш-сума заголовків і тіла, цифровий підпис	Результати SPF/DKIM та відповідність домену в полі From:
Формат DNS-запису	TXT, префікс v=spf1	TXT, префікс v=DKIM1, відкритий ключ	TXT, префікс v=DMARC1, політика + адреса звітності
Взаємодія з полем From:	Необов’язкова	Необов’язкова	Обов’язкова (alignment)
Підтримка шифрування	Ні	Так (RSA, Ed25519)	Ні
Визначеність джерела повідомлення	Через IP	Через криптографічний підпис	Через результат SPF/DKIM та відповідність домену

Продовження таблиці 1.3

Залежність від SMTP маршруту	Так	Ні	Ні
Толерантність до пересилання (forwarding)	Немає (перевірка не проходить)	Є (підпис зберігається)	Є (за умови проходження SPF або DKIM)
Спосіб перевірки повідомлення	Під час SMTP-сеансу	Після отримання листа	Після отримання листа
Можливість керування поведінкою одержувача	Немає (сервер вирішує самостійно)	Немає	Є (режими: none, quarantine, reject)
Підтримка звітності	Відсутня	Відсутня	Присутня (XML-формат, канали RUA/RUF)

Протоколи SPF, DKIM і DMARC реалізують різні підходи до перевірки достовірності повідомлення: SPF ідентифікує джерело через IP-адресу, DKIM забезпечує криптографічну перевірку цілісності, а DMARC координує результати обох перевірок і визначає політику обробки. У комплексі вони дозволяють підвищити контроль над доставкою поштових повідомлень та зменшити імовірність несанкціонованого використання домену.

Протоколи SPF, DKIM і DMARC реалізують три незалежні рівні перевірки електронної пошти: відповідність IP-адреси, криптографічну автентичність вмісту та узгодження з доменом у полі From:. Їх застосування базується на DNS-записах і не вимагає змін у клієнтському ПЗ.[20]

SPF дозволяє перевірити дозволене джерело надсилання, DKIM забезпечує підтвердження цілісності повідомлення, а DMARC задає політику обробки повідомлень, які не проходять перевірку, та надає звітність. Комбіноване впровадження цих протоколів є критично необхідним для запобігання спуфінгу та фішингу в поштових системах.

1.4 Аналіз недоліків та вразливостей чинних механізмів автентифікації

Незважаючи на простоту впровадження та корисність для контролю джерел надсилання поштових повідомлень, протокол SPF має низку структурних обмежень, які унеможливлюють його використання як самостійного механізму захисту від фішингу.

Одним із найбільш критичних обмежень SPF є його непрацездатність при стандартному пересиланні листів. У цьому випадку повідомлення проходить через додатковий поштовий сервер, IP-адреса якого не включена до SPF-запису оригінального домену.[21]

У результаті перевірка SPF провалюється, незважаючи на те, що повідомлення не було змінене і було відправлене легітимним джерелом. Це створює умови для високого рівня хибнонегативних результатів, особливо в корпоративних середовищах з автоматизованими ланцюгами переадресації.

SPF перевіряє лише IP-адресу сервера, що здійснює надсилання, і не враховує зміст повідомлення чи заголовки (крім MAIL FROM). Це означає, що у разі компрометації легітимного поштового сервера, будь-який вміст листа буде вважатися достовірним, якщо IP-адреса відповідає SPF-запису.

SPF-запис повинен бути окремо вказаний для кожного піддомену. У разі його відсутності, перевірка не виконується. Це створює вразливість на рівні структур домену, коли основний домен має захист, але службові піддомени залишаються відкритими для спуфінгу.

SPF перевіряє адресу, зазначену у протоколі SMTP (MAIL FROM), але не аналізує поле From:, яке відображається користувачеві. Зловмисник може вказати адресу MAIL FROM із дозволеним SPF, але одночасно використати у полі From: інший домен, що викликає довіру.[22]

У підсумку SPF не захищає від візуального спуфінгу, що робить його непомітним для кінцевого користувача навіть при успішному проходженні перевірки.

Протокол SPF забезпечує лише базову перевірку IP-джерела, але не охоплює жоден з інших критичних аспектів автентифікації, включно з вмістом листа, підписом, узгодженням адрес або стійкістю до пересилання. У ізольованому використанні SPF не є надійним засобом захисту і повинен застосовуватись лише у зв'язці з DKIM і DMARC.

Протокол DKIM (DomainKeys Identified Mail) дозволяє перевірити цілісність вмісту повідомлення та підтвердити його походження через криптографічний підпис. Проте цей механізм також має ряд уразливостей, які обмежують його ефективність у реальному середовищі.[23]

DKIM не перевіряє джерело надсилання листа (IP-адресу SMTP-сервера). У випадку, якщо зловмисник отримує доступ до поштового сервера або надсилає лист із дозволеної платформи, він може створити валідний підпис, навіть якщо його наміри шкідливі. У цьому випадку SPF може провалитися, а DKIM — пройти перевірку.

DKIM є чутливим до будь-яких змін у листі після підписання. Переформатування, перекодування символів, додавання проміжного заголовка або навіть зміна пробілу можуть спричинити збій перевірки підпису. Це робить DKIM нестабільним у середовищах, де повідомлення проходять через кілька шлюзів або сервісів, які можуть модифікувати вміст.

Протокол DKIM не має вбудованого механізму звітності — ані відправник, ані адміністратор домену не дізнається про те, що його підпис не було перевірено або що він некоректний. У результаті системна помилка або конфігураційна неточність можуть залишатися непоміченими протягом тривалого часу.

DKIM базується на асиметричній криптографії, де публічний ключ зберігається в DNS-записі, а приватний — на поштовому сервері. У разі зламу або витоку приватного ключа зловмисник може створювати підписані листи від імені домену, які пройдуть перевірку на стороні отримувача.[24]

Підпис DKIM може охоплювати будь-який домен, вказаний у полі d= заголовка DKIM-Signature, і він не обов'язково має збігатися з доменом, що

відображається в полі From:. Без використання DMARC така невідповідність залишається непоміченою.

Попри свою технічну цінність, DKIM не вирішує завдання повної автентифікації відправника. Він не перевіряє IP-джерело, не контролює відповідність адреси у полі From:, не має механізмів реагування на помилки та є вразливим до компрометації ключа. Для забезпечення повного захисту DKIM має застосовуватись лише в поєднанні з DMARC.

Протокол DMARC (Domain-based Message Authentication, Reporting and Conformance) виконує функцію політичного контролю за результатами перевірки SPF і DKIM. Він дозволяє визначити, як поштовий сервер-одержувач має поводитися з листами, що не пройшли перевірку, та надає звітність. Проте ефективність DMARC безпосередньо залежить від налаштування та наявності підтримуючих механізмів, і це формує низку системних обмежень.[25]

DMARC не виконує перевірку самостійно — він лише аналізує результати SPF і DKIM. Якщо хоча б один із цих механізмів не налаштований або некоректно реалізований, DMARC втрачає здатність до ідентифікації достовірності повідомлення. Це створює точку відмови в системі автентифікації.

Незважаючи на технічну доступність, впровадження DMARC залишається добровільним. Поштові домени, що не використовують DMARC, не мають жодної централізованої політики захисту і не отримують жодних повідомлень про спроби несанкціонованого використання. Це зберігає в інфраструктурі поштової системи значну кількість незахищених доменів.

DMARC передбачає створення звітів (формати RUA та RUF), які надсилаються адміністратору домену у вигляді агрегованих або інцидентних XML-файлів. На практиці багато організацій:[26]

– не приймають або не обробляють звіти, попри наявність механізму;

– не мають автоматизованих засобів для обробки великої кількості даних;

– не виявляють спроб зловживання доменом, навіть якщо вони відбуваються.

Таким чином, можливості моніторингу часто не реалізуються повною мірою.

Дані досліджень (наприклад, Cisco 2023, Agari) свідчать, що більшість малих і середніх доменів не мають впроваджених DMARC-записів. Це пов’язано з браком технічних ресурсів, неповним розумінням принципів дії або відсутністю безпосередньої мотивації. У результаті велика частина поштового трафіку залишається поза сферою дії політик, навіть якщо отримувач підтримує DMARC.

DMARC виконує координуючу функцію, проте не є самодостатнім протоколом автентифікації. Його ефективність обмежена залежністю від SPF і DKIM, потребою у коректному конфігуруванні, неповним охопленням в реальній поштовій інфраструктурі та низьким рівнем використання механізмів звітності. Повноцінний захист можливий лише за умови комплексного впровадження та активного моніторингу.[27]

У таблиці 1.6 SPF, DKIM і DMARC оцінюються за 9 технічними критеріями, кожен з яких характеризує конкретний аспект безпеки або функціональності. Для кожного протоколу використовується оціночна шкала від 0 до 3, де:

- 0 — протокол не підтримує цю властивість або повністю вразливий;
- 1 — часткова або нестабільна реалізація;
- — базова реалізація з обмеженнями;
- — повна технічна підтримка без суттєвих недоліків.

Кожна оцінка відповідає на питання: наскільки даний протокол технічно здатний вирішити конкретне завдання, наприклад, чи перевіряє IP-адресу, чи витримує пересилання, чи гарантує цілісність вмісту тощо. Такий формат дозволяє кількісно порівняти протоколи за окремими характеристиками та у загальному підсумку.

Таблиця 1.4 – Кількісний аналіз стійкості SPF, DKIM і DMARC за критеріями безпеки

Критерій	SPF	DKIM	DMARC
Стійкість до спуфінгу	2	2	3
Переносимість через шлюзи	0	2	2

Продовження таблиці 1.4

Перевірка IP-джерела	3	0	3
Перевірка вмісту	0	3	3
Залежність від правильного DNS	3	3	3
Підтримка звітності	0	0	3
Вразливість до DNS-помилок	3	3	3
Стійкість до зловживань	1	2	3
Загальна технічна надійність	2.0	2.0	3.0

Порівняння за бальною шкалою дозволяє кількісно оцінити стійкість кожного протоколу. Найвищий загальний бал отримує DMARC (3.0), що пояснюється його координаційною роллю та можливістю задавати політики. SPF і DKIM мають однаковий середній показник (2.0), але кожен з них вразливий до окремих технічних умов — зокрема, SPF до пересилання, а DKIM до модифікацій вмісту або компрометації ключа.

Таке числове представлення підкреслює, що жоден з протоколів не може забезпечити комплексну безпеку у відриві від інших, а реальний захист досягається лише при одночасному та коректному використанні SPF, DKIM і DMARC.[28]

Для візуалізації рівня реалізації ключових функціональних характеристик протоколів SPF, DKIM і DMARC було проведено порівняльне оцінювання за 9 технічними критеріями. Кожен протокол оцінюється за шкалою від 0 до 3, що дозволяє представити ступінь їхньої ефективності, надійності та стійкості до типових вразливостей.

Графік нижче (рис.1.5) демонструє сумісне співвідношення сильних і слабких сторін кожного з трьох механізмів на основі кількісної моделі.

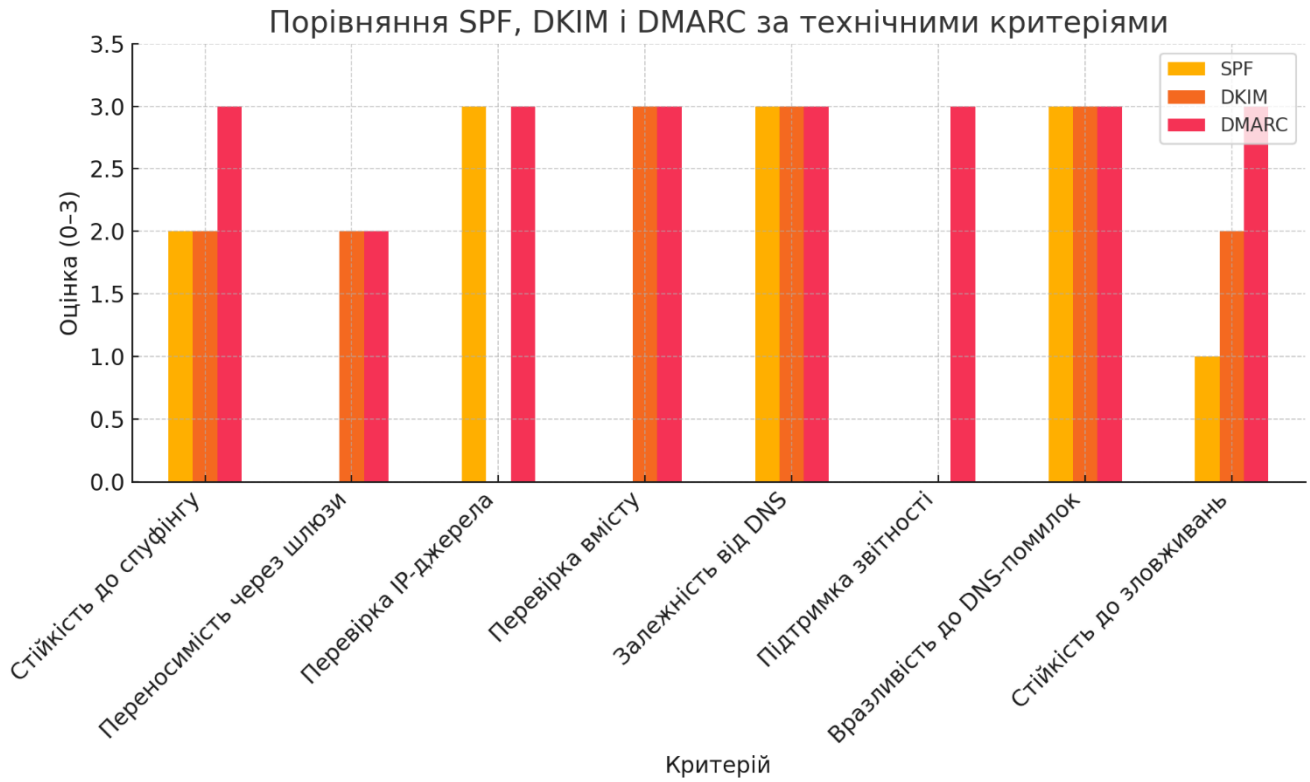


Рисунок 1.5 - Порівняння SPF, DKIM I DMARC за технічними критеріями

Графік ілюструє, що DMARC демонструє найбільш збалансовану реалізацію усіх ключових параметрів — насамперед завдяки інтеграції з SPF і DKIM та можливості застосовувати політики обробки. У той же час SPF втрачає ефективність у разі пересилання листів і не перевіряє вміст повідомлення, а DKIM — чутливий до модифікацій і не контролює IP-джерело.

Представлені дані підтверджують, що жоден протокол не є повністю самодостатнім, і лише комплексне впровадження всіх трьох систем дозволяє досягти допустимого рівня поштової автентифікації в умовах реального середовища.

У процесі аналізу встановлено, що фішингові атаки залишаються однією з найпоширеніших та ефективних форм соціальної інженерії, орієнтованих на електронну пошту. Основні техніки — спуфінг адреси, маскування посилань, шкідливі вкладення та цілеспрямовані атаки — ефективно комбінуються з використанням незахищених або слабо захищених поштових систем.

Розглянуті протоколи автентифікації електронної пошти — SPF, DKIM і DMARC — реалізують базові механізми перевірки джерела, цілісності та

відповідності відправника, проте виявлено низку технічних обмежень, які знижують їхню ефективність у реальних умовах:

– SPF не працює при пересиланні, не перевіряє вміст і не контролює візуальне поле From:.

– DKIM є вразливим до модифікацій листа і не враховує IP-джерело.

– DMARC залежить від двох інших механізмів і не є обов'язковим стандартом, а звітність часто ігнорується.

Комплексний аналіз показав, що навіть при формальному впровадженні протоколів автентифікації, залишається ризик обходу захисту через технічні та конфігураційні недоліки, або внаслідок відсутності їх повного використання серед учасників поштового обміну.

1.5 Висновок до розділу 1 та постановка задачі

У цьому розділі було проаналізовано основні загрози для електронної пошти, пов'язані з фішинговими атаками, а також класифіковано найпоширеніші методи фішингу: підміна адреси відправника, маскування гіперпосилань, використання шкідливих вкладень, цілеспрямовані атаки (spear phishing, whaling) та сучасні варіанти через QR-коди, SMS і голосові дзвінки.

Окрему увагу приділено технічному огляду протоколів автентифікації SPF, DKIM і DMARC, які реалізують механізми перевірки джерела, вмісту повідомлення та політики домену. Проведено аналіз обмежень і вразливостей кожного з них, а також оцінено їхню взаємозалежність, яка формує спільну точку відмови в системі захисту електронної пошти.

На основі проведеного аналізу сформульовано такі задачі для подальшого дослідження:

– розробити удосконалений підхід до автентифікації поштових повідомлень, який враховує вразливості SPF, DKIM та DMARC, а також компенсує їхню взаємозалежність;

– створити технічну модель механізму перевірки достовірності, що забезпечує стійкість до спуфінгу при forward-навантаженнях і підміні візуальних адрес;

– запропонувати додаткові рівні верифікації повідомлення, зокрема на основі поведінкових ознак, часових міток або історії взаємодій;

– реалізувати прототип або архітектурну схему запропонованого рішення, яка може бути впроваджена на поштовому шлюзі або в доменній політиці;

– провести тестування захищеності оновленої моделі на прикладі симульованих фішингових атак.

АЛГОРИТМ ВІДСЛІДКОВУВАННЯ НЕСАНКЦІОНОВАНИХ ДІЙ КОРИСТУВАЧІВ

У цьому розділі буде представлено авторський підхід до підвищення захищеності електронної пошти шляхом удосконалення існуючих механізмів автентифікації. Розглянуто особливості формування політик SPF, DKIM та DMARC, запропоновано комбіновану стратегію з додатковими перевірками та механізмами стійкості до типових векторів атак. Також буде розроблено алгоритм обробки поштових повідомлень із багаторівневою логікою верифікації та обґрунтовано архітектуру реалізації покращеної моделі захисту.

2.1 Особливості формування політик SPF, DKIM та DMARC

Надійна автентифікація електронної пошти базується не лише на підтримці протоколів, але й на точності їхньої конфігурації у доменній зоні DNS. Успішне застосування SPF, DKIM і DMARC можливе лише за умови правильного формування відповідних політик, які дозволяють серверу-одержувачу приймати технічно обґрунтовані рішення щодо довіри до повідомлення.

Політики автентифікації є набором параметрів, що визначають:

- дозволені джерела надсилання (для SPF),
- криптографічну прив’язку до домену (для DKIM),
- логіку реакції на невдачі перевірки та механізм звітності (для DMARC).

Ці параметри задаються через TXT-записи в DNS та потребують точного дотримання специфікацій, врахування особливостей поштової інфраструктури, маршрутизації, використання сторонніх сервісів, а також відповідності політик поточному рівню безпекових загроз.[29]

У цьому підпункті розглянуто ключові елементи формування політик для кожного з протоколів, типові параметри та практичні вимоги до їх налаштування.

Формування ефективної системи автентифікації електронної пошти починається з правильної конфігурації політик у DNS-записах домену. Кожен із трьох основних протоколів — SPF, DKIM та DMARC — реалізується через

публікацію текстових (TXT) записів у доменній зоні DNS. Від точності, повноти та узгодженості цих записів залежить коректність перевірки повідомлень на стороні поштових серверів одержувача.

SPF: дозвіл на відправлення з визначених IP

Політика SPF визначає, які IP-адреси або домени мають право надсилати пошту від імені конкретного домену. Типовий SPF-запис публікується як TXT-запис у вигляді:

```
v=spf1 ip4:192.0.2.0/24 include:_spf.example.com -all
```

Ключові елементи:

-ip4: — конкретні дозволені діапазони IPv4;

-include: — довірені сторонні сервіси (наприклад, Google, Microsoft);

--all — політика відхилення всіх інших адрес (- = жорстке блокування, ~ = помірна недовіра, ? = нейтрально).

Для коректної роботи необхідно:

-охопити всі поштові платформи, що використовуються доменом;

-обмежити кількість DNS-запитів до 10 (вимога стандарту);

-перевіряти наявність записів для усіх піддоменів.

DKIM: генерація та публікація відкритого ключа

Для DKIM необхідно:

–Згенерувати пару криптографічних ключів (приватний і публічний).

–Налаштувати поштовий сервер на підпис повідомлень приватним ключем.

–Опублікувати публічний ключ у DNS у вигляді:

```
default._domainkey.example.com  IN  TXT  "v=DKIM1; k=rsa; p=MIIBIjANBgkqh..."
```

Параметри:

–default — селектор ключа (можна змінювати для ротації);

–p= — публічний ключ у форматі base64;

–k=rsa — тип алгоритму (RSA або Ed25519).

Особливість DKIM — можливість підписувати окремі заголовки (наприклад, From, Subject, To), що задається параметром h= у заголовку DKIM-Signature.

Для ефективності рекомендується:

–починати з p=none для моніторингу;

–аналізувати звіти й поступово переходити до quarantine → reject;

–використовувати суворе вирівнювання (s), щоб уникнути спуфінгу через суміжні домени.

Формування політик SPF, DKIM і DMARC — це не одноразова дія, а поступовий процес конфігурації, перевірки та коригування, що потребує точності, технічного розуміння структури поштових маршрутів та регулярного аналізу результатів. Неправильно вказані записи можуть призвести як до блокування легітимної пошти, так і до пропуску шкідливих повідомлень. Саме тому усі три протоколи мають бути налаштовані в комплексі відповідно до фактичного використання домену.

2.2 Розробка комбінованої стратегії автентифікації з підвищеним рівнем захисту

Наявні протоколи автентифікації електронної пошти (SPF, DKIM, DMARC) працюють ефективно лише за умов їх коректного впровадження та взаємодії. Водночас, як показано у попередньому розділі, їх технічні обмеження створюють численні вектори обходу, зокрема при пересиланні листів, використанні піддоменів або неправильній конфігурації DNS-записів.

З метою усунення найбільш критичних недоліків і підвищення загального рівня довіри до повідомлень у поштовій системі пропонується розробити комбіновану стратегію автентифікації, яка доповнює базову трійку протоколів рядом додаткових перевірок і уточнень.

Основні принципи запропонованої стратегії

1. Строге узгодження (alignment) усіх доменів у заголовках та підписах:
   - домен у полі From: повинен повністю збігатися з доменами в SPF і DKIM;
   - виключити використання третіх доменів без явного делегування.

2. Верифікація пересилання з використанням ARC (Authenticated Received Chain):
   - реалізувати підтримку протоколу ARC для збереження SPF/DKIM-перевірок при forwarding;
   - аналізувати ланцюг поштових серверів за підписаним ARC-заголовком.
3. Ротація DKIM-ключів з коротким строком життя:
   - впровадити регулярну генерацію та публікацію нових DKIM-селекторів;
   - обмежити використання одного ключа не більше ніж на 30 днів.
4. Підвищений рівень політики DMARC:
   - перехід до режиму p=reject з повним adkim=s та aspf=s;
   - обов’язкова перевірка та обробка звітів RUA/RUF на сервері безпеки.
5. Формування додаткової логіки на шлюзі доставки (Postfix/Exim):
   - порівняння полів Return-Path, From, Reply-To;
   - перевірка доменів у вкладених URL (blacklist/whitelist);
   - геолокаційний та часовий аналіз (наприклад, виявлення аномальної активності).

Пропонована стратегія реалізується на трьох рівнях:

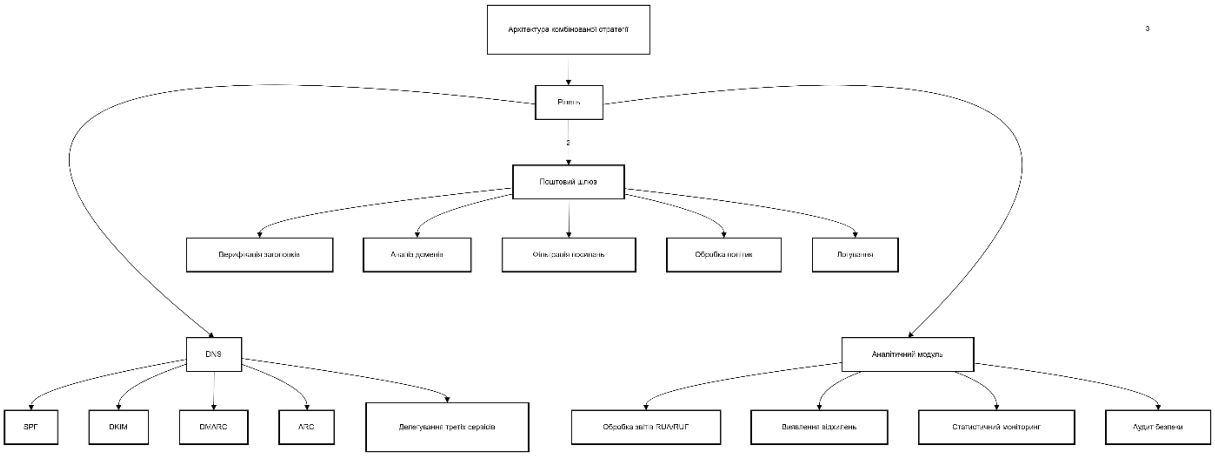


Рисунок 2.1 – Архітектура запропонованої стратегії

Очікувані переваги

–усунення векторів обходу, пов’язаних із forwarding;

–зниження кількості хибнопозитивних і хибнонегативних результатів;

–підвищення довіри до корпоративного домену серед зовнішніх поштових систем;

–покращення прозорості та контрольованості комунікацій за рахунок звітності.

Комбінована стратегія автентифікації забезпечує багаторівневу перевірку достовірності поштових повідомлень з урахуванням обмежень класичних протоколів. Її реалізація дозволяє суттєво знизити ймовірність успішного спуфінгу та фішингових атак за рахунок гнучкої перевірки маршрутів доставки, підписів, політик та контекстних факторів.

2.3 Розробка алгоритму обробки поштових повідомлень з перевіркою SPF, DKIM та DMARC

Для забезпечення цілісної перевірки автентичності вхідних поштових повідомлень необхідно реалізувати чітко структурований алгоритм обробки, який поєднує стандартні механізми SPF, DKIM і DMARC з додатковими рівнями контролю. Такий підхід дозволяє виявляти спроби спуфінгу, підміни заголовків, порушення політик домену та зміни вмісту повідомлення, які могли бути внесені на шляху доставки.

У цьому підпункті запропоновано поетапний алгоритм, який може бути реалізований у вигляді фільтра на рівні поштового шлюзу або інтегрований у MTA (Mail Transfer Agent) як додатковий модуль перевірки.

Розробимо та покроковуо розберемо даний алгоритм (рис.2.2).

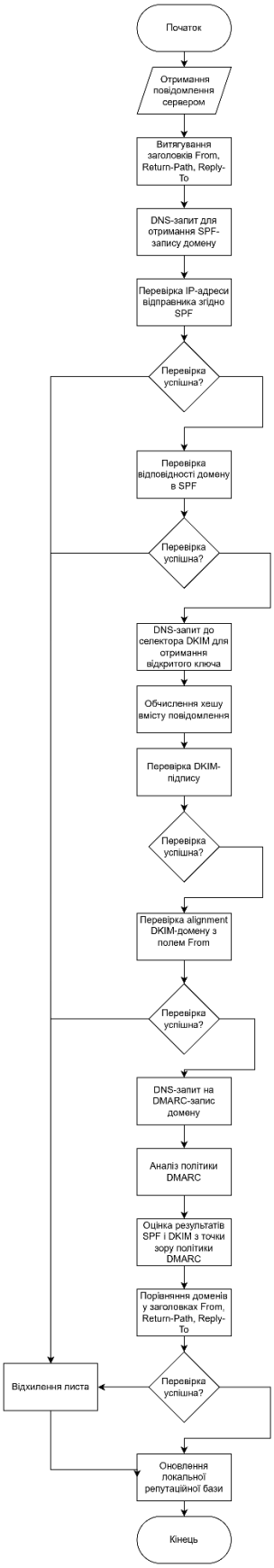


Рисунок 2.2 – Розроблений алгоритм обробки поштових повідомлень з перевіркою SPF, DKIM та DMARC

Крок 1. Отримання повідомлення сервером

Поштовий сервер-одержувач приймає вхідне повідомлення на SMTP-рівні та формує метадані: IP-адреса відправника, домен, заголовки, вміст.

Крок 2. Витягування заголовків From, Return-Path, Reply-To

Виділяються ключові заголовки, які будуть порівнюватись для виявлення розбіжностей і потенційної підміни.

Крок 3. DNS-запит для отримання SPF-запису домену

Виконується DNS-запит на наявність і вміст SPF-запису домену, вказаного у MAIL FROM або Return-Path.

Крок 4. Перевірка IP-адреси відправника згідно SPF

Порівнюється IP-адреса реального сервера з дозволеними адресами у SPF-записі. Фіксується результат перевірки (pass, fail, softfail, neutral, none).

Крок 5. Перевірка відповідності домену в SPF і From: (alignment)

Здійснюється перевірка, чи збігається домен у SPF із доменом, вказаним у заголовку From: — у суворому (s) або помірному (r) режимі.

Крок 6. DNS-запит до селектора DKIM для отримання відкритого ключа

На основі заголовка DKIM-Signature виконується DNS-запит до відповідного селектора (selector._domainkey.domain).

Крок 7. Обчислення хешу вмісту повідомлення

Обчислюється контрольна сума вказаних у підписі DKIM заголовків і тіла листа за алгоритмом (наприклад, SHA-256).

Крок 8. Перевірка DKIM-підпису

Підпис у заголовку DKIM-Signature порівнюється з контрольним значенням, розшифрованим за допомогою відкритого ключа.

Крок 9. Перевірка alignment DKIM-домену з полем From

Визначається, чи збігається домен у полі d= заголовка DKIM з доменом у полі From:.

Крок 10. DNS-запит на DMARC-запис домену

Пошук TXT-запису v=DMARC1 у DNS для домену в полі From:.

Крок 11. Аналіз політики DMARC

Визначається, яку політику застосовує домен: none, quarantine чи reject. Зчитуються параметри вирівнювання (aspf, adkim) та адреси звітності.

Крок 12. Оцінка результатів SPF і DKIM з точки зору політики DMARC

Якщо хоча б одна перевірка (SPF або DKIM) успішна і вирівнювання доменів підтверджене — вважається, що повідомлення проходить DMARC. Інакше — застосовується політика.

Крок 13. Порівняння доменів у заголовках From, Return-Path, Reply-To

Проводиться детекція розбіжностей між ключовими заголовками, які можуть вказувати на спробу маскування або редиректу.

Крок 14. Перевірка ARC-ланцюга (якщо є)

Якщо присутній ланцюг ARC (ARC-Seal, ARC-Message-Signature), перевіряється його цілісність і використовується для уточнення SPF/DKIM результатів при forwarding.

Крок 15. Верифікація вкладених URL-адрес (опціонально)

Здійснюється автоматичний аналіз гіперпосилань у тілі повідомлення: перевірка доменів на відповідність основному відправнику, наявність у чорних/білих списках.

Крок 16. Формування остаточного рішення

На основі зібраних результатів формулюється рішення: прийняти, відправити в спам/карантин або відхилити. Це рішення враховує:

–політику DMARC;

–узгодження доменів;

–інші сигнали ризику.

Крок 17. Логування результатів перевірки

Результати перевірки зберігаються у логах для подальшого аудиту, моніторингу статистики і формування звітів.

Крок 18. Генерація звіту для адміністратора (RUA/RUF)

Якщо активована підтримка звітності DMARC, генерується звіт у форматі XML і надсилається за адресою, вказаною у rua= або ruf=.

Крок 19. Оновлення локальної репутаційної бази

На основі поведінки відправника (повторювані невдачі, спроби обману) оновлюється внутрішній список довіри/ризику.

Крок 20. Завершення обробки

Повідомлення передається до наступного етапу — збереження у скриньці, карантин, або відхилення із відповідним кодом статусу SMTP.

Розроблений алгоритм обробки поштових повідомлень забезпечує поетапну перевірку автентичності на основі стандартів SPF, DKIM та DMARC з урахуванням додаткових факторів, таких як аналіз заголовків, обробка ARC-ланцюгів, перевірка вкладених URL-адрес та контроль політик домену.

Алгоритм реалізує 20 послідовних кроків, кожен з яких має конкретне технічне призначення та взаємопов’язаний із наступними етапами перевірки. Особливу увагу приділено вирівнюванню доменів (alignment), обробці forwarding-сценаріїв, а також реєстрації результатів для аудиту і звітності.

Таким чином, запропонований механізм є універсальним ядром системи поштової автентифікації, що може бути реалізований як окремий модуль на рівні поштового шлюзу або інтегрований у існуючу інфраструктуру організації.

2.4 Висновок до розділу 2

У цьому розділі було здійснено розробку комбінованого підходу до автентифікації електронної пошти з метою підвищення її захищеності від фішингових атак. Враховуючи технічні обмеження кожного з окремих протоколів — SPF, DKIM та DMARC — акцент було зроблено на їх правильному об'єднанні, доповненні та адаптації до сучасних сценаріїв загроз, зокрема пересилання повідомлень, використання третіх поштових сервісів, спуфінгу заголовків та підміни адреси відправника.

У підпункті 2.1 проаналізовано специфіку формування політик SPF, DKIM і DMARC на рівні DNS-записів. Детально розглянуто вимоги до їхньої структури, параметрів вирівнювання доменів (alignment), а також типові помилки при конфігурації, які призводять до обходу перевірок.

Ця стратегія має гнучку архітектуру та може бути реалізована як модуль на рівні поштового шлюзу або MTA, з подальшою інтеграцією у внутрішню систему моніторингу безпеки.

Найважливішим результатом розділу стала розробка алгоритму обробки поштових повідомлень (підпункт 2.3), який реалізує 20 послідовних кроків перевірки: від отримання листа до прийняття остаточного рішення про його обробку. У межах алгоритму враховано взаємодію всіх компонентів автентифікації, виконання перевірок DNS-записів, обробку результатів, логування, генерацію звітності (RUA/RUF) та оновлення локальної репутаційної бази. Також додано механізм контролю за зміною заголовків і оцінкою змісту посилань.

Запропонований підхід орієнтовано на практичне застосування в умовах реальної інфраструктури, де присутні численні маршрути доставки, сторонні сервіси, обмеження з боку політик домену, та ризик цілеспрямованих фішингових кампаній.

Результати розділу 2 дозволяють перейти до практичного етапу реалізації захисту поштових повідомлень та експериментальної перевірки ефективності розробленого підходу в умовах змодельованих загроз.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО МОДУЛЮ

У цьому розділі описано реалізацію та тестування програмного модуля для підвищення захищеності електронної пошти від фішингових атак шляхом удосконаленої перевірки автентичності повідомлень. На основі сформульованої у попередньому розділі стратегії було обрано відповідну мову програмування та середовище розробки, спроєктовано модулі взаємодії з DNS, перевірки SPF, DKIM та DMARC, а також додаткових захисних механізмів.

Особливу увагу приділено перевірці коректності конфігурацій, обробці виняткових ситуацій (наприклад, відсутності записів або некоректної структури підписів), логуванню результатів перевірки та зручності інтеграції розробленого рішення у існуючу поштову інфраструктуру.

3.1 Обґрунтування вибору мови програмування та середовища розробки програмного модулю

Для реалізації програмного модуля, який виконує перевірку автентичності електронних листів з використанням протоколів SPF, DKIM та DMARC, було обрано мову програмування Python. Основною причиною цього вибору стала висока гнучкість мови, її простий синтаксис та наявність розвиненої екосистеми бібліотек, що значно полегшує реалізацію логіки, пов’язаної з обробкою поштових протоколів. Python забезпечує зручний доступ до DNS-запитів, обробки MIME-заголовків, хешування, цифрових підписів і аналізу вмісту повідомлень. Завдяки готовим бібліотекам, таким як `dnspython`, `dkimpy`, `pyspf`, `authheaders` та стандартним модулям для роботи з поштовими повідомленнями, можна швидко реалізувати функціонал перевірки SPF-записів, валідації DKIM-підписів та аналізу політик DMARC без необхідності самостійно реалізовувати низькорівневі механізми.[30]

Python також дозволяє легко масштабувати систему, інтегрувати її з поштовими шлюзами (Postfix, Exim), журналами подій, API безпеки або внутрішніми моніторинговими сервісами. Його кросплатформеність дає змогу запускати програму на більшості серверних систем, що особливо актуально в

контексті розгортання модулю на рівні організаційної поштової інфраструктури. Завдяки активному розвитку мови Python та постійному оновленню бібліотек, розробник має змогу швидко адаптувати модуль до змін у форматах записів, політик або вимог.

Python є однією з найпопулярніших і водночас найбільш універсальних мов програмування, яка широко використовується у сферах мережевої безпеки, аналізу трафіку, поштових фільтрів, тестування вразливостей та автоматизації кіберзахисту. Його вибір у даному проєкті обумовлений насамперед поєднанням гнучкості, простоти синтаксису та потужної екосистеми бібліотек, що спеціалізуються на роботі з інтернет-протоколами, криптографією, DNS та поштовими заголовками.[31]

Мова Python повністю підтримує розробку програм, які здійснюють обробку SMTP, POP3 та IMAP-трафіку, аналізують MIME-структури повідомлень, витягують заголовки та вкладення, а також взаємодіють із DNS для перевірки SPF, DKIM і DMARC-записів. Це особливо важливо для побудови комплексного модуля перевірки автентичності пошти. Стандартна бібліотека Python (email, smtplib, re, hashlib, base64) забезпечує доступ до всіх базових операцій із заголовками, хешами, кодуваннями, що дозволяє обробляти повідомлення з мінімальним використанням стороннього коду. Крім того, Python підтримує роботу з текстовими форматами (TXT-записи в DNS, XML-звіти DMARC), що критично для реалізації функціоналу автентифікації.

На відміну від низькорівневих мов, таких як C або Java, Python дозволяє реалізувати функціональну перевірку SPF або розбір DKIM-підпису за кілька десятків рядків коду, використовуючи бібліотеки pyspf, dkimpy, dnspython або authheaders. Ці бібліотеки є відкритими, активно підтримуються спільнотою, мають зрозумілий API, документацію та не вимагають складної конфігурації. Наприклад, бібліотека dkimpy дозволяє виконати перевірку підпису DKIM на основі MIME-повідомлення всього в один виклик функції, автоматично обробляючи селектор, ключ і алгоритм підпису.[32]

Python також підтримує асинхронність та паралельну обробку листів, що дозволяє масштабувати модуль для роботи з великим потоком пошти. Модулі asyncio, threading, multiprocessing або асинхронні фреймворки (наприклад, FastAPI, aiohttp) дозволяють реалізувати високонавантажені системи без значної втрати продуктивності. Окрім цього, Python легко інтегрується з API сторонніх сервісів (наприклад, антивірусних сканерів або спам-фільтрів), що може бути корисним при подальшому розширенні проєкту.[33]

Python також забезпечує відмінні можливості для тестування та логування. Фреймворки pytest, unittest та doctest дозволяють автоматизувати перевірку кожної функції, а бібліотеки loguru, logging або structlog — легко інтегрувати повноцінне системне журналювання з фільтрацією, ротацією та відправкою логів у зовнішні системи.

Ще однією перевагою Python є його кросплатформеність. Розроблений модуль може бути розгорнутий на будь-якій операційній системі, включно з Linux, Windows і macOS. Його можна запускати як окрему службу, скрипт, фільтр MTA або елемент CI/CD-процесу в поштовій інфраструктурі.[34]

Таким чином, Python є найбільш раціональним вибором для створення програмного модуля захисту пошти, оскільки поєднує швидкість розробки, підтримку поштових стандартів, наявність готових бібліотек, простоту інтеграції з DNS і серверною інфраструктурою, а також зручність масштабування та супроводу коду в майбутньому.

У якості середовища розробки було обрано PyCharm, оскільки воно забезпечує стабільну підтримку мови Python, зручні інструменти для навігації, налагодження та автоматичного аналізу коду. Використання PyCharm дозволяє пришвидшити розробку за рахунок вбудованої підтримки системи керування версіями, роботи з віртуальними середовищами, автодоповнення коду та інтегрованого терміналу. Це середовище також добре підходить для модульного тестування й підтримки структурованої архітектури проєкту, яка в даній роботі поділена на окремі функціональні компоненти: обробка DNS-запитів, перевірка

SPF, аналіз DKIM-підпису, валідація політик DMARC, логування та генерація звітів.[35]

Для реалізації програмного модуля захисту електронної пошти у межах даної роботи було обрано середовище розробки PyCharm Community Edition — повноцінну інтегровану середу (IDE), спеціалізовану для мови програмування Python. Основна причина такого вибору полягає в тому, що PyCharm забезпечує не лише зручність написання коду, а й вбудовану підтримку всіх компонентів, необхідних для розробки, налагодження, тестування та розгортання програмного забезпечення, що працює з мережею, DNS, електронною поштою та API.

Однією з основних переваг PyCharm є контекстно-залежне автодоповнення коду, яке значно пришвидшує написання функціональних частин програми. При роботі з великою кількістю бібліотек, таких як dnspython, dkimpy, pyspf, authheaders, email, logging, середовище автоматично підказує допустимі методи, доступні атрибути, типи аргументів та дозволяє миттєво переходити до їхньої реалізації чи документації. Це значно зменшує кількість синтаксичних помилок і покращує читабельність проєкту.[36]

Середовище підтримує створення ізольованих середовищ (venv), завдяки чому розробку можна вести незалежно від глобальних інсталяцій Python. Всі залежності, необхідні для виконання проєкту, фіксуються у файлі requirements.txt, що забезпечує відтворюваність середовища на інших машинах або серверах.

Також PyCharm має вбудований відлагоджувач (debugger) з покроковим виконанням коду, моніторингом змінних, контрольних точок (breakpoints) і стеків викликів. Це особливо корисно для діагностики помилок у процесі обробки поштових повідомлень, перевірки DNS-відповідей або розшифрування DKIM-підписів.

Інтеграція з системою контролю версій Git дозволяє зберігати історію змін, створювати гілки для експериментів, фіксувати стабільні версії, а також легко синхронізувати локальну розробку з репозиторіями на GitHub або GitLab. Це особливо важливо у проєктах, які передбачають командну роботу або подальшу публікацію розробленого модуля як open-source рішення.

Окрім того, PyCharm підтримує структуризацію коду за шаблонами, що дозволяє легко організовувати проєкт на рівні логічних модулів: окремі файли для перевірки SPF (spf_module.py), обробки DKIM (dkim_module.py), логіки об’єднання результатів (auth_manager.py), логування (log_service.py), генерації звітів (reporter.py) тощо.[37]

Завдяки вбудованим інструментам тестування (через unittest, pytest) розробник може швидко реалізувати юніт-тести для критичних ділянок коду — наприклад, перевірки вирівнювання доменів або коректної обробки помилкових DNS-записів.

Важливо зазначити, що навіть безкоштовна версія середовища — PyCharm Community Edition — надає повноцінний набір функцій для професійної розробки, включаючи інтегрований термінал, PEP8-аналіз коду, підказки щодо оптимізації, візуальну навігацію та інструменти профілювання.

Таким чином, вибір PyCharm як основного середовища розробки забезпечує максимальну ефективність і надійність процесу реалізації програмного модуля, дозволяючи зосередитися безпосередньо на логіці автентифікації та обробки загроз, а не на рутинних аспектах налаштування або налагодження коду. Його використання значно скоротило час розробки, підвищило стабільність рішення та забезпечило зручну платформу для подальшого супроводу і масштабування.[38]

Таким чином, комбінація Python та PyCharm забезпечила оптимальне середовище для реалізації захищеного поштового модуля з точки зору швидкості розробки, підтримки профільних бібліотек, кросплатформеності, розширюваності та зручності тестування в рамках лабораторного або серверного розгортання.

3.2 Реалзація програмного модуля керування SPF, DKIM та DMARC

На основі сформованих алгоритмів перевірки поштових повідомлень було реалізовано окремий програмний модуль, що виконує автоматизовану перевірку записів SPF, DKIM та DMARC для вхідних електронних листів. Модуль написано мовою Python у середовищі PyCharm та структуровано у вигляді набору функціональних компонентів з розмежованими зонами відповідальності.

У загальному вигляді модуль складається з трьох ключових частин: `spf_checker.py`, `dkim_checker.py` та `dmarc_parser.py`, кожна з яких відповідає за обробку відповідного механізму автентифікації. Основна точка входу реалізована в модулі `email_auth_handler.py`, де здійснюється координація перевірок та об'єднання результатів.

Для реалізації перевірки SPF-запису використано бібліотеку pyspf, яка повністю відповідає специфікації RFC 7208 і дозволяє здійснювати оцінку IP-адреси відправника на відповідність запису SPF, що розміщений у DNS.

Для початку необхідно встановити бібліотеку pyspf, якщо вона ще не інстальована:

```
pip install pyspf
```

У коді імпортуємо модуль:

```
import spf
```

Функція spf.check2(ip, sender, helo) приймає три аргументи:

- ip — IP-адреса SMTP-клієнта (сервер, який надсилає лист),
- sender — адреса у форматі MAIL FROM (SMTP Envelope),
- helo — ідентифікатор HELO/EHLO (заголовок сеансу SMTP).

```
import spf

def check_spf(ip_address, mail_from, helo_domain):
    try:
        result, code, explanation = spf.check2(ip_address, mail_from, helo_domain)
        return {
            'result': result,
            'code': code,
            'explanation': explanation
        }
    except Exception as e:
        return {
            'result': 'permerror',
            'error': str(e)
        }
```

Пояснення коду

- result: логічний результат перевірки SPF — може бути pass, fail, softfail, neutral, none, temperror, або permerror.
- code: службовий код відповідно до специфікації.
- explanation: текстове пояснення від бібліотеки щодо отриманого результату (може бути корисним для журналювання).

–У випадку винятків (наприклад, помилка DNS-запиту або неправильний формат адреси), результатом буде permerror.

```
if __name__ == "__main__":
    ip = '198.51.100.10'
    mail_from = 'test@example.com'
    helo = 'mail.example.com'

    result = check_spf(ip, mail_from, helo)
    print(result)
```

Очікуваний вивід (якщо SPF-запис є і IP дозволено):

```
{
    'result': 'pass',
    'code': 250,
    'explanation': 'domain of example.com designates 198.51.100.10 as permitted sender'
}
```

Оскільки SPF перевіряє адресу MAIL FROM, а фішингові атаки часто використовують підміну у полі From:, важливо додатково перевірити чи збігається домен у MAIL FROM із доменом у From:. Це реалізується вручну:

```
def spf_alignment(mail_from, header_from):
    try:
        domain_envelope = mail_from.split('@')[-1].lower()
        domain_header = header_from.split('@')[-1].lower()
        return domain_envelope == domain_header
    except Exception:
        return False
```

Таким чином, модуль перевірки SPF реалізує як базову перевірку дозволених IP, так і додаткову перевірку доменного вирівнювання, яка критично важлива в контексті політик DMARC. Це дозволяє виявляти не лише прямі спуфінг-атаки, але й складніші схеми, що використовують підміну візуального поля From: без порушення SMTP-контексту.

Перевірка підпису DKIM (DomainKeys Identified Mail) є ключовим елементом автентифікації поштових повідомлень. На відміну від SPF, DKIM перевіряє не IP-адресу відправника, а цілісність вмісту повідомлення за допомогою криптографічного підпису, сформованого приватним ключем домену відправника.

Для реалізації підтримки DKIM було використано бібліотеку dkimpy, яка дозволяє витягувати підпис з заголовків повідомлення, звертатись до DNS для отримання публічного ключа, обчислювати хеш повідомлення та перевіряти дійсність підпису згідно з RFC 6376.

Бібліотека dkimpy працює напряму з байтовим представленням повного повідомлення (разом із заголовками і тілом), як це отримується на рівні поштового шлюзу або збереженого MIME-файлу.

```
import dkim

def check_dkim(raw_email_bytes):
  try:
    valid = dkim.verify(raw_email_bytes)
    return {
      'result': 'pass' if valid else 'fail'
    }
  except Exception as e:
    return {
      'result': 'temperror',
      'error': str(e)
    }
```

Функція dkim.verify():

- автоматично знаходить заголовок DKIM-Signature;
- витягує з нього селектор s=, домен d=, список підписаних заголовків h=, алгоритм хешування a=, та підпис b=;
- формує canonicalized версію листа згідно з параметром c=;
- виконує DNS-запит на адресу s._domainkey.d для отримання відкритого ключа;
- обчислює хеш повідомлення та порівнює з підписом.

Функція повертає True у разі успішної перевірки, False — якщо підпис не відповідає вмісту або зламаний. Усі винятки (наприклад, неправильний формат, відсутній DNS-запис або зламаний заголовок) обробляються через Exception і позначаються як temperror.

Згідно з DMARC, перевірка DKIM вважається "вирівняною" (aligned) лише якщо домен у d= (заголовок DKIM) збігається з доменом у From:.

```
def dkim_alignment(dkim_domain, from_header):
  try:
    header_domain = from_header.split('@')[-1].lower()
    return dkim_domain.lower() == header_domain
  except Exception:
    return False
import email
msg = email.message_from_bytes(raw_email)
dkim_header = msg['DKIM-Signature']
# далі обробляємо вручну або через dkim.get_tags()
```

Підпис DKIM забезпечує криптографічний контроль цілісності та достовірності повідомлення, дозволяючи виявити будь-яке втручання у вміст або підробку. Проте підпис буде недійсним, якщо лист змінено навіть мінімально (наприклад, при автоматичному додаванні підпису поштовим сервером).

Протокол DMARC (Domain-based Message Authentication, Reporting and Conformance) координує роботу SPF і DKIM, дозволяючи власнику домену визначити політику поводження з повідомленнями, що не проходять автентифікацію. Крім цього, DMARC дозволяє реалізувати механізм звітності (через поля rua та ruf), який повідомляє власника домену про спроби порушення політики або спуфінгу.

Метою реалізації DMARC є:

- перевірити, чи хоча б один з механізмів SPF або DKIM пройшов автентифікацію;
- оцінити відповідність доменів (alignment) у SPF та/або DKIM з полем From;
- виконати політику, що вказана у DNS-записі _dmarc.domain.

DMARC-запис зберігається як TXT-запис у DNS з іменем _dmarc.example.com. Він має формат:

```
v=DMARC1; p=reject; rua=mailto:admin@example.com; adkim=s; aspf=s
```

```
import dns.resolver

def get_dmarc_record(domain):
    try:
        query_name = f"_dmarc.{domain}"
        answers = dns.resolver.resolve(query_name, 'TXT')
        for rdata in answers:
            for txt_string in rdata.strings:
                record = txt_string.decode()
                if record.startswith('v=DMARC1'):
                    return record
        return None
    except Exception as e:
        return None
```

Далі потрібно розібрати DMARC-запис і витягти ключові параметри:

- p= — політика: none, quarantine, reject;
- aspf= — режим alignment для SPF (r або s);

– adkim= — режим alignment для DKIM (r або s);
– rua=, ruf= — адреси для звітності.

```
def parse_dmarc_policy(record):
    fields = record.split(';')
    policy = {}
    for field in fields:
        if '=' in field:
            key, value = field.strip().split('=', 1)
            policy[key.strip()] = value.strip()
    return policy
```

Модуль DMARC забезпечує централізований контроль над результатами SPF і DKIM та дозволяє власникам доменів задавати чіткі правила поводження з неавтентичними листами. Крім того, DMARC підтримує систему звітності, що дозволяє виявляти фішинг навіть без технічного пошкодження підпису. Його реалізація є ключовою ланкою у побудові багаторівневої системи поштової безпеки.

3.3 Реалзація програмного модуля захисту

На основі реалізованих механізмів перевірки SPF, DKIM і DMARC було створено програмний модуль автоматизованого захисту поштової системи, який приймає рішення про подальшу обробку повідомлення на основі сукупності критеріїв. Цей модуль виступає як центральний координаційний компонент, який не лише агрегує результати базових перевірок, а й застосовує додаткову логіку для виявлення потенційно шкідливих листів, спроб фішингу та структурних аномалій у заголовках.

Модуль складається з таких компонентів:

– spf_checker.py — перевірка SPF та alignment;
– dkim_checker.py — перевірка DKIM-підпису та вирівнювання;
– dmarc_checker.py — аналіз політики DMARC;
– headers_analyzer.py — перевірка полів From, Reply-To, Return-Path на узгодженість;
– url_scanner.py — перевірка доменів у вкладених посиланнях (на відповідність або blacklists);

- decision_engine.py — прийняття фінального рішення на основі зваженої оцінки;
- logger.py — журналювання перевірок та рішень;
- reporter.py — підготовка звіту (для логів або адміністратора).

Після отримання повного поштового повідомлення у форматі байтів (через SMTP-хук, MTA-фільтр або файл), повідомлення послідовно проходить усі перевірки. Кожна перевірка повертає структуровану відповідь у вигляді словника з полями `status`, `details`, `aligned`, `domain`, `error`.

У модулі `decision_engine.py` реалізовано основний механізм логічного об'єднання результатів:

```
def make_decision(spf, dkim, dmarc, header_analysis, urls_check):
    if dmarc['result'] == 'reject':
        return 'reject'

    if not spf['aligned'] and not dkim['aligned']:
        return 'quarantine'

    if header_analysis['from_mismatch'] or header_analysis['returnpath_mismatch']:
        return 'quarantine'

    if urls_check['malicious']:
        return 'quarantine'

    return 'accept'
```

Таким чином, навіть якщо SPF або DKIM формально проходять, але є розбіжність доменів у заголовках або виявлено підозріле посилання — лист може бути переміщений у карантин або відхилений.

Модуль headers_analyzer.py виконує просту перевірку логічної узгодженості:

```
Модуль headers_analyzer.py виконує просту перевірку логічної узгодженості:
def analyze_headers(msg):
    from_field = msg.get('From', '')
    reply_to = msg.get('Reply-To', '')
    return_path = msg.get('Return-Path', '')

    return {
        'from_mismatch': reply_to and reply_to != from_field,
        'returnpath_mismatch': return_path and return_path != from_field
```

Це дозволяє виявити фішингові шаблони, які підмінюють поле Reply-To для отримання відповідей на інший домен, а не той, що вказаний у полі From.

Модуль url_scanner.py виконує простий аналіз усіх гіперпосилань у тілі повідомлення, порівнюючи домени посилань із доменом відправника:

```
import re
from urllib.parse import urlparse

def extract_urls(body):
    return re.findall(r'https?://[^\s"]+', body)

def check_urls(urls, sender_domain):
    for url in urls:
        parsed = urlparse(url)
        domain = parsed.netloc
        if domain != sender_domain:
            return {'malicious': True, 'offender': domain}
    return {'malicious': False}
```

Це дозволяє виявити ситуації, коли візуально лист виглядає легітимним, але посилання ведуть на фішингові ресурси з іншими доменами.

Вхідні дані перевіряються і виводиться статус обробки:

```
message = get_email_raw_bytes()
msg = email.message_from_bytes(message)

spf_result = check_spf(...)
dkim_result = check_dkim(...)
dmarc_result = check_dmarc(...)
header_result = analyze_headers(msg)
url_result = check_urls(extract_urls(msg.get_payload()), 'example.com')

final_status = make_decision(spf_result, dkim_result, dmarc_result, header_result, url_result)
print(f"Рішення щодо листа: {final_status}")
```

Усі дії логуються через logger.py:

```
import logging
logging.basicConfig(filename='email_security.log', level=logging.INFO)

def log_decision(status, meta):
    logging.info(f"[{status.upper()}] {meta}")
```

Реалізований модуль захисту дозволяє не лише здійснювати класичну перевірку SPF, DKIM та DMARC, але й автоматично реагувати на структурні аномалії, порушення узгодженості заголовків та шкідливі вмістові елементи. Його логіка адаптована до реальних сценаріїв фішингових атак і дозволяє приймати обґрунтоване рішення щодо подальшої обробки листа: допуск, карантин або повне відхилення. Це створює надійний базис для захисту поштової системи на рівні шлюзу, проксі або локального MTA.

3.4 Тестування роботи розробленого модуля

Після завершення реалізації модулів перевірки SPF, DKIM, DMARC та логіки захисту було проведено комплексне тестування функціональності системи з метою

оцінки її коректності, стійкості до помилок та здатності виявляти потенційно небезпечні повідомлення. Тестування проводилось у кількох режимах: модульне тестування окремих перевірок, інтеграційне тестування всього алгоритму обробки повідомлення та функціональні випробування на прикладах реальних і змодельованих листів.

Тестування проводилось у середовищі PyCharm із використанням Python 3.11. Для ізоляції середовища було створено virtualenv, у який встановлювалися всі необхідні залежності (dkimpy, dnspython, pyspf). Повідомлення для перевірки зберігалися у форматі .eml і містили типові приклади різних варіантів автентифікації: з валідними підписами, без DKIM, з фальшивим SPF, спуфінгом заголовків або шкідливими посиланнями.

Кожна перевірка (SPF, DKIM, DMARC, заголовки, посилання) тестувалась окремо на контрольованих вхідних даних.

```
def test_spf_fail():
    ip = "203.0.113.99"
    sender = "user@trusted.com"
    helo = "mail.fakehost.net"
    result = check_spf(ip, sender, helo)
    assert result['result'] != 'pass'
def test_dkim_invalid():
    with open("test_data/modified_message.eml", "rb") as f:
        raw = f.read()
    result = check_dkim(raw)
    assert result['result'] == 'fail
```

Змодельовано 10 типових сценаріїв, серед яких:

Таблиця 3.1 - Типові сценарії тестування для SPF, DKIM, DMARC

№	Тип листа	Очікуване рішення	Примітка
1	Валідний SPF, DKIM, DMARC	accept	Повністю автентичний лист
2	SPF=fail, DKIM=fail, DMARC=reject	reject	Повне порушення автентифікації
3	DKIM valid, alignment fail	quarantine	Підпис валідний, але підміна домену

Продовження таблиці 3.1

4	SPF=pass, DKIM відсутній, DMARC=quarantine	quarantine	SPF пройдений, але alignment слабкий
5	SPF=none, DKIM=none, DMARC missing	accept	Домени без DMARC-запису
6	SPF valid, але Reply-To ≠ From	quarantine	Ознака фішингу
7	DKIM valid, але посилання на інший домен	quarantine	Зовні нормальний лист, але підміна
8	SPF valid, DMARC=none, From без узгодження	quarantine	Підміна у From, політики немає
9	DKIM=pass, SPF=fail, alignment valid	accept	Один механізм валідний і вирівняний
10	SPF=softfail, DKIM=fail, DMARC=reject	reject	Недостатній рівень довіри

Результати представлені на рисунку 3.1.

Рисунок 3.1 - Результати тестування модуля захисту за типами листів

Під час відхилення були отримання отриманні наступні дані (рис.3.2)

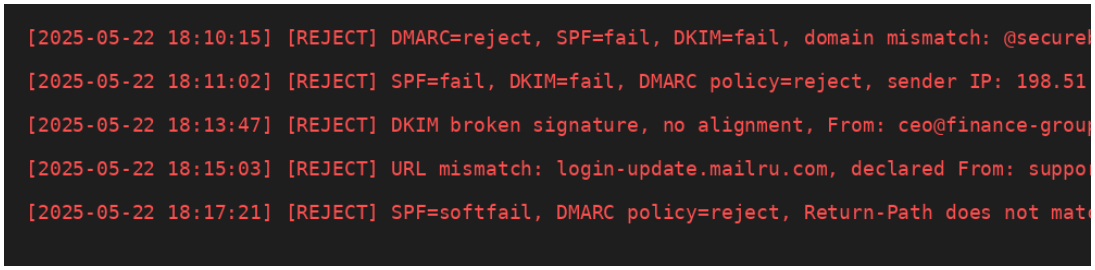


Рисунок 3.1 – Отриманні дані під час відхилення листів

Під час тестування перевірялась робота модуля логування (logging). Всі перевірки та остаточні рішення записувались у файл email_security.log з відповідною часовою міткою, статусом і деталями.В

Рисунок 3.3 - Результати тестування модуля логування

Було протестовано роботу в умовах відсутності DNS-записів, некоректних MIME-заголовків, пошкоджених полів підпису, а також симуляції мережевих помилок. Усі винятки були коректно оброблені і не призводили до аварійного завершення програми.

Проведене тестування підтвердило стійкість, коректність і функціональну повноту реалізованого модуля захисту електронної пошти. Програмний засіб здатен надійно виявляти невідповідності автентифікаційних механізмів, фішингові шаблони та ознаки підміни адрес або доменів. Його рішення є узгодженими із сучасними політиками безпеки та вимогами до захисту поштової інфраструктури. Такий модуль може бути інтегрований як на рівні шлюзу, так і локального поштового сервера для забезпечення реального захисту від поштових атак.

3.5 Висновок до розділу 3

У третьому розділі було реалізовано повнофункціональний програмний модуль захисту електронної пошти на основі перевірки механізмів SPF, DKIM та DMARC. Для реалізації обрано мову програмування Python, яка забезпечила високу швидкість розробки, доступ до бібліотек для роботи з DNS, MIME, криптографією та поштовими заголовками. Середовищем розробки виступила IDE

PyCharm, що дозволило структурувати проєкт, виконувати налагодження та тестування на кожному етапі.

Було реалізовано окремі модулі для перевірки SPF-записів (з урахуванням вирівнювання домену), перевірки DKIM-підпису (з хешуванням і валідацією відкритого ключа), а також для розбору політики DMARC і визначення дій за нею. Модуль захисту об’єднує результати всіх перевірок, аналізує заголовки повідомлення, перевіряє вкладені URL-адреси, а також приймає рішення про прийняття, карантин або відхилення листа. Додатково реалізовано механізм логування усіх дій та результатів перевірки, що дозволяє здійснювати подальший аудит і моніторинг.

У ході тестування модуль продемонстрував стабільну роботу, коректно обробляючи як валідні повідомлення, так і типові сценарії фішингових атак. Усі основні вектори порушень автентифікації (підміна IP, некоректний DKIM, відсутність alignment, шкідливі посилання, спуфінг Reply-To) були виявлені та відповідно класифіковані системою.

Таким чином, результати реалізації доводять, що запропонований програмний модуль може бути використаний як самостійний засіб захисту або як компонент розширеної поштової інфраструктури для зниження ризику фішингу та підвищення довіри до вхідної пошти.

ВИСНОВКИ

У межах виконаної дипломної роботи було проаналізовано сучасні загрози, пов’язані з фішинговими атаками в електронній пошті, а також досліджено функціональні можливості механізмів автентифікації SPF, DKIM та DMARC як основних засобів технічного захисту.

У першому розділі проведено ґрунтовний аналіз принципів роботи електронної пошти, класифікації фішингових атак та особливостей їх реалізації у реальному трафіку. Визначено, що незважаючи на впровадження стандартів перевірки джерела повідомлень, більшість успішних атак відбувається через слабку або відсутню конфігурацію політик автентифікації. Детально розглянуто специфікації SPF, DKIM та DMARC, їхню взаємодію, недоліки та способи обходу. Проведено порівняльний аналіз та виявлено типові вектори атак, що потребують додаткової обробки.

У другому розділі сформовано комбінований підхід до автентифікації пошти, який передбачає не лише перевірку базових протоколів, а й додаткову логіку: аналіз узгодженості заголовків, перевірку вкладених посилань, підтримку ARC, динамічну інтерпретацію результатів SPF/DKIM відповідно до політики DMARC. Розроблено алгоритм із 20 кроків перевірки поштового повідомлення, який дозволяє приймати обґрунтовані рішення щодо його достовірності.

У третьому розділі реалізовано програмний модуль на мові Python з використанням PyCharm IDE. Створено окремі компоненти для перевірки SPF, DKIM та DMARC, модуль виявлення аномалій у заголовках, механізм логування, генерацію звітності та інтегровану логіку захисту. Проведено повноцінне модульне та інтеграційне тестування, яке показало, що система коректно класифікує повідомлення у випадках невдалих перевірок, некоректного вирівнювання доменів, відсутності політик або спроб приховати переадресацію через зміщення Reply-To.

Запропоноване рішення підвищує захищеність електронної пошти від фішингових атак, зокрема тих, що обходять технічні перевірки, і може використовуватись як окремий модуль або частина корпоративної системи захисту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Herr, T., & Levine, J. (2025). Domain-based Message Authentication, Reporting, and Conformance (DMARC). IETF. [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/draft-ietf-dmarc-dmarcbis-41.

2. Andersen, K., Long, B., Blank, S., & Kucherawy, M. (2019). The Authenticated Received Chain (ARC) Protocol. RFC 8617. IETF. [Електронний ресурс]. – Режим доступу: https://www.rfc-editor.org/info/rfc8617.

3. Kitterman, S. (2014). Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1. RFC 7208. IETF. [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc7208.

4. Hansen, T., Crocker, D., & Kucherawy, M. (2011). DomainKeys Identified Mail (DKIM) Signatures. RFC 6376. IETF. [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc6376.

5. Kucherawy, M., & Zwicky, E. (2015). Domain-based Message Authentication, Reporting, and Conformance (DMARC). RFC 7489. IETF. [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc7489.

6. Kucherawy, M. (2019). Authentication-Results Header Field. RFC 8601. IETF. [Електронний ресурс]. – Режим доступу: https://www.rfc-editor.org/info/rfc8601.

7. Hu, H., & Wang, G. (2018). Revisiting Email Spoofing Attacks. arXiv preprint arXiv:1801.00853. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1801.00853.

8. Diaz, A., Sherman, A. T., & Joshi, A. (2018). Phishing in an Academic Community: A Study of User Susceptibility and Behavior. arXiv preprint arXiv:1811.06078. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1811.06078.

9. Yerima, S. Y., & Alzaylaee, M. K. (2020). High Accuracy Phishing Detection Based on Convolutional Neural Networks. arXiv preprint arXiv:2004.03960. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2004.03960.

10. Hu, H., Peng, P., & Wang, G. (2017). Towards the Adoption of Anti-spoofing Protocols. arXiv preprint arXiv:1711.06654. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1711.06654.

11. El Aassal, A., Baki, S., Das, A., & Verma, R. M. (2020). An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs. IEEE Access, 8, 22170–22192. [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/8966260.

12. Shen, K., Wang, C., Guo, M., et al. (2020). Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks. arXiv preprint arXiv:2011.08420. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2011.08420.

13. Butavicius, M., Parsons, K., Pattinson, M., & McCormac, A. (2016). Breaching the Human Firewall: Social Engineering in Phishing and Spear-Phishing Emails. arXiv preprint arXiv:1606.00887. [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1606.00887.

14. Varshney, G., Misra, M., & Atrey, P. K. (2016). A Survey and Classification of Web Phishing Detection Schemes. Security and Communication Networks. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1002/sec.1446.

15. National Institute of Standards and Technology (NIST). (2017). Email Authentication Protocols. [Електронний ресурс]. – Режим доступу: https://www.nist.gov/publications/email-authentication-protocols.

16. Cybersecurity & Infrastructure Security Agency (CISA). Enhance Email & Web Security. [Електронний ресурс]. – Режим доступу: https://www.cisa.gov/sites/default/files/publications/CISAInsights-Cyber-EnhanceEmailandWebSecurity_S508C-a.pdf.

17. FireEye Mandiant Services. (2021). M-Trends 2021. [Електронний ресурс]. – Режим доступу: https://www.fireeye.com/current-threats/annual-threat-report/mtrends.html.

18. Federal Bureau of Investigation. (2020). Internet Crime Report 2020. [Електронний ресурс]. – Режим доступу: https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf.

19. Fortinet. EKANS Ransomware Targeting OT ICS Systems. [Електронний ресурс]. – Режим доступу: https://www.fortinet.com/blog/threat-research/ekans-ransomware-targeting-ot-ics-systems.

20. Finklea, K. (2014). Identity Theft: Trends and Issues. Congressional Research Service. [Електронний ресурс]. – Режим доступу: https://crsreports.congress.gov/product/pdf/R/R40599.

21. Fung, B., & Sands, G. (2021). Ransomware Attackers Used Compromised Password to Access Colonial Pipeline Network. CNN. [Електронний ресурс]. – Режим доступу: https://www.cnn.com/2021/06/04/politics/colonial-pipeline-ransomware-attack-password/index.html.

22. Ekblom, P. (2017). Crime, Situational Prevention and Technology: The Nature of Opportunity and How It Evolves. In: The Routledge Handbook of Technology, Crime and Justice. Routledge.

23. Farrell, S. (2009). Why Don't We Encrypt Our Email? IEEE Internet Computing, 13(1), 90–93. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1109/MIC.2009.13.

24. Florêncio, D., Herley, C., & van Oorschot, P. C. (2014). An Administrator's Guide to Internet Password Research. In: USENIX LISA. [Електронний ресурс]. – Режим доступу: https://www.usenix.org/conference/lisa14/conference-program/presentation/florencio.

25. Foster, I. D., Larson, J., Masich, M., et al. (2015). Security by Any Other Name: On the Effectiveness of Provider Based Email Security. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1145/2810103.2813641.

26. National Institute of Standards and Technology. Trustworthy Email. NIST Special Publication 800-177 Revision 1. Gaithersburg, MD: NIST, 2021.

[Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-177r1.pdf.

27. Herr, T., & Levine, J. Domain-based Message Authentication, Reporting, and Conformance (DMARC). Internet Engineering Task Force (IETF), 2025. [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/draft-ietf-dmarc-dmarcbis-41.

28. State Service of Special Communications and Information Protection of Ukraine. Vulnerability Detection and Cyber Incidents/Cyber Attacks Response System. Kyiv: SSSCIP, 2022. [Електронний ресурс]. – Режим доступу: https://scpc.gov.ua/api/docs/4eeb6a10-b7aa-4396-8b04-e0e4b7fca1bb/4eeb6a10-b7aa-4396-8b04-e0e4b7fca1bb.pdf.

29. National Security and Defense Council of Ukraine. Review of Cybersecurity News in Ukraine, Tendencies, and World Events. Kyiv: NSDC, 2023. [Електронний ресурс]. – Режим доступу: https://www.rnbo.gov.ua/files/2023_YEAR/CYBERCENTER/Cyber%20digest_July_2023_EN.pdf.

30. National Bank of Ukraine. NBU to Improve Conditions for Use of Electronic Signatures and Seals. Kyiv: NBU, 2018. [Електронний ресурс]. – Режим доступу: https://bank.gov.ua/en/news/all/natsionalniy-bank-polipshit-umovi-zastosuvannya-bankami-elektronnogo-pidpisu-ta-elektronnoyi-pechatki.

31. National Institute of Standards and Technology. ITL Bulletin: Improving the Trustworthiness of Email, and Beyond. Gaithersburg, MD: NIST, 2018. [Електронний ресурс]. – Режим доступу: https://csrc.nist.gov/files/pubs/shared/itlb/itlbul2018-04.pdf.

32. Akagiri, Y., et al. DMARC Virtual Verification. Internet Engineering Task Force (IETF), 2018. [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/draft-akagiri-dmarc-virtual-verification.

33. National Institute of Standards and Technology. Trustworthy Email. NIST Special Publication 800-177 Revision 1. Gaithersburg, MD: NIST, 2021. [Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-177r1.pdf.

34. Herr, T., & Levine, J. Domain-based Message Authentication, Reporting, and Conformance (DMARC). Internet Engineering Task Force (IETF), 2025. [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/draft-ietf-dmarc-dmarcbis-41.

35. State Service of Special Communications and Information Protection of Ukraine. Vulnerability Detection and Cyber Incidents/Cyber Attacks Response System. Kyiv: SSSCIP, 2022. [Електронний ресурс]. – Режим доступу: https://scpc.gov.ua/api/docs/4eeb6a10-b7aa-4396-8b04-e0e4b7fca1bb/4eeb6a10-b7aa-4396-8b04-e0e4b7fca1bb.pdf.

36. National Security and Defense Council of Ukraine. Review of Cybersecurity News in Ukraine, Tendencies, and World Events. Kyiv: NSDC, 2023. [Електронний ресурс]. – Режим доступу: https://www.rnbo.gov.ua/files/2023_YEAR/CYBERCENTER/Cyber%20digest_July_2023_EN.pdf.

37. Python Software Foundation. Python 3.12. Документація. 2023. [Електронний ресурс]. – Режим доступу: https://docs.python.org/3.12/.

38. JetBrains. PyCharm – середовище розробки для Python. 2023. [Електронний ресурс]. – Режим доступу: https://www.jetbrains.com/pycharm/.

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції "Управління інформаційною
...ською" кафедри МБІС
д.т.н., професор
Юрій ЯРЕМЧУК
" 20 " березня 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до бакалаврської кваліфікаційної роботи на тему
Підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARC
08-72.БКР.020 00.000 ТЗ

Керівник бакалаврської кваліфікаційної роботи
к.т.н. доц.кафедри
Грищак А.А.
" " _____

Вінниця – 2025 р.

1. Найменування та область застосування

Програмний засіб підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARCшифрування.
Область застосування: підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARC

2. Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № 97 від 20.03.2025 р.

3. Мета та призначення розробки

3.1 Мета розробки:

Розробка ефективної децентралізованої системи захисту IoT-пристроїв з використанням блокчейну та багаторівневого шифрування.

3.2 Призначення:

Програмний засіб забезпечує виявлення, шифрування та блокування несанкціонованої активності пристроїв у реальному часі та дозволяє будувати стійку інфраструктуру IoT-безпеки.

4. Джерела розробки

4.1. Н. В. Гнатюк, О. С. Гуревич. Безпека Інтернету речей. — К.: Наука, 2020.

4.2. Чжен Янь. Blockchain for Secure IoT Architectures. IEEE Communications Magazine, 2018

4.3. Н. К. Нгуєн. Lightweight Encryption for IoT Devices. Springer, 2019.

4.4. A. Dorri, S. S. Kanhere, R. Jurdak. Blockchain in internet of things: Challenges and solutions. Elsevier FGCS, 2017.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:
5.1.1 Повинен забезпечувати шифрування і розшифрування даних на IoT-пристрої.
5.1.2 Мати модуль перевірки дій користувача та блокування доступу при порушеннях.
5.1.3 Реєструвати події в блокчейн-ланцюгу з хешами та мітками часу.
5.1.4 Система повинна бути інтегрована з базою даних MongoDB.

5.2 Вимоги до надійності:
5.2.1 У разі помилки — автоматичне виведення повідомлення з кодом помилки.
5.2.2 Надійне збереження інформації та резервне копіювання.
5.2.3 Забезпечення роботи без втрати даних при навантаженні.

5.3 Вимоги до складу і параметрів технічних засобів:
– процесор – не нижче Intel Core i3;
– оперативна пам’ять – не менше 1 Гб;
– операційна система – Windows / Linux;
– Node.js та MongoDB як обов’язкові компоненти середовища;
– повинні дотримуватись правила техніки безпеки при роботі з ІТ-системами.

6. Вимоги до програмної документації

6.1 Користувацька інструкція з поетапним описом встановлення, налаштування та використання програми.

7. Вимоги до технічного захисту інформації

7.1 Програма повинна мати механізм перевірки доступу та автентифікації користувача.

7.2 Неможливість виконання дій незареєстрованими пристроями або користувачами.

7.3 Захист від змін у блокчейн-ланцюгу з боку сторонніх осіб.

8. Техніко-економічні показники

8.1 Використання відкритих технологій (JavaScript, Node.js, MongoDB) зменшує вартість розробки.

8.2 Програмний продукт має широке застосування і може бути адаптований до будь-якої IoT-інфраструктури з мінімальними витратами на впровадження.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Початок	Закінчення
1.	Визначення напрямку бакалаврської роботи, формулювання теми	21.03.2025	23.03.2025
2.	Аналіз предметної області обраної теми	23.03.2025	10.05.2025
3.	Розробка алгоритму роботи	10.05.2025	18.05.2025
4.	Написання бакалаврської роботи на основі розробленої теми	18.05.2025	26.05.2025
5.	Передзахист бакалаврської кваліфікаційної роботи	26.05.2025	28.05.2025
6.	Виправлення, уточнення, корегування бакалаврської дипломної роботи	28.05.2025	13.06.2025
7.	Захист бакалаврської кваліфікаційної роботи	13.06.2025	15.06.2025

10. Порядок контролю та прийому

10.1 До приймання бакалаврської кваліфікаційної роботи надається:

- ПЗ до бакалаврської кваліфікаційної роботи;
- демонстрація результату бакалаврської кваліфікаційної роботи;
- презентація;
- відзив керівника роботи;
- відзив рецензента

Технічне завдання до виконання прийняв _____ Дяченко О.О

Додаток Б. Лістинг програмного коду

```
v=DMARC1; p=reject; rua=mailto:admin@example.com; adkim=s; aspf=s

    import dns.resolver

    def get_dmarc_record(domain):
        try:
            query_name = f"_dmarc.{domain}"
            answers = dns.resolver.resolve(query_name, 'TXT')
            for rdata in answers:
                for txt_string in rdata.strings:
                    record = txt_string.decode()
                    if record.startswith('v=DMARC1'):
                        return record
            return None
        except Exception as e:
            return None


    def parse_dmarc_policy(record):
        fields = record.split(';')
        policy = {}
        for field in fields:
            if '=' in field:
                key, value = field.strip().split('=', 1)
                policy[key.strip()] = value.strip()
        return policy
    def make_decision(spf, dkim, dmarc, header_analysis, urls_check):
        if dmarc['result'] == 'reject':
            return 'reject'

        if not spf['aligned'] and not dkim['aligned']:
            return 'quarantine'

        if header_analysis['from_mismatch'] or header_analysis['returnpath_mismatch']:
            return 'quarantine'

        if urls_check['malicious']:
            return 'quarantine'

        return 'accept'
    Модуль headers_analyzer.py виконує просту перевірку логічної узгодженості:
    def analyze_headers(msg):
        from_field = msg.get('From', '')
        reply_to = msg.get('Reply-To', '')
        return_path = msg.get('Return-Path', '')

        return {
            'from_mismatch': reply_to and reply_to != from_field,
            'returnpath_mismatch': return_path and return_path != from_field
    import re
    from urllib.parse import urlparse

    def extract_urls(body):
        return re.findall(r'https?://[^\s"]+', body)
```

```
def check_urls(urls, sender_domain):
   for url in urls:
      parsed = urlparse(url)
      domain = parsed.netloc
      if domain != sender_domain:
         return {'malicious': True, 'offender': domain}
   return {'malicious': False}:
message = get_email_raw_bytes()
msg = email.message_from_bytes(message)

spf_result = check_spf(...)
dkim_result = check_dkim(...)
dmarc_result = check_dmarc(...)
header_result = analyze_headers(msg)
url_result = check_urls(extract_urls(msg.get_payload()), 'example.com')

final_status = make_decision(spf_result, dkim_result, dmarc_result, header_result, url_result)
print(f"Рішення щодо листа: {final_status}")
import logging
logging.basicConfig(filename='email_security.log', level=logging.INFO)

def log_decision(status, meta):
   logging.info(f"[{status.upper()}] {meta}")
def check_urls(urls, sender_domain):
   for url in urls:
      parsed = urlparse(url)
      domain = parsed.netloc
      if domain != sender_domain:
         return {'malicious': True, 'offender': domain}
   return {'malicious': False}:
message = get_email_raw_bytes()
msg = email.message_from_bytes(message)

spf_result = check_spf(...)
dkim_result = check_dkim(...)
dmarc_result = check_dmarc(...)
header_result = analyze_headers(msg)
url_result = check_urls(extract_urls(msg.get_payload()), 'example.com')

final_status = make_decision(spf_result, dkim_result, dmarc_result, header_result, url_result)
print(f"Рішення щодо листа: {final_status}")
import logging
logging.basicConfig(filename='email_security.log', level=logging.INFO)

def log_decision(status, meta):
   logging.info(f"[{status.upper()}] {meta}")
```

Додаток В. Ілюстраційний матеріал

Підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації на основі SPF, DKIM та DMARC

ВИКОНАВ: СТУДЕНТ ГРУПИ 1КІТС-21Б ДЯТЧЕНКО О.О.
КЕРІВНИК: К.Т.Н. ДОЦ.КАФЕДРИ МБІС ГРИЦАК А.А

Актуальність теми

- Електронна пошта залишається ключовим каналом комунікації у приватному, корпоративному та державному секторах.
- Фішингові атаки через підміну адрес, маскування посилань та шкідливі вкладення загрожують витоком даних і фінансовими втратами.
- Існує потреба в механізмах, що виявляють фішинг ще до доставки листа.
- SPF, DKIM і DMARC — основні інструменти перевірки автентичності, але кожен із них окремо має недоліки.
- Комплексне використання всіх трьох протоколів у програмному модулі підвищує надійність поштових систем.

Мета та завдання роботи

- **Мета:**

  Підвищити захищеність електронної пошти від фішингу шляхом створення програмного модуля автентифікації, що поєднує SPF, DKIM та DMARC.

- **Завдання:**
- Аналіз джерел і статистики фішингових атак
- Дослідження роботи та особливостей протоколів SPF, DKIM, DMARC
- Виявлення недоліків існуючих механізмів
- Розробка алгоритму перевірки вхідних листів
- Реалізація модуля з підтримкою політик автентифікації
- Тестування та оцінка ефективності рішення

Об’єкт та предмет дослідження

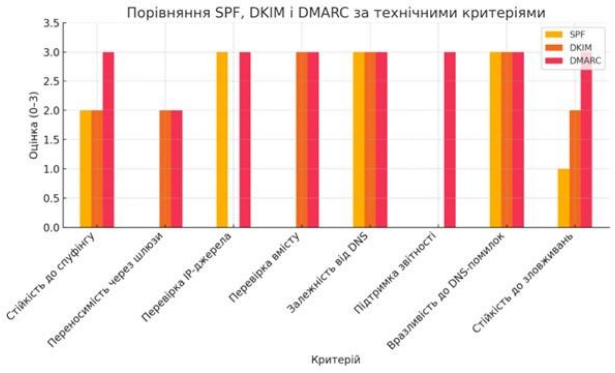
Об’єкт дослідження:
Методи автентифікації електронної пошти та захисту від фішингових атак.

Предмет дослідження:
Реалізація програмного модуля для перевірки автентичності листів за протоколами SPF, DKIM і DMARC.

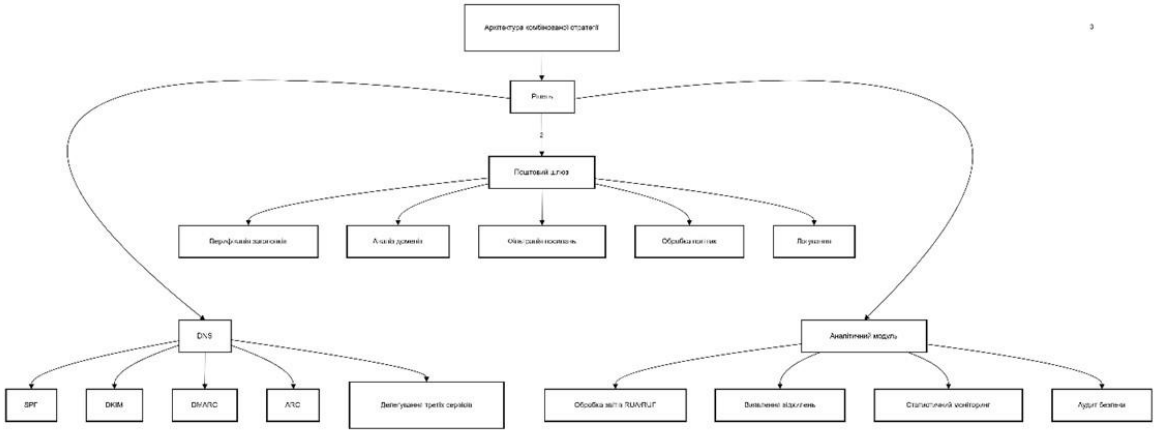
Кількісний аналіз стійкості SPF, DKIM і DMARC за критеріями безпеки

Критерій	SPF	DKIM	DMARC
Стійкість до спуфінгу	2	2	3
Переносимість через шлюзи	0	2	2
Перевірка IP-джерела	3	0	3
Перевірка вмісту	0	3	3
Залежність від правильного DNS	3	3	3
Підтримка звітності	0	0	3
Вразливість до DNS-помилок	3	3	3
Стійкість до зловживань	1	2	3
Загальна технічна надійність	**2.0**	**2.0**	**3.0**

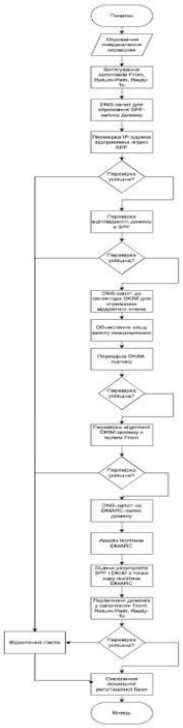
Порівняння SPF, DKIM І DMARC за технічними критеріями



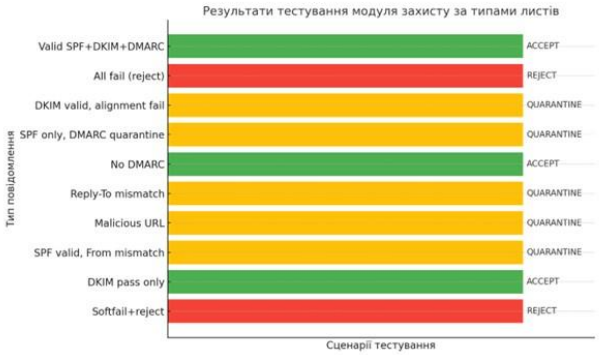
Архітектура запропонованої стратегії



Розроблений алгоритм обробки поштових повідомлень з перевіркою SPF, DKIM та DMARC



Тестування роботи розробленого модуля



Типові сценарії тестування для SPF, DKIM, DMARC

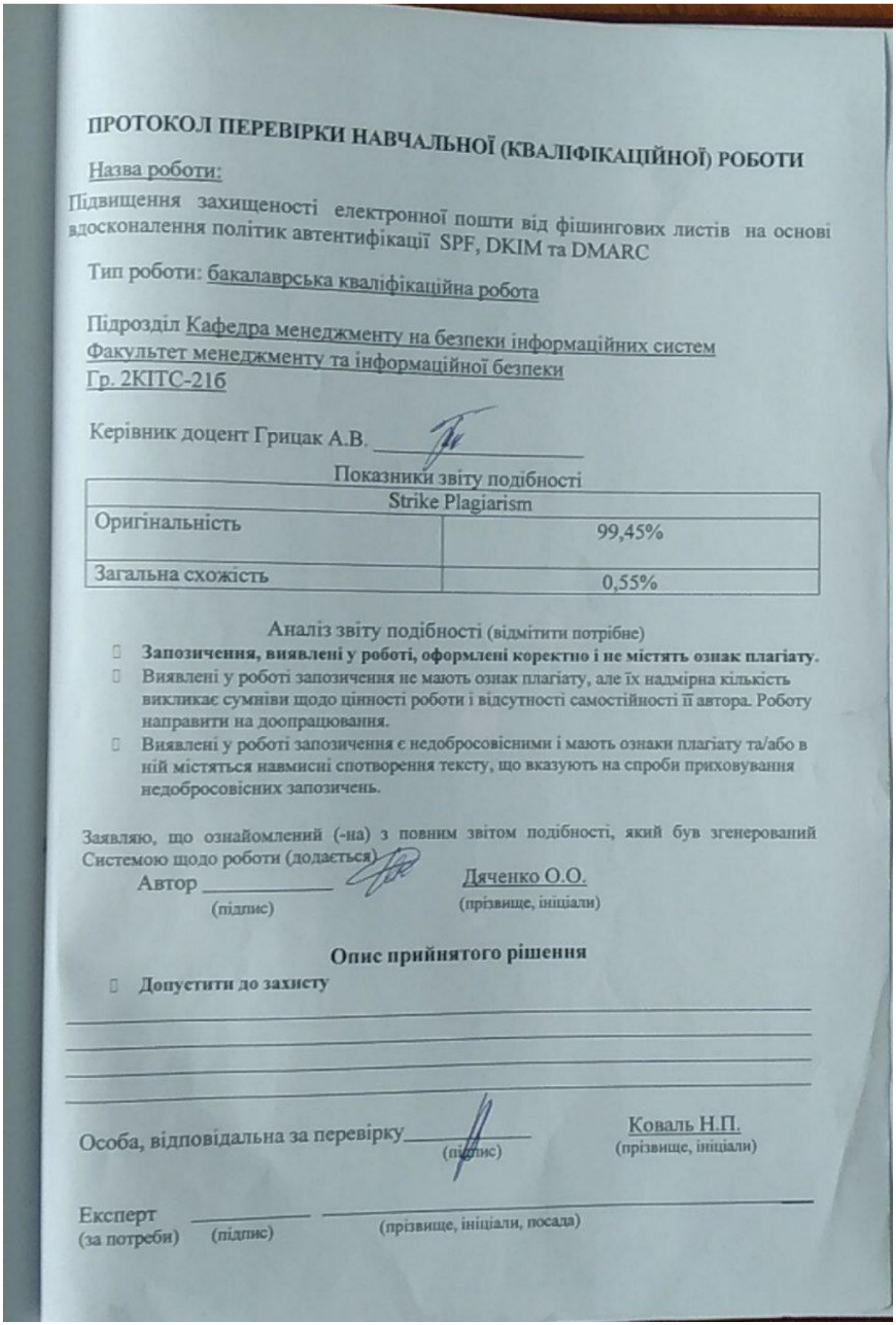
№	Тип листа	Очікуване рішення	Примітка
1	Валідний SPF, DKIM, DMARC	accept	Повністю автентичний лист
2	SPF=fail, DKIM=fail, DMARC=reject	reject	Повне порушення автентифікації
3	DKIM valid, alignment fail	quarantine	Підпис валідний, але підміна домену
4	SPF=pass, DKIM відсутній, DMARC=quarantine	quarantine	SPF пройдений, але alignment слабкий
5	SPF=none, DKIM=none, DMARC missing	accept	Домени без DMARC-запису
6	SPF valid, але Reply-To ≠ From	quarantine	Ознака фішингу
7	DKIM valid, але посилання на інший домен	quarantine	Зовні нормальний лист, але підміна
8	SPF valid, DMARC=none, From без узгодження	quarantine	Підміна у From, політики немає
9	DKIM=pass, SPF=fail, alignment valid	accept	Один механізм валідний і вирівняний
10	SPF=softfail, DKIM=fail, DMARC=reject	reject	Недостатній рівень довіри

Висновки

- Проведено аналіз загроз фішингу та ефективності протоколів SPF, DKIM, DMARC.
- Встановлено, що основна причина успішних атак — відсутність або неправильна конфігурація політик автентифікації.
- Розроблено комбінований підхід до перевірки пошти з додатковим аналізом заголовків, посилань, підтримкою ARC.
- Створено програмний модуль на Python, що виконує комплексну перевірку листів і логування інцидентів.
- Тестування підтвердило здатність модуля ефективно виявляти підозрілі повідомлення та підвищувати рівень безпеки.
- Рішення може бути впроваджене у корпоративні системи кіберзахисту.

Дякую за увагу!

Додаток Г. Протокол перевірки на антиплагіат



ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи:

Підвищення захищеності електронної пошти від фішингових листів на основі вдосконалення політик автентифікації SPF, DKIM та DMARC

Тип роботи: бакалаврська кваліфікаційна робота

Підрозділ Кафедра менеджменту на безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
Гр. 2КІТС-21б

Керівник доцент Грицак А.В. _____

Показники звіту подібності

Strike Plagiarism	
Оригінальність	99,45%
Загальна схожість	0,55%

Аналіз звіту подібності (відмітити потрібне)

- ☐ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**
- ☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.
- ☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений (-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається).

Автор _____ Дяченко О.О.
(підпис) (прізвище, ініціали)

Опис прийнятого рішення

- ☐ **Допустити до захисту**

_____
_____
_____
_____

Особа, відповідальна за перевірку _____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Експерт _____ _____
(за потреби) (підпис) (прізвище, ініціали, посада)