Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка та розгортання системи моніторингу DNS записів з використанням платформи Airflow

Виконав: студент IV курсу, групи СН-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

(підпис) Блащишин Д.І. (прізвище та ініціали)

Керівник (підпис) Бревус Г.Б. (прізвище та ініціали)

Нормоконтроль (підпис) Шимчук Г.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
« » 20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Блащишин Дмитро Ігорович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та розгортання системи моніторингу DNS записів з використанням платформи Airflow

Керівник роботи Бревус Галина Богданівна, асистент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*07*» *травня* 20*25* року № *4/7-444*

2. Термін подання студентом завершеної роботи 23 червня 2025 р.

3. Вихідні дані до роботи Літературні та інтернет джерела

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1.Огляд DNS та необхідність його моніторингу. Аналіз існуючих рішень моніторингу. Висновок. 2.Проектування системи моніторингу DNS-записів. Реалізація ETL-процесу. Автоматизація перевірки змін у DNS-записах. Висновок. 3.Налаштування середовища для роботи. Розробка DAG для моніторингу DNS. Тестування. Реальні сценарії використання та реагування. Висновок. 4. Забезпечення безпеки та охорони праці. Захист персональних даних у процесі моніторингу. Висновок. Перелік джерел. Додаток А. Додаток Б.

5. Перелік графічного матеріалу:
1. Ієрархічна система доменних імен DNS. 2. Головна сторінка сервісу Apache Airflow. 3. Характеристики онлайн-сервісів моніторингу DNS. 4. Характеристики спеціалізованих систем моніторингу інфраструктури. 5. Характристики платформ класу Thream Intelligence для моніторингу DNS-записів.
6. Характеристики платформ автоматизації для створення нестандартних рішень.
7. Оркестровані графи завдань DAG. 8. Архітектура системи моніторингу DNS-записів на базі Apache Airflow. 9. Схема ETL – процесу збору та збереження даних. 10. ER-діаграма бази даних моніторингу DNS. 11. Завершення Ініціалізації середовища Docker Compose з компонентами Apache Airflow. 12. Консольне повідомлення про успішний запуск середовища Apache Airflow за допомогою Docker Compose. 13. Сторінка входу у веб-інтерфейс Apache Airflow. 14. DAG у списку Apache Airflow. 15. Виконання DAG. 16. Лог задачі DAG з обробкою помилки NXDOMAIN під час запиту до неіснуючого домену.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці		02.06.2025	05.06.2025

7. Дата видачі завдання 27 січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	27.01.2025	*Виконано*
2.	Підбір та опрацювання літературних джерел по темі кваліфікаційної роботи	08.05.2025-15.05.2025	*Виконано*
3.	Виконання дослідження та розгортання системи		
	моніторингу DNS записів	16.05.2025-22.05.2025	*Виконано*
4.	Оформлення розділу «Аналіз предметної області та		
	Постановка завдань»	23.05.2025-01.06.2025	*Виконано*
5.	Оформлення розділу «Проектування системи моніто-		
	рингу DNS-записів»	23.05.2025-01.06.2025	*Виконано*
6.	Оформлення розділу «Реалізація та тестування		
	системи»	23.05.2025-01.06.2025	*Виконано*
7.	Виконання завдання до підрозділу «Забезпечення безпеки та охорони праці»	02.06.2025-05.06.2025	*Виконано*
9.	Оформлення кваліфікаційної роботи	06.06.2025-10.06.2025	*Виконано*
10.	Нормоконтроль	11.06.2025-13.06.2025	*Виконано*
11.	Перевірка на плагіат	14.06.2025	*Виконано*
12.	Попередній захист кваліфікаційної роботи	15.06.2025	*Виконано*
13.	Захист кваліфікаційної роботи	24.06.2025	

Студент _____ Блащишин Дмитро Ігорович
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка та розгортання системи моніторингу DNS-записів з використанням платформи Airflow // Кваліфікаційна робота освітнього рівня «Бакалавр» // Блащишин Дмитро Ігорович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-41 // Тернопіль, 2025 // С. 62, рис. – 12, табл. – 4, додат. – 2, бібліогр. – 35.

Ключові слова: dns-записи, моніторинг, apache airflow, автоматизація, інформаційна безпека, система сповіщень, docker, dag

Кваліфікаційна робота присвячена дослідженню питань автоматизації процесу моніторингу DNS-записів з метою підвищення рівня інформаційної безпеки та стабільності функціонування мережевих сервісів. У роботі обґрунтовано доцільність створення нестандартного рішення з використанням інструментів оркестрації, що забезпечують гнучкість та масштабованість перевірок.

У першому розділі описано принципи роботи системи доменних імен, актуальні ризики для DNS-інфраструктури та сучасні інструменти моніторингу. Обґрунтовано вибір Apache Airflow для автоматизації перевірок.

У другому розділі подано архітектуру системи моніторингу, описано її компоненти, базу даних та логіку ETL-процесу.

У третьому розділі розглянуто розгортання середовища за допомогою Docker, реалізацію DAG-файлів, результати тестування та приклади практичного застосування системи.

Об'єкт дослідження: процес моніторингу змін у системі доменних імен.

Предмет дослідження: програмна реалізація автоматизованої системи моніторингу DNS-записів на базі Apache Airflow.

ANNOTATION

Development and Deployment of a DNS Records Monitoring System Using the Airflow Platform // Bachelor's qualification work // Blashchyshyn Dmytro Igorovich // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, Group SN-41 // Ternopil, 2025 // p. 62, fig. – 12, tabl. – 4, app. – 2, bibl. – 35.

Keywords: dns records, monitoring, apache airflow, automation, information security, notification system, docker, dag

This qualification work is dedicated to the study of automating the DNS records monitoring process in order to enhance information security and ensure the stable operation of network services. The work substantiates the feasibility of developing a custom solution using orchestration tools that provide flexibility and scalability in verification processes.

The first chapter describes the principles of the Domain Name System (DNS), current risks to DNS infrastructure, and modern monitoring tools. The choice of Apache Airflow for automation was justified.

The second chapter presents the architecture of the monitoring system, describes its components, the database structure, and the ETL process logic.

The third chapter covers the deployment of the working environment using Docker, the implementation of DAG files, testing results, and practical use cases of the system.

Object of the study: the process of monitoring changes in the Domain Name System.

Subject of the study: software implementation of an automated DNS records monitoring system based on Apache Airflow.

ЗМІСТ

ДОДАТКИ

ПЕРЕЛІК СКОРОЧЕНЬ

DNS – Domain Name System

NS – Name Server

IP – Internet Protocol

DAG – Directed Acyclic Graph

ETL – Extract, Transform, Load

SQL – Structured Query Language

UI – User Interface

ID – Identifier

UPS – Uninterruptible Power Supply

GDPR – General Data Protection Regulation

SIEM – Security Information and Event Management

IDS – Intrusion Detection System

IPS – Intrusion Prevention System

API – Application Programming Interface

TLS – Transport Layer Security

SPF – Sender Policy Framework

MX – Mail Exchange

TXT – Text Record

A (Record) – Address Record

NXDOMAIN – Non-Existent Domain

SERVFAIL – Server Failure

ВСТУП

Актуальність теми. Сучасне суспільство характеризується стрімким розвитком Інтернет-технологій та безперервним розширенням цифрової інфраструктури. У цих умовах надійність та стабільність роботи веб-сервісів і застосунків стають критично важливими для ефективного функціонування бізнесу, державних установ та повсякденного життя користувачів. Одним із фундаментальних елементів, що забезпечує коректну навігацію в мережі є система доменних імен (англ. DNS). Навіть незначні збої або несанкціоновані зміни в DNS-записах можуть спричинити серйозні зупинки сервісів, інформаційні витоки чи репутаційні втрати.

У відповідь на зростаючі вимоги до доступності та безпеки онлайнових ресурсів, особливо в сучасних гібридних чи хмарних інфраструктурах, постає потреба у створенні високоефективного та масштабованого механізму моніторингу DNS. Своєчасне виявлення та попередження критичних змін або помилок дає змогу уникнути фінансових збитків і підтримувати безперебійну роботу цифрових послуг. Тому розробка й впровадження автоматизованої системи моніторингу DNS-записів з використанням сучасних технологій оркестрації робочих процесів, таких як Apache Airflow, є одним із перспективних напрямів досліджень у сфері адміністрування та безпеки мережевих інфраструктур.

Метою даного дослідження є підвищення якості послуг за допомогою створення ефективної та гнучкої системи моніторингу DNS-записів на базі Apache Airflow. Для досягнення поставленої мети потрібно виконати наступні **завдання**:

- проаналізувати стан досліджень у галузі моніторингу DNS та автоматизації робочих процесів;
- обґрунтувати та спроєктувати архітектуру системи для збору, обробки та перевірки DNS-записів;

- розробити ETL-процес із використанням Apache Airflow, що забезпечуватиме регулярний моніторинг і виявлення змін у DNS-записах;
- реалізувати та протестувати систему, оцінити її продуктивність та надійність;
- дослідити можливості масштабування та впровадження системи в різних інфраструктурних сценаріях.

Об’єктом дослідження є процеси автоматизованого моніторингу DNS-записів у мережевих інфраструктурах.

Предметом дослідження є методи та інструменти створення, розгортання і оркестрації системи моніторингу DNS-записів на базі Apache Airflow.

Практичне значення одержаних результатів. Запропонована система моніторингу DNS-записів дає змогу оперативно виявляти й аналізувати зміни, які можуть впливати на доступність та безпеку веб-сервісів. У результаті впровадження розробленої системи підвищується стійкість інфраструктури до збоїв, зменшується час простою й оптимізується адміністрування доменних записів. Це, своєю чергою, сприяє забезпеченню якісного надання ІТ-послуг і може бути успішно використане в компаніях різного масштабу, які потребують надійного та гнучкого контролю за станом DNS.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Огляд DNS та необхідність його моніторингу

Система доменних імен (Domain Name System, DNS) є одним із компонентів сучасної Інтернет-інфраструктури, що забезпечує зручний та ефективний механізм перетворення зрозумілих людині доменних імен у числові IP-адреси, які використовуються мережевими протоколами для маршрутизації трафіку. Без DNS-каталогу взаємодія між користувачами та сервісами була б значно ускладнена, адже замість простих доменів, таких як google.com чи ukr.net, довелося б запам’ятовувати послідовності чисел у вигляді 172.217.20.206 або 195.24.68.19 [1].

Архітектура системи DNS є ієрархічною та розподіленою, що дозволяє розподіляти навантаження між великою кількістю серверів, кожен з яких відповідає за обслуговування своєї частини доменних просторів. Центральну роль у системі відіграють кореневі сервери, сервери доменів верхнього рівня (TLD), авторитетні сервери окремих доменів та рекурсивні резолвери, які обслуговують кінцевих користувачів. Завдяки такій структурі забезпечується масштабованість та відмовостійкість глобальної системи доменних імен.

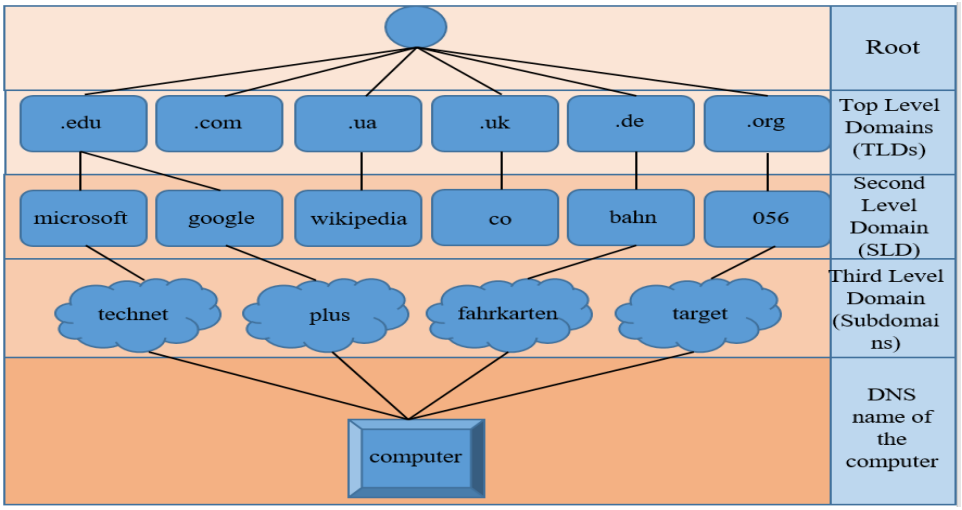


Рисунок 1.1 – Ієрархічна система доменних імен DNS

Функціонування DNS включає декілька важливих процесів, таких як делегування зон, створення та підтримка актуальних записів (A, AAAA, NS, MX, TXT тощо), а також кешування відповідей для зменшення затримок при повторних запитах. У той же час, через відкриту та розподілену природу системи DNS, вона залишається вразливою до цілого спектра загроз, які можуть виникати як з боку зловмисників, так і через технічні помилки або ненавмисні дії адміністраторів [2].

З огляду на важливість коректної роботи DNS для бізнесу, державних установ та користувачів, виникає нагальна потреба у постійному моніторингу стану та змін DNS-записів. Навіть незначні відхилення або помилки, такі як неправильне налаштування NS-записів, зміна IP-адреси важливого сервера чи зникнення критичних записів, можуть призвести до повної недоступності вебресурсів, перебоїв у роботі поштових сервісів, витоку конфіденційної інформації або масштабних репутаційних втрат.

Крім технічних збоїв, значну загрозу становлять цілеспрямовані атаки, зокрема DNS Spoofing, Cache Poisoning та перехоплення або підміна DNS-відповідей на рівні мережевих провайдерів або компрометованих серверів. У таких випадках користувачі можуть бути перенаправлені на фішингові або шкідливі сайти, що створює додаткові ризики фінансових втрат та компрометації даних.

Моніторинг DNS-записів дозволяє у режимі реального часу відстежувати їхню цілісність та відповідність очікуваним значенням, своєчасно виявляти несанкціоновані зміни та оперативно реагувати на потенційні загрози. Це особливо актуально для організацій із великою кількістю доменів, піддоменів та складною інфраструктурою, де навіть одинична помилка може спричинити каскадні збої та значні фінансові втрати [3].

Зважаючи на зростаючу кількість онлайн-сервісів та критичну залежність сучасного суспільства від цифрових технологій, моніторинг DNS стає обов’язковим елементом комплексної системи інформаційної безпеки та безперервності бізнес-процесів. Саме тому питання автоматизації цього процесу

із використанням сучасних інструментів оркестрації, таких як Apache Airflow (див. рисунок 1.2), набуває особливої актуальності та становить значний інтерес для дослідників, системних адміністраторів та інженерів з безпеки [4].

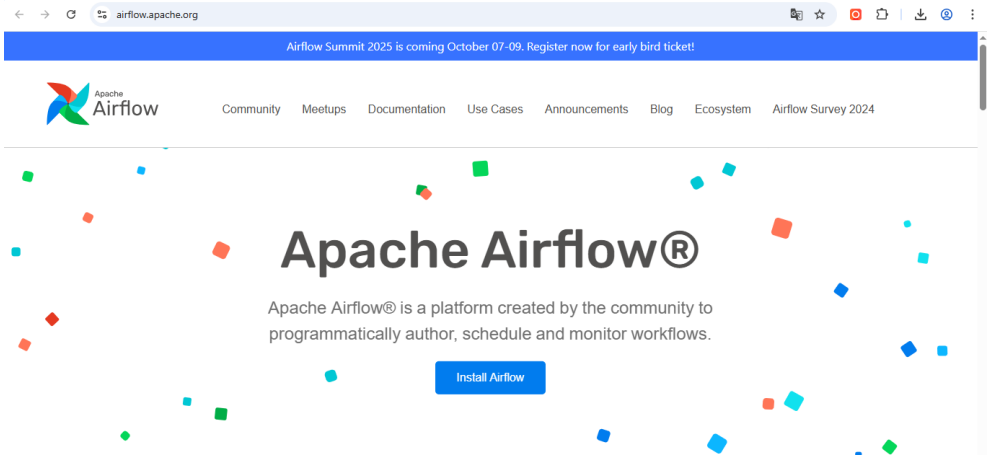


Рисунок 1.2 – Головна сторінка сервісу Apache Airflow

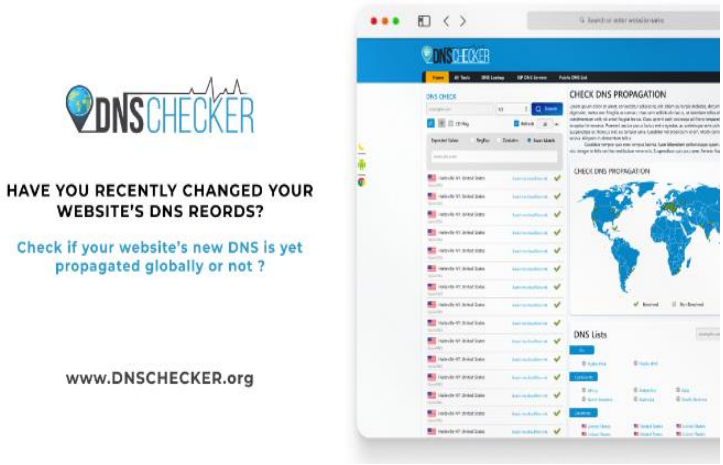
Зважаючи на описані вище фактори, стає очевидною необхідність автоматизації процесу моніторингу DNS-записів. Ручний контроль навіть для середніх інфраструктур може стати складним і ресурсозатратним завданням, що призводить до зниження оперативності реагування та зростання ризиків безпеки. Саме тому використання сучасних платформ автоматизації, таких як Apache Airflow, дозволяє ефективно вирішувати поставлені завдання, забезпечуючи регулярний, оперативний та масштабований моніторинг. Такий підхід дає можливість зменшити ймовірність людських помилок, оптимізувати витрати ресурсів та посилити контроль за цілісністю та безпекою DNS-інфраструктури. Впровадження автоматизованої системи моніторингу DNS також створює підґрунтя для інтеграції з іншими компонентами системи інформаційної безпеки організації, що сприяє формуванню комплексного і надійного захисту її цифрових активів.

1.2 Аналіз існуючих рішень моніторингу DNS-записів

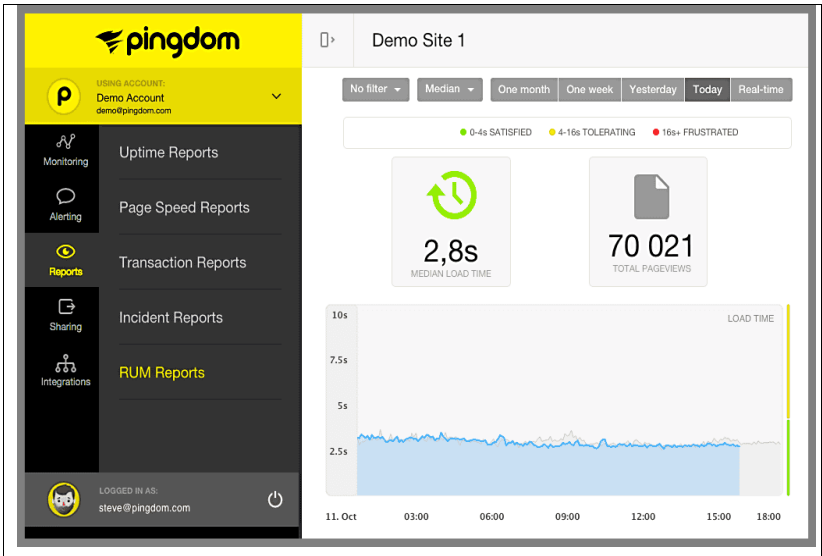
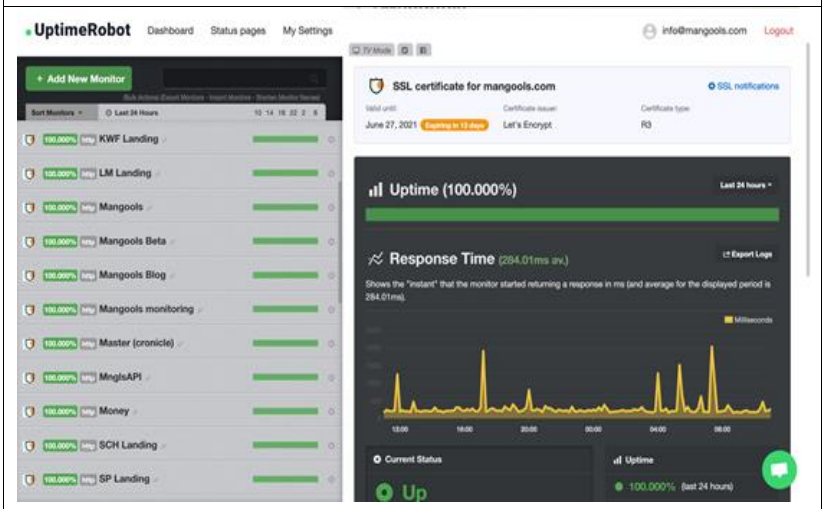
Моніторинг DNS-записів є важливим компонентом забезпечення доступності, коректності та безпеки функціонування інформаційних систем і вебресурсів. З огляду на критичну роль DNS у мережевій взаємодії, протягом останніх років з’явилося чимало рішень, які дозволяють відслідковувати зміни DNS-записів, оперативно реагувати на інциденти та забезпечувати цілісність доменної інформації. Існуючі інструменти можна умовно розділити на декілька груп залежно від їхньої функціональності, архітектури та методів інтеграції з іншими системами [5, 6].

Одна з найбільш поширених категорій – це онлайн-сервіси моніторингу DNS, які пропонують платформи на кшталт DNSChecker, Pingdom, UptimeRobot та інших (див. таблицю 1.1) [7, 8, 9].

Таблиця 1.1 – Характеристики онлайн-сервісів моніторингу DNS

Характеристики сервісів	**Характеристика**
	DNSchecker - простий веб-сервіс, що дозволяє перевіряти DNS-записи в реальному часі з різних географічних локацій. Підтримує багато типів записів (A, MX, NS, TXT тощо). Має зрозумілий інтерфейс і швидку перевірку стану DNS-записів.

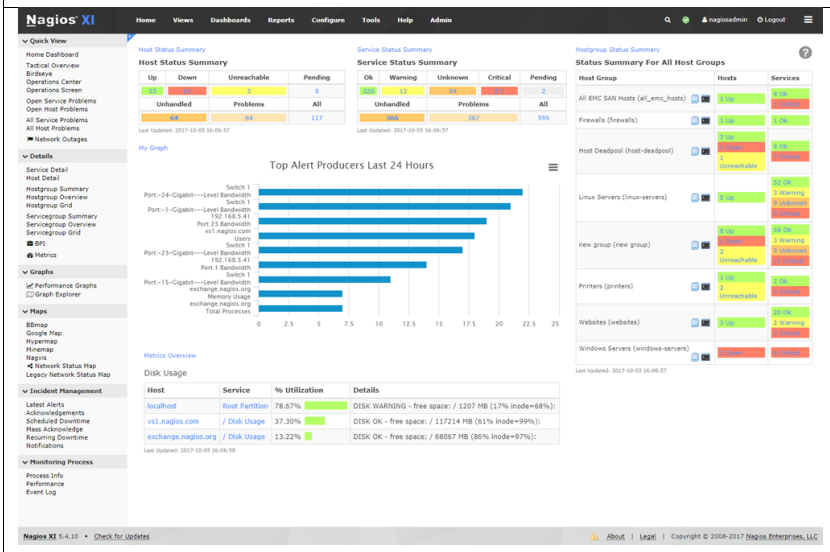
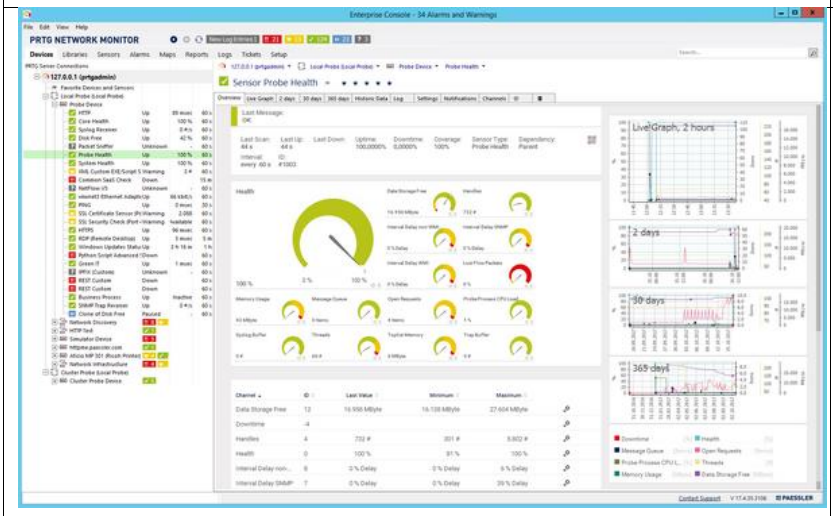
Продовження таблиці 1.1

	Pingdom - хороший онлайн-інструмент для моніторингу доступності вебресурсів, включаючи перевірку DNS-записів. Дозволяє налаштувати сповіщення про зміни чи помилки через email, SMS або інші канали зв'язку. Забезпечує статистику доступності, швидкодії та історію змін.
	UptimeRobot - безкоштовний і простий у використанні сервіс, що дозволяє регулярно перевіряти доступність та коректність DNS-записів. Пропонує швидкі налаштування, автоматичні сповіщення при виникненні проблем (наприклад, через email чи месенджери), а також збереження історичних даних для аналізу стану ресурсів.

Такі сервіси зазвичай надають користувачам вебінтерфейс, де можна налаштувати список доменів і типи записів для моніторингу. Періодично система надсилає запити до публічних DNS-серверів і порівнює отримані відповіді з еталонними даними. У разі виявлення змін або недоступності домену система сповіщає користувача через email, SMS або месенджери. Основні переваги таких рішень – простота використання та відсутність необхідності налаштовувати власну інфраструктуру. Водночас, такі сервіси мають обмежену гнучкість, часто не дозволяють інтегруватися з внутрішніми корпоративними DNS-серверами та мають обмежену частоту перевірок у безкоштовних версіях.

Інша категорія – спеціалізовані системи моніторингу інфраструктури, такі як Zabbix, Nagios або PRTG Network Monitor (див. таблицю 1.2) [10, 11, 12].

Таблиця 1.2 – Характеристики спеціалізованих систем моніторингу інфраструктури.

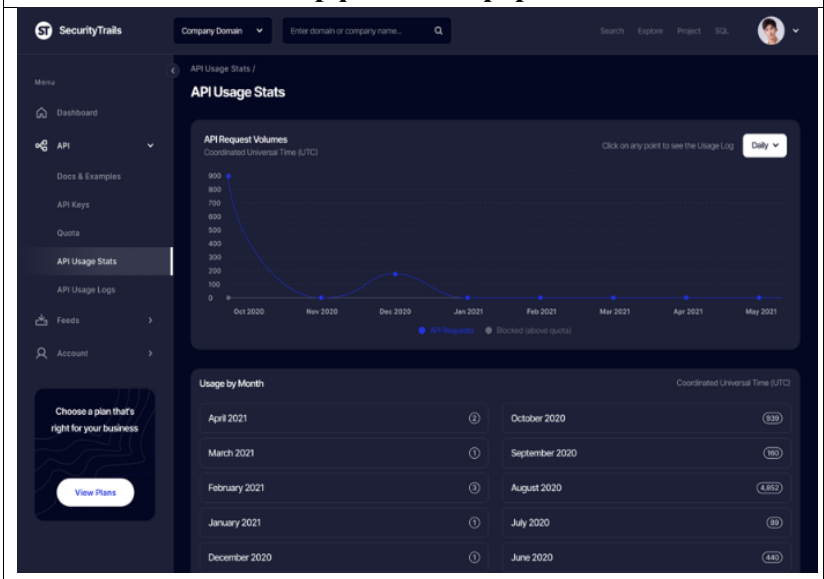
Інтерфейс систем Zabbix, Nagios, PRTG Network	Характеристика
	Zabbix - сильна open-source платформа для моніторингу IT-інфраструктури. Має широкі можливості налаштування, підтримує моніторинг серверів, мереж, DNS-записів та сервісів. Включає систему сповіщень, звіти та графіки для аналізу роботи інфраструктур.
	Nagios - популярне open-source рішення для моніторингу серверів, мереж та сервісів, у тому числі DNS. Відоме своєю стабільністю та гнучкістю завдяки використанню великої кількості плагінів. Забезпечує оперативні сповіщення про зміни статусу та збої.
	PRTG Network - комплексне комерційне рішення для моніторингу мереж, серверів, сервісів і DNS-записів. Має зручний вебінтерфейс, можливість створення інтерактивних дашбордів і розвинуту систему сповіщень. Підходить як для малих, так і великих компаній завдяки легкості налаштувань і високій продуктивності.

Ці системи можуть включати модулі для перевірки DNS-записів та перевірки доступності DNS-серверів. Вони забезпечують значно більшу гнучкість у налаштуванні, дозволяють інтегрувати моніторинг DNS у загальну систему контролю за станом мережі, серверів та сервісів. Крім того, такі рішення

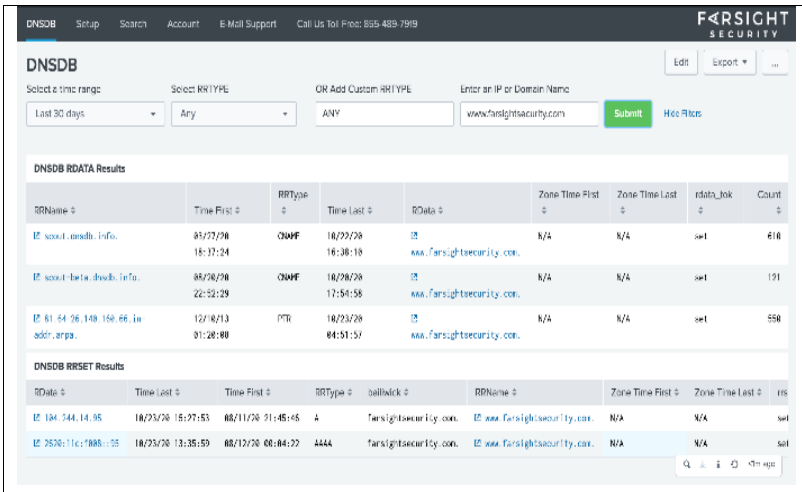
часто підтримують налаштування порогів тривог, сценарії автоматичного реагування та збереження історичних даних для подальшого аналізу. Однак, повноцінне впровадження таких систем вимагає значних витрат часу та ресурсів на початкове налаштування та супровід, що може бути складним для невеликих компаній.

Окрему нішу займають рішення для забезпечення інформаційної безпеки та виявлення загроз (Threat Intelligence), які можуть відслідковувати зміни DNS-записів у контексті виявлення атак, таких як DNS hijacking, DNS spoofing або takeover доменів. Серед таких рішень можна виділити SecurityTrails, Farsight DNSDB (див. таблицю 1.3) та аналогічні платформи, які зберігають історичні дані про зміни DNS-записів для мільйонів доменів у глобальному масштабі [13, 14].

Таблиця 1.3 – Характеристики зазначених платформ класу Threat Intelligence для моніторингу DNS-записів

Інтерфейс платформи	Характеристика
	SecurityTrails - онлайн-сервіс, який спеціалізується на історичному аналізі DNS-записів та IP-адрес. Надає можливість відстежувати зміни у DNS, допомагаючи оперативно виявляти та попереджати такі загрози, як DNS hijacking та domain takeover. Інструмент має зрозумілий інтерфейс і дозволяє швидко аналізувати історичні дані.

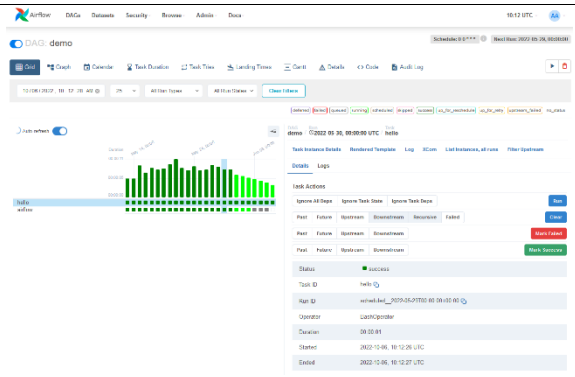
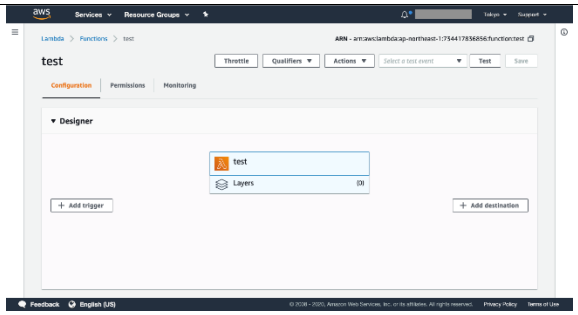
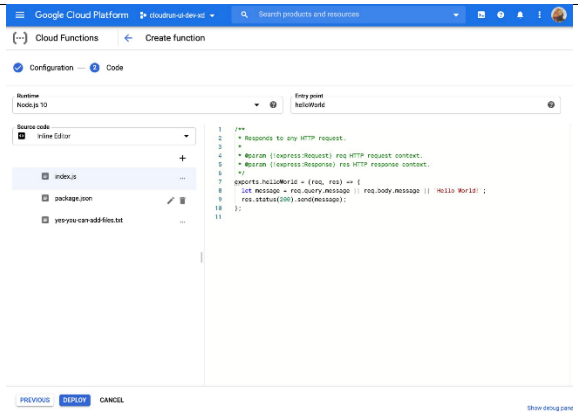
Продовження таблиці 1.3

	Forsight - потужна платформа для моніторингу та аналізу DNS-інформації, яка пропонує доступ до великого обсягу історичних DNS-даних у глобальному масштабі. Дозволяє виявляти й аналізувати підозрілі зміни DNS-записів, оперативно реагуючи на загрози, такі як DNS spoofing та domain hijacking. Підходить для глибокого аналізу та виявлення трендів безпеки.

Ці рішення дозволяють відстежувати не лише поточний стан DNS-записів, а й отримувати інформацію про попередні зміни, виявляти підозрілі тенденції та аналізувати потенційні загрози. Вартість використання таких платформ може бути досить високою, що обмежує їхнє використання для невеликих компаній та приватних користувачів.

Останнім часом набувають популярності кастомні рішення на основі хмарних платформ та автоматизованих робочих процесів, які дозволяють компаніям створювати власні гнучкі системи моніторингу DNS із використанням таких інструментів, як Apache Airflow, AWS Lambda, Google Cloud Functions або Azure Logic Apps (див. таблицю 1.4) [4, 15, 16].

Таблиця 1.4 – Характеристики зазначених платформ автоматизації для створення кастомних рішень

Інтерфейс платформи	Характеристика
	Гнучка open-source платформа для створення, оркестрації та автоматизації робочих процесів (ETL). Використовує Directed Acyclic Graphs (DAG) для планування й моніторингу задач. Дозволяє легко інтегруватися з різними системами, підтримує складні сценарії перевірки та реагування.
	Сервіс хмарних функцій без серверів від Amazon, що дозволяє запускати код автоматично за заданими подіями або за розкладом. Підходить для швидкого створення гнучких рішень моніторингу DNS, які не вимагають підтримки серверної інфраструктури.
	Сервіс хмарних функцій від Google, який надає можливість виконання подієвих задач (event-driven). Простий у налаштуванні, дозволяє створювати легкі, швидкі та масштабовані рішення моніторингу DNS з автоматичною інтеграцією у Google Cloud.

Такі підходи дозволяють поєднувати перевірку DNS-записів із іншими етапами моніторингу та аналітики, автоматично зберігати дані в базах даних або дата-лейках, запускати сповіщення чи контрзаходи у разі виявлення критичних змін. Головна перевага – повна гнучкість та можливість адаптації системи під специфічні потреби компанії. Недоліком є необхідність володіти компетенціями з налаштування таких платформ та побудови ETL-процесів, а також необхідність постійного технічного супроводу.

Аналіз існуючих рішень моніторингу DNS-записів свідчить про широкий спектр підходів та інструментів, кожен з яких має свої переваги та недоліки

залежно від конкретних потреб, масштабів та ресурсів організації. Вибір оптимальної стратегії моніторингу залежить від багатьох факторів, зокрема вимог до оперативності реагування, обсягу доменів, необхідності інтеграції з іншими системами та політики інформаційної безпеки. Враховуючи сучасні тенденції автоматизації та оркестрації IT-процесів, перспективним напрямом є розробка гнучких кастомних систем моніторингу DNS із використанням платформ, які забезпечують масштабованість, інтеграцію та централізоване управління робочими процесами, таких як Apache Airflow.

1.3 Apache Airflow як платформа для автоматизації потоків робіт

Apache Airflow є платформою для створення, планування та моніторингу робочих процесів (workflows), яка забезпечує зручні інструменти для автоматизації складних ETL-процесів, інтеграції з різними системами та оркестрації завдань у гнучкий та прозорий спосіб. Спочатку розроблений у компанії Airbnb для вирішення внутрішніх завдань обробки даних, Airflow швидко здобув популярність завдяки своїй модульній архітектурі, широким можливостям налаштування та активному розвитку у рамках відкритого програмного забезпечення під егідою Apache Software Foundation.

Основу Apache Airflow складають Directed Acyclic Graphs (DAGs) – оркестровані графи завдань, які виконуються у визначеній послідовності або з урахуванням залежностей. Кожне окреме завдання (task) може бути представлено окремим викликом функції, запуском скрипту, виконанням запиту до зовнішньої системи або запуском контейнера (див. рисунок 1.3). Завдяки такому підходу, Airflow дозволяє створювати складні багатоступеневі робочі процеси, які поєднують збори даних, обробку, перевірку, збереження результатів та сповіщення відповідальних осіб [17, 18].

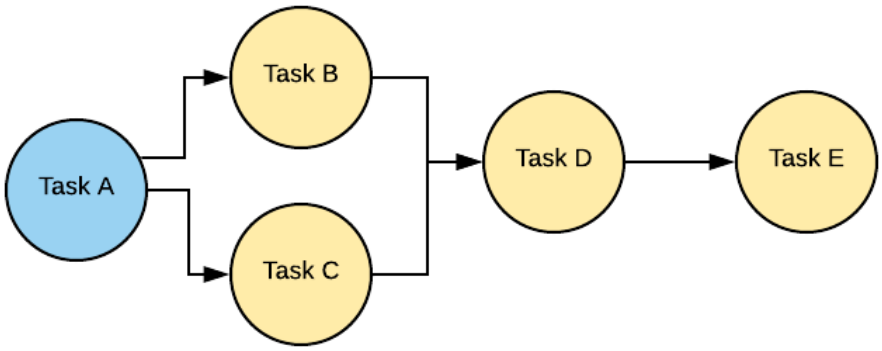


Рисунок 1.3 – Оркестровані графи завдань DAG

Одна з переваг Airflow полягає у його гнучкості щодо налаштування розкладів виконання завдань. Кожен DAG може бути запущений за розкладом (наприклад, щогодини або щодня), за подією або вручну, що особливо важливо для задач моніторингу, які потребують регулярних перевірок у режимі реального часу. Для моніторингу DNS-записів така функціональність дозволяє автоматизувати періодичні запити до DNS-серверів, збирати актуальні дані, порівнювати їх із попередніми значеннями та фіксувати виявлені зміни у базі даних або системі логування.

Airflow також забезпечує зручний механізм обробки параметризованих DAG'ів, де один робочий процес може запускатися з різними наборами параметрів. Це дозволяє налаштувати єдиний універсальний DAG для моніторингу широкого переліку доменів і різних типів DNS-записів – A, NS, MX, TXT тощо. Кожне завдання у DAG може виконувати перевірку конкретного домену за заданим типом запису, що забезпечує високу гнучкість та масштабованість системи моніторингу [19].

Ще однією важливою перевагою Airflow є його інтеграція з системами сповіщень та аналітики. У разі виявлення критичних змін у DNS-записах, таких як зміна NS або A-записів на невідомі або підозрілі адреси, Airflow може автоматично генерувати сповіщення у Slack, Telegram, електронну пошту чи будь-яку іншу систему алертів, що дозволяє адміністраторам оперативно реагувати на інциденти.

Крім того, завдяки гнучкій інтеграції з системами зберігання даних, такими як PostgreSQL, MySQL, Google BigQuery або Amazon S3, Airflow дозволяє накопичувати історичні дані DNS-записів для подальшого аналізу, побудови звітності або виявлення довгострокових трендів. Це особливо цінно для великих компаній із розгалуженою доменною структурою, де критично важливо мати централізований контроль над змінами у DNS [20].

Ще однією важливою перевагою є модульність Apache Airflow. Завдяки можливості підключення власних операторів (Operators) та сенсорів (Sensors), можна розробити кастомні компоненти, які будуть виконувати специфічні перевірки DNS-записів, враховуючи особливості внутрішньої інфраструктури компанії чи специфічні вимоги до контролю доменів. Це дозволяє легко адаптувати систему до потреб конкретного бізнесу або навіть галузі.

Apache Airflow є надзвичайно зручним і гнучким інструментом для побудови автоматизованих систем моніторингу DNS-записів. Його масштабованість, розвинена екосистема інтеграцій, можливість створення параметризованих DAG'ів та підтримка складних логік обробки даних робить його одним із найбільш перспективних рішень для організацій, які прагнуть забезпечити надійний контроль за станом своїх DNS-записів, знизити ризики інцидентів та автоматизувати процес реагування на потенційні загрози.

Важливою перевагою Apache Airflow є її універсальність – платформа може використовуватись не лише для вузькоспеціалізованих ETL-процесів, але й для автоматизації широкого спектра завдань, пов'язаних з оркестрацією робочих потоків, обробкою даних та інтеграцією різних інформаційних систем. Завдяки цьому Airflow є одним із найбільш універсальних та ефективних рішень для автоматизації й керування робочими процесами у сфері IT.

1.4 Визначення вимог до системи моніторингу DNS

Проектування ефективної системи моніторингу DNS-записів потребує чіткого визначення функціональних і нефункціональних вимог, які забезпечать

коректність, стабільність і надійність її роботи. Враховуючи особливості сучасних мережевих інфраструктур та потреби організацій у своєчасному виявленні змін у DNS-записах, система має відповідати низці ключових критеріїв.

З функціональної точки зору, система повинна забезпечувати можливість регулярного виконання перевірок для широкого переліку доменів, включаючи як основні домени, так і численні піддомени. Під час кожної перевірки система має запитувати актуальні значення DNS-записів певних типів – зокрема A, AAAA, NS, MX, TXT та CNAME, – та порівнювати їх із попередньо зафіксованими значеннями, зберігаючи історію змін для подальшого аналізу. У разі виявлення змін або помилок розв’язування імен, система повинна автоматично створювати відповідні сповіщення для адміністративного персоналу або систем інформаційної безпеки.

Важливою вимогою є підтримка гнучкого налаштування частоти перевірок для різних груп доменів, оскільки для критичних ресурсів може бути необхідним моніторинг із мінімальними інтервалами, тоді як для менш важливих доменів достатньо перевірок раз на добу або тиждень. При цьому система має підтримувати централізоване зберігання налаштувань і конфігурацій, що дозволить зручно керувати списками доменів і правилами перевірки.

Окремим аспектом є збереження результатів перевірок у базі даних для подальшого аналізу та формування звітності. Система повинна забезпечувати можливість аналізу історичних даних, побудови трендів, визначення частоти змін та виявлення аномалій. Для цього передбачається підтримка гнучкої схеми бази даних, яка дозволяє фіксувати не лише поточні значення DNS-записів, але й додаткову метаінформацію, наприклад, час проведення перевірки, джерело даних (який саме DNS-сервер відповідав) та статус запису (актуальний, змінений, видалений).

З нефункціональної точки зору, система повинна забезпечувати високу надійність та відмовостійкість. Враховуючи, що моніторинг DNS є критичною функцією для забезпечення інформаційної безпеки, навіть короткострокові збої

у роботі системи можуть призвести до пропуску важливих змін або атак. Тому система повинна мати механізми автоматичного відновлення роботи після збоїв та забезпечувати збереження даних навіть у разі відмови окремих компонентів.

Ще однією важливою вимогою є масштабованість системи, оскільки кількість доменів, які потребують моніторингу, може зростати. Система повинна бути здатна ефективно обробляти запити до тисяч доменів без суттєвих затримок і збереженням цілісності даних. Для цього архітектура має передбачати можливість розподілу навантаження між кількома вузлами або горизонтального масштабування.

Окремо варто наголосити на вимогах до безпеки. Оскільки система обробляє критичну інформацію про корпоративні домени та внутрішню інфраструктуру, усі дані мають передаватися та зберігатися з використанням сучасних засобів шифрування. Доступ до системи має бути захищений засобами автентифікації та авторизації, а всі дії користувачів мають бути протокольовані для подальшого аудиту.

Крім того, система повинна підтримувати інтеграцію з існуючими засобами оповіщення та управління інцидентами, такими як Slack, Telegram, Microsoft Teams або системи ITSM. Це забезпечить оперативне реагування на виявлені проблеми та зменшить час між фіксацією інциденту і його усуненням.

Комплексно визначені функціональні та нефункціональні вимоги до системи моніторингу DNS-записів створюють передумови для розробки надійного, гнучкого та масштабованого рішення, здатного ефективно вирішувати завдання забезпечення стабільності, безпеки та контрольованості доменної інфраструктури організації. Чітке визначення цих вимог дозволяє не лише сформувати стійку технічну основу системи, але й забезпечити оптимальну взаємодію її компонентів, інтеграцію із зовнішніми інформаційними системами та платформами, а також надати необхідні механізми для оперативного реагування на інциденти чи зміни в конфігурації DNS. Завдяки цьому організація отримує можливість значно скоротити час простоїв онлайн-сервісів, зменшити

ризики інформаційних витоків і помилок через людський фактор, а також підвищити загальний рівень довіри до її цифрових послуг.

1.5 Висновки до першого розділу

У першому розділі проведено комплексний аналіз предметної області, що охоплює основні аспекти функціонування системи доменних імен (DNS), обґрунтовано важливість та необхідність її моніторингу, а також досліджено сучасні підходи та інструменти для автоматизації цього процесу. Встановлено, що DNS є критично важливою складовою сучасної мережевої інфраструктури, від стабільності та коректності роботи якої залежить доступність онлайн-сервісів, інформаційна безпека компаній та захист даних користувачів. Навіть незначні зміни або збої у конфігурації DNS-записів можуть призвести до значних фінансових та репутаційних втрат.

Аналіз існуючих рішень продемонстрував, що на ринку представлено широкий спектр інструментів моніторингу DNS-записів – від хмарних сервісів та систем моніторингу інфраструктури до спеціалізованих платформ з виявлення загроз та кастомних рішень на базі сучасних засобів автоматизації. Разом із тим, багато готових рішень мають певні обмеження щодо гнучкості налаштувань, інтеграції з внутрішніми системами або адаптації до специфічних вимог окремих компаній. Це створює потребу у розробці власної системи моніторингу, яка враховуватиме специфіку конкретного середовища та забезпечуватиме високу надійність і функціональність.

Зважаючи на сучасні тенденції автоматизації та оркестрації робочих процесів, доцільним є використання платформи Apache Airflow як базової технології для побудови системи моніторингу DNS-записів. Apache Airflow забезпечує гнучке налаштування робочих процесів, підтримку параметризованих перевірок, зручну інтеграцію зі сторонніми сервісами та можливість масштабування відповідно до потреб інфраструктури.

Також у цьому розділі визначено основні функціональні та нефункціональні вимоги до системи моніторингу DNS-записів, які враховують необхідність регулярного збору та збереження даних, порівняння поточного стану з еталонними показниками, забезпечення централізованого контролю, захисту інформації та інтеграції з системами сповіщень і реагування. Таким чином, перший розділ закладає теоретичну та методологічну основу для подальшого проєктування та реалізації автоматизованої системи моніторингу DNS-записів з використанням Apache Airflow.

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ DNS-ЗАПИСІВ

2.1 Архітектурний дизайн системи

Проєктування системи моніторингу DNS-записів передбачає створення комплексної архітектури, яка забезпечить виконання всіх функціональних вимог, визначених у першому розділі, а також гарантуватиме гнучкість, масштабованість і надійність системи в умовах реальної експлуатації. Основою для розробки такої системи обрано платформу Apache Airflow, що дозволяє будувати оркестровані робочі процеси з регулярним виконанням завдань, обробкою результатів і подальшим реагуванням на виявлені відхилення.

Загальна архітектура системи моніторингу складається з кількох основних компонентів, кожен із яких відповідає за певний етап обробки даних та управління процесами. Центральним елементом є контролер робочих процесів, представлений системою Apache Airflow. Його основна функція – керування виконанням DAG-процесів, кожен із яких включає набір завдань для моніторингу конкретних DNS-записів заданих доменів. Airflow дозволяє автоматизувати планування, запуск, моніторинг і логування таких перевірок, а також забезпечує візуальний контроль за станом виконання та результатами кожного завдання [21].

Джерелом даних для системи є публічні та внутрішні DNS-сервери, до яких надсилаються запити для отримання актуальних даних щодо записів доменів. Система підтримує одночасну перевірку кількох типів DNS-записів (A, NS, MX, TXT, CNAME тощо), що дозволяє забезпечити комплексний контроль за всіма критично важливими параметрами доменної інфраструктури.

Зібрані дані проходять через процес попередньої обробки, в рамках якого система перевіряє валідність відповідей, зіставляє отримані дані з попередніми значеннями та фіксує виявлені зміни. Усі результати моніторингу зберігаються у централізованій базі даних, структура якої передбачає збереження історії

кожного DNS-запису, часу перевірки, джерела відповіді та статусу (актуальний, змінений, відсутній).

Окрему роль у системі відіграє модуль сповіщень, який інтегрується з популярними системами комунікації, такими як Telegram, Slack, електронна пошта чи системи ITSM. У разі виявлення критичних змін або проблем із доступністю DNS-записів, система автоматично генерує сповіщення для відповідальних осіб, що дозволяє оперативно реагувати на потенційні інциденти [22].

Архітектура системи також передбачає можливість розширення функціональності за рахунок додаткових модулів, таких як аналітичний блок для побудови звітів і візуалізації історичних даних, або модуль інтеграції з зовнішніми платформами Threat Intelligence для перевірки підозрілих змін у DNS-записах.

З технічної точки зору, система спроєктована таким чином, щоб забезпечити горизонтальну масштабованість і можливість розподілу навантаження між кількома вузлами. Це досягається за рахунок використання контейнеризованих компонентів, що розгортаються у середовищі Docker або Kubernetes, а також можливості динамічного додавання нових DAG для контролю додаткових доменів чи типів записів без переривання роботи всієї системи (див. рисунок 2.1).

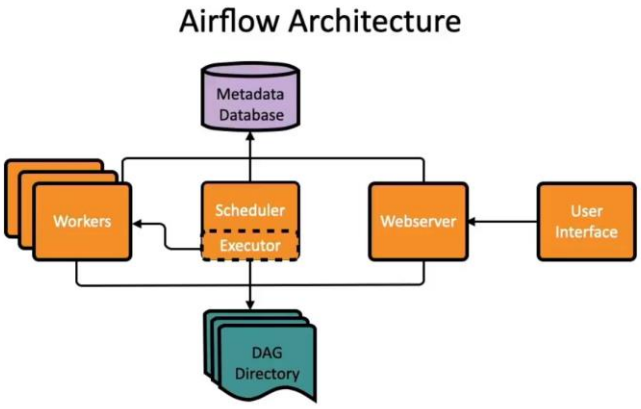


Рисунок 2.1 – Архітектура системи моніторингу DNS-записів на базі Apache Airflow

Таким чином, архітектурний дизайн системи моніторингу DNS-записів поєднує сучасні підходи до автоматизації, оркестрації та обробки даних, забезпечуючи високу гнучкість, надійність та інтеграцію з іншими елементами IT-інфраструктури організації. Це створює необхідне підґрунтя для побудови масштабованого та ефективного рішення, яке здатне забезпечити стабільний контроль за станом DNS-записів та оперативне реагування на інциденти.

2.2 Реалізація ETL-процесу збору DNS-даних

Реалізація системи моніторингу DNS-записів передбачає створення ефективного ETL-процесу (Extract, Transform, Load), який забезпечить регулярний збір даних, їхню обробку та збереження у централізованому сховищі для подальшого аналізу та контролю змін. З огляду на використання платформи Apache Airflow як основного засобу оркестрації процесів, кожен етап ETL-процесу виконується в межах окремих завдань (Tasks), які об’єднані у єдиний DAG.

На етапі Extract (витягування даних) система отримує дані про поточний стан DNS-записів для кожного домену та кожного типу запису, що підлягає моніторингу. Для цього використовуються стандартні бібліотеки мови Python, такі як dnspython, які дозволяють формувати DNS-запити до визначених серверів та отримувати відповіді у структурованому вигляді. На цьому етапі важливо забезпечити підтримку різних типів DNS-записів – A, AAAA, NS, MX, TXT, CNAME – оскільки кожен із них має значення для повноцінного контролю стану доменів.

Процес збору даних може бути параметризованим, тобто при виконанні DAG передаються параметри, які визначають перелік доменів, типи записів та частоту перевірок. Це забезпечує гнучкість і дозволяє масштабувати систему для обробки великої кількості доменів без створення окремих DAG для кожного.

Етап Transform (обробка та перевірка даних) виконується безпосередньо після отримання даних. Основна задача:

- розбиття даних;
- агрегація одного чи більше значень;
- видалення зайвого (видалення частини даних, або видалення цілих значень);
- обрахунок нових значень (додавання, віднімання, множення даних), тощо.

Також на цьому етапі можуть застосовуватись додаткові перевірки – наприклад, валідація IP-адрес, перевірка належності нових IP до певних мережевих діапазонів або зіставлення отриманих даних із базами Threat Intelligence.

Етап Load (збереження даних) відповідає за запис актуальної інформації до централізованого сховища даних. Вибір типу сховища залежить від вимог до масштабованості та гнучкості системи. Для зберігання поточних та історичних даних найчастіше використовуються реляційні бази даних, такі як PostgreSQL або MySQL, оскільки вони забезпечують зручний механізм для зберігання структурованих даних у вигляді таблиць із можливістю виконання аналітичних запитів.

У базі даних зберігаються наступні ключові параметри для кожного запису:

- доменне ім’я;
- тип запису;
- значення запису;
- час отримання даних;
- джерело відповіді (DNS-сервер, який надав дані);
- статус запису (актуальний, змінений, видалений)

Крім збереження у базі даних, у системі може бути передбачене додаткове логування у зовнішні системи журналювання або моніторингу, наприклад у Grafana Loki чи Elasticsearch, що дозволяє отримати централізовану історію подій для подальшого аудиту.

Оскільки процес моніторингу DNS є періодичним, важливою частиною ETL-процесу є механізм налаштування розкладів запуску, який реалізується безпосередньо через Apache Airflow. Завдяки цьому адміністратори системи можуть гнучко налаштовувати періодичність перевірок залежно від важливості окремих доменів або зон [23].

Таким чином, ETL-процес збору DNS-даних у рамках системи моніторингу є основним технічним механізмом, що забезпечує отримання актуальної інформації, її аналіз та збереження для подальшого контролю і реагування. Реалізація цього процесу на базі Apache Airflow забезпечує високу гнучкість, масштабованість та можливість інтеграції з іншими елементами IT-інфраструктури організації.

2.3 Проектування бази даних для збереження та обробки моніторингових даних

Ефективна робота системи моніторингу DNS-записів неможлива без надійного сховища даних, яке забезпечує збереження, обробку та аналітику зібраних результатів. Правильне проектування структури бази даних є важливим етапом розробки системи, оскільки від нього залежить швидкість отримання даних, зручність виконання аналітичних запитів та можливість масштабування у разі збільшення обсягу даних.

З урахуванням вимог до системи моніторингу, база даних повинна забезпечувати зберігання як поточного стану DNS-записів, так і їх історичних змін. Це дозволяє відстежувати динаміку змін у розрізі конкретних доменів, аналізувати тенденції та виявляти аномалії, які можуть свідчити про технічні проблеми або спроби несанкціонованого втручання.

Проектування бази даних передбачає створення декількох основних таблиць, які логічно відображають структуру даних і забезпечують зв’язки між доменами, їх DNS-записами та історією змін. Основними об’єктами моделі даних є домен, запис та історія змін (див. рисунок 2.3).

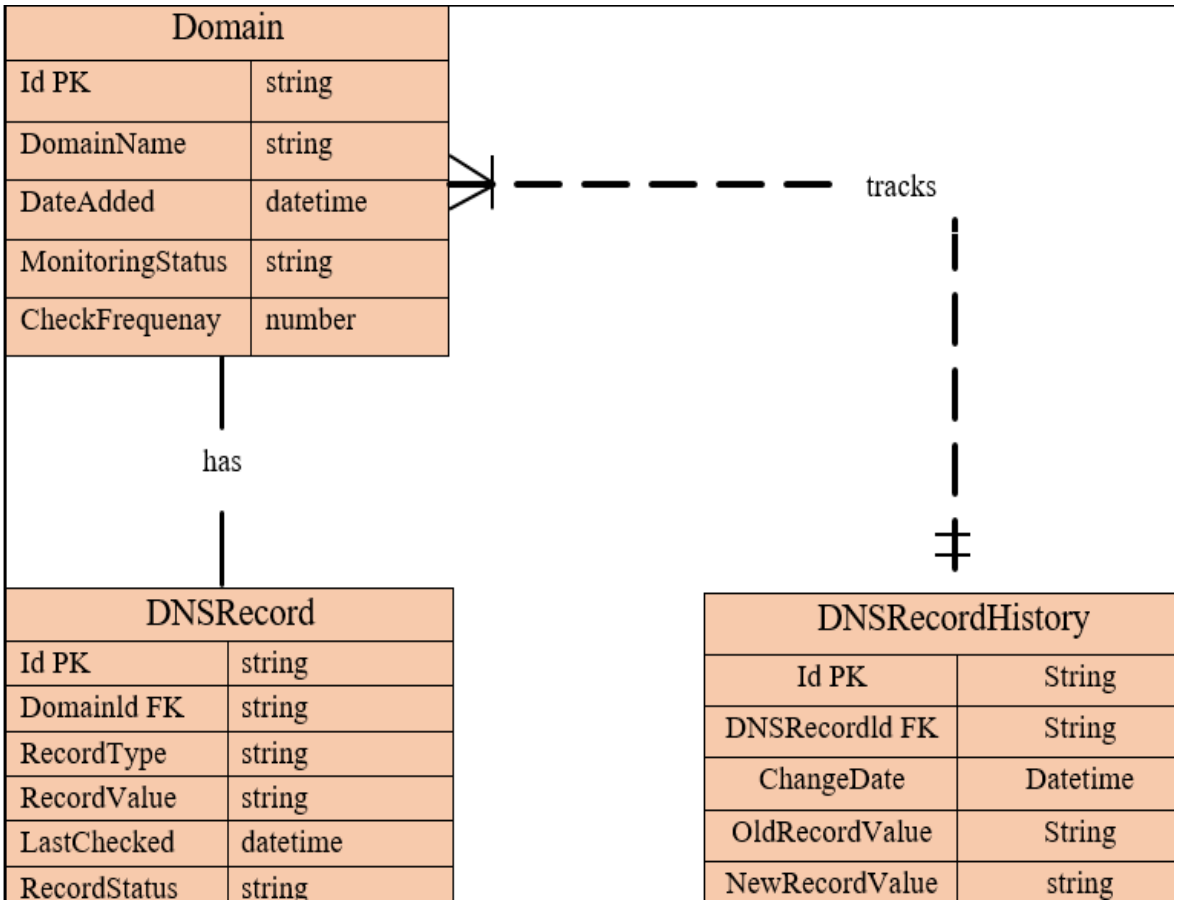


Рисунок 2.3 – ER-діаграма бази даних моніторингу DNS

Таблиця domains містить інформацію про всі домени, які перебувають під моніторингом. Для кожного домену фіксуються базові мета-дані, такі як дата додавання до системи, статус моніторингу та налаштування періодичності перевірок.

Таблиця dns_records зберігає поточний стан DNS-записів для кожного домену. Вона містить інформацію про тип запису (A, NS, MX тощо), значення (IP-адреса, доменне ім’я або текстова інформація), час перевірки та статус запису.

Структура бази даних передбачає використання індексів для прискорення пошуку даних за доменами, типами записів та часом перевірки. Це критично важливо для ефективної роботи системи у разі моніторингу великої кількості доменів, особливо для корпоративних інфраструктур із сотнями або тисячами доменів та піддоменів.

Зберігання даних у реляційній базі даних, такій як PostgreSQL, забезпечує не лише структурованість та цілісність даних, але й можливість побудови складних аналітичних запитів для оцінки стабільності роботи DNS-інфраструктури, ідентифікації проблемних зон та оцінки ефективності заходів із забезпечення інформаційної безпеки.

Окрему увагу при проектуванні бази даних слід приділити питанням безпеки. Зокрема, доступ до таблиць із моніторинговими даними повинен бути обмежений лише для авторизованих користувачів, а всі операції запису та зміни даних мають протоколюватися для подальшого аудиту. Усі дані, що передаються між компонентами системи, повинні шифруватися для запобігання перехопленню або підміни.

Проектування бази даних є одним із головних етапів створення системи моніторингу DNS-записів, оскільки саме база даних забезпечує надійне зберігання, історичний контроль та аналітичну обробку всіх зібраних даних. Коректно спроектована структура забезпечує ефективність та продуктивність системи в умовах високого навантаження та динамічного масштабування.

2.4 Автоматизація перевірки змін у DNS-записах

Автоматизація процесу перевірки змін у DNS-записах є ключовою функцією системи моніторингу, яка дозволяє своєчасно виявляти будь-які відхилення у конфігурації доменів та оперативно інформувати відповідальних осіб. Враховуючи, що система базується на Apache Airflow, автоматизація реалізована у вигляді набору параметризованих DAG-процесів, які виконуються відповідно до заданого розкладу або за потреби вручну.

Кожен DAG відповідає за виконання серії завдань (Tasks), які охоплюють отримання поточних DNS-записів для заданих доменів, їх порівняння з попередньо збереженими значеннями, фіксацію змін та формування звіту для подальшої обробки. Завдяки параметризації, один DAG може одночасно

обробляти кілька доменів, а перелік типів записів (A, NS, MX, TXT тощо) визначається у конфігурації при запуску.

Процес перевірки складається з кількох етапів. Спочатку система завантажує перелік доменів та налаштувань із централізованого сховища конфігурацій або бази даних. Для кожного домену та кожного типу запису формується окремий запит до визначених DNS-серверів, результати якого зберігаються у тимчасовій структурі для подальшої обробки.

На наступному етапі кожен отриманий запис порівнюється з останньою зафіксованою версією у базі даних. Якщо значення запису змінилося, додався новий запис або зник існуючий, система фіксує цю подію у таблиці історії змін та додає її до звіту для подальшого аналізу. При цьому зберігається як старе значення, так і нове, а також фіксується точний час виявлення змін та джерело відповіді (конкретний DNS-сервер, який надав дані).

Для підвищення надійності та зниження ризику помилкових спрацювань система може виконувати повторну перевірку у разі виявлення критичних змін – наприклад, зміни NS-записів або повного зникнення ключових A-записів. Повторна перевірка може проводитись із використанням альтернативних DNS-серверів, щоб виключити вплив кешування або локальних збоїв на стороні одного з серверів.

Окремий компонент системи відповідає за аналіз критичності виявлених змін. Наприклад, зміна A-запису, який вказує на публічний веб-сервер, може вважатися менш критичною подією порівняно зі зміною NS-запису, що відповідає за делегування всієї зони домену. У разі виявлення критичних змін система ініціює формування повідомлення, яке передається до системи сповіщень – Telegram, Slack, електронна пошта або інші канали комунікації. Формат повідомлення включає інформацію про домен, тип запису, старе та нове значення, час виявлення та рекомендації для подальших дій.

Завдяки використанню Apache Airflow як платформи оркестрації, весь процес є прозорим та контрольованим – всі етапи перевірок, виконані завдання та результати фіксуються у вбудованих журналах Airflow, що дозволяє

адміністраторам отримувати детальну інформацію про роботу системи та діагностувати можливі проблеми.

Автоматизована перевірка змін забезпечує високу частоту моніторингу без необхідності постійного втручання з боку оператора. Це дозволяє не лише зменшити навантаження на персонал, а й мінімізувати час між фактичною зміною у DNS-записах та її виявленням, що є критичним для забезпечення безперервності роботи онлайн-сервісів та швидкого реагування на інциденти [24].

Таким чином, автоматизована перевірка змін у DNS-записах забезпечує комплексний контроль за актуальністю та коректністю конфігурації доменів, дозволяє оперативно виявляти та аналізувати відхилення та створює надійну основу для побудови сучасної системи моніторингу, орієнтованої на високу доступність та безпеку інфраструктури.

2.5 Висновки до другого розділу

У другому розділі було спроєктовано архітектуру автоматизованої системи моніторингу DNS-записів з використанням платформи Apache Airflow. Запропонована архітектура поєднує сучасні підходи до оркестрації робочих процесів, автоматизації збору даних та аналізу історичних змін, що забезпечує високу гнучкість та масштабованість системи. Завдяки використанню Apache Airflow, система отримала можливість гнучкого налаштування розкладів виконання перевірок, централізованого контролю за всіма етапами моніторингу та інтеграції з існуючими системами сповіщень.

Окрему увагу приділено реалізації ETL-процесу, який включає збір актуальних DNS-даних із публічних та внутрішніх серверів, порівняння їх із попередньо збереженими значеннями, виявлення змін та збереження результатів у централізованому сховищі. Такий підхід забезпечує не лише оперативне виявлення критичних змін, а й створює повноцінну історію всіх зафіксованих

змін, яка може бути використана для подальшого аналізу, побудови звітів та виявлення довгострокових трендів.

Запропонована структура бази даних дозволяє ефективно зберігати поточний стан DNS-записів, фіксувати історію змін та забезпечувати швидкий доступ до даних для виконання аналітичних запитів. Це створює основу для глибокого аналізу стабільності та безпеки DNS-інфраструктури, а також дозволяє інтегрувати систему моніторингу з іншими інформаційно-аналітичними системами компанії.

Автоматизація процесу перевірки змін у DNS-записах, реалізована за допомогою Apache Airflow, забезпечує гнучке управління робочими процесами, контроль за їхнім виконанням та своєчасне формування сповіщень у разі виявлення критичних змін. Завдяки цьому система може оперативно інформувати відповідальних осіб про потенційні загрози або збої, що дозволяє мінімізувати час реагування та запобігти серйозним інцидентам.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ

3.1 Налаштування середовища для роботи

Для реалізації системи моніторингу DNS-записів було обрано гнучке та масштабоване середовище на основі платформи Apache Airflow з використанням контейнеризації через Docker. Такий підхід дозволяє забезпечити швидке розгортання, легке оновлення, ізоляцію залежностей і підтримку розподіленої архітектури.

Проєктна структура включає каталоги `dags`, `logs`, `plugins` та файл конфігурації `docker-compose.yaml`. У каталозі `dags` зберігатимуться DAG-файли (робочі процеси Airflow), `logs` містить журнали виконання задач, а `plugins` призначений для можливих розширень функціональності.

Основу середовища складає файл `docker-compose.yaml`, який описує усі необхідні сервіси, включаючи веб-інтерфейс Apache Airflow, планувальник (scheduler), воркери, базу даних PostgreSQL, брокер повідомлень Redis та інші компоненти.

Конфігурація виглядає наступним чином див. лістинг 3.1.

Лістинг 3.1 – Конфігурація docker-compose.yaml (В лістингу наведено частину конфігурації docker-compose.yaml, повний файл конфігурації знаходиться в додатку Б) .

```
version: '3'

services:
  postgres:
    image: postgres:13
    environment:
      POSTGRES_USER: airflow
      POSTGRES_PASSWORD: airflow
      POSTGRES_DB: airflow
    volumes:
      - ./pgdata:/var/lib/postgresql/data
    restart: always

  redis:
```

```
    image: redis:latest
    restart: always

  airflow-webserver:
    image: apache/airflow:2.8.1
    command: webserver
    depends_on:
      - postgres
      - redis
    ports:
      - "8080:8080"
    environment:
      AIRFLOW__CORE__EXECUTOR: CeleryExecutor
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__RESULT_BACKEND:
db+postgresql://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__BROKER_URL: redis://redis:6379/0
      AIRFLOW__CORE__FERNET_KEY: ''
      AIRFLOW__CORE__LOAD_EXAMPLES: 'False'
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
    restart: always
```

Після створення конфігурації, всі сервіси запускаються за допомогою команд:

```
docker-compose up airflow-init
docker-compose up -d
```

Це дозволяє ініціалізувати базу даних Airflow та створити адміністративного користувача для доступу до веб-інтерфейсу за адресою `http://localhost:8080`.

У середовищі контейнера встановлюється бібліотека `dnspython`, яка відповідає за здійснення DNS-запитів. Вона інсталюється у вебсервері через команду:

```
docker exec -it airflow-dns-monitoring-airflow-webserver-
1 bash
pip install dnspython
```

Після встановлення виходимо з контейнера, і середовище готове до виконання моніторингових задач.

Таким чином, середовище включає повний стек сервісів: Apache Airflow як платформу автоматизації, PostgreSQL для збереження даних, Redis як брокер, dnspython як інструмент для збору DNS, і має можливість масштабування за допомогою Celery Worker.

3.2 Розробка DAG для моніторингу DNS

На основі попередньо налаштованого середовища було створено DAG (Directed Acyclic Graph), який реалізує логіку моніторингу DNS-записів. Граф задач у Apache Airflow дозволяє описати ETL-процес як набір послідовних дій: витягування DNS-записів, аналіз змін, збереження результатів у базу та, при потребі, надсилання сповіщення.

DAG створюється у вигляді Python-файлу у каталозі `dags/`. Файл `dns_monitoring_dag.py` містить наступний код (див. лістинг 3.3).

Лістинг 3.3 – DAG код у вигляді Python-файлу

```
from airflow import DAG
from airflow.operators.python import PythonOperator
from datetime import datetime
import dns.resolver
import psycopg2

def check_dns(domain, record_type):
    try:
        result = dns.resolver.resolve(domain, record_type)
        values = [r.to_text() for r in result]

        conn = psycopg2.connect(
            dbname="airflow",
            user="airflow",
```

```
                password="airflow",
                host="postgres"
            )
            cur = conn.cursor()
            cur.execute(
                "INSERT INTO dns_log (domain, type, values,
checked_at) VALUES (%s, %s, %s, %s)",
                (domain, record_type, str(values), datetime.now())
            )
            conn.commit()
            cur.close()
            conn.close()
        except Exception as e:
            print(f"Error checking {domain}: {e}")

    default_args = {
        'start_date': datetime(2024, 1, 1),
        'retries': 1,
    }

    with DAG(
        dag_id='dns_monitoring_multiple',
        default_args=default_args,
        schedule_interval='@hourly',
        catchup=False,
        description='Моніторинг кількох доменів',
    ) as dag:

        domains = ['google.com', 'amazon.com', 'github.com']
        record_type = 'A'

        for domain in domains:
            PythonOperator(
                task_id=f'check_dns_{domain.replace(".", "_")}',
                python_callable=check_dns,
```

```
                op_kwargs={'domain':    domain,    'record_type':
record_type}
            )
```

Цей DAG запускається щогодини, здійснює запит до DNS для визначеного домену, отримує записи типу A і зберігає результати у таблицю `dns_log`. Гнучкість реалізації дозволяє передавати як параметри інші домени та типи записів, а також розширювати DAG на кілька паралельних задач. Завдяки інтеграції з базою PostgreSQL кожен запуск фіксується і може бути використаний для подальшого аналізу, виявлення змін чи потенційних загроз.

3.3 Тестування та аналіз продуктивності

Після розгортання та впровадження основної логіки системи моніторингу DNS-записів важливою частиною роботи стало її тестування. Цей етап мав на меті перевірити правильність виконання кожного з етапів DAG, стабільність роботи середовища Apache Airflow, правильність збереження даних у базі, а також оцінити ефективність і швидкодію розробленого рішення. Особлива увага приділялася надійності, стійкості до помилок, а також тому, наскільки легко система справляється з розширенням обсягу задач і масштабуванням на більшу кількість доменів.

Після створення конфігураційного файлу docker-compose.yaml було здійснено процес ініціалізації середовища за допомогою інструмента Docker Compose. Цей процес включає в себе завантаження та встановлення контейнерів для основних компонентів системи: Apache Airflow, бази даних PostgreSQL, брокера Redis, а також сервісів керування та виконання задач (webserver, scheduler, worker).

Для запуску початкового сценарію ініціалізації, який створює службову базу даних Airflow та адміністративного користувача, використовується команда:

```
docker-compose up airflow-init
```

Ця команда автоматично завантажує всі необхідні образи з репозиторію Docker Hub, у тому числі:

- apache/airflow – образ з усіма компонентами платформи Airflow;
- postgres – образ реляційної бази даних;
- redis – брокер черг для обміну повідомленнями між компонентами Celery.

Процес «pulling» (завантаження) може тривати кілька хвилин, залежно від швидкості з'єднання з мережею. Після завершення завантаження, у консольному виведенні відображається статус створення кожного контейнера (див. Рисунок 3.1).

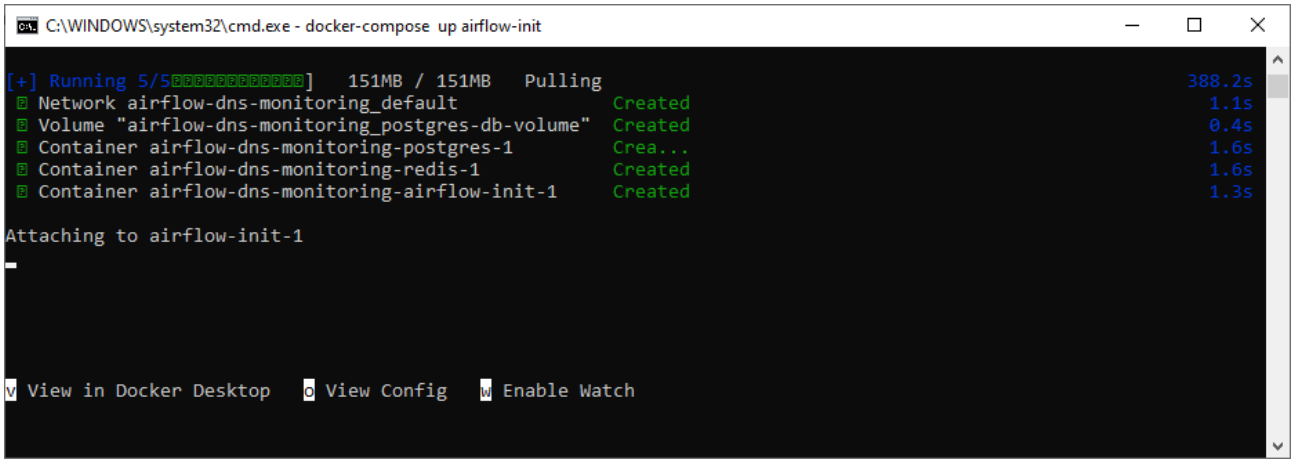


Рисунок 3.1 – Завершення ініціалізації середовища Docker Compose з компонентами Apache Airflow

Після успішної ініціалізації середовище активується командою:

```
docker-compose up -d
```

Це дозволяє запустити всі сервіси у фоновому режимі (див. рисунок 3.2).

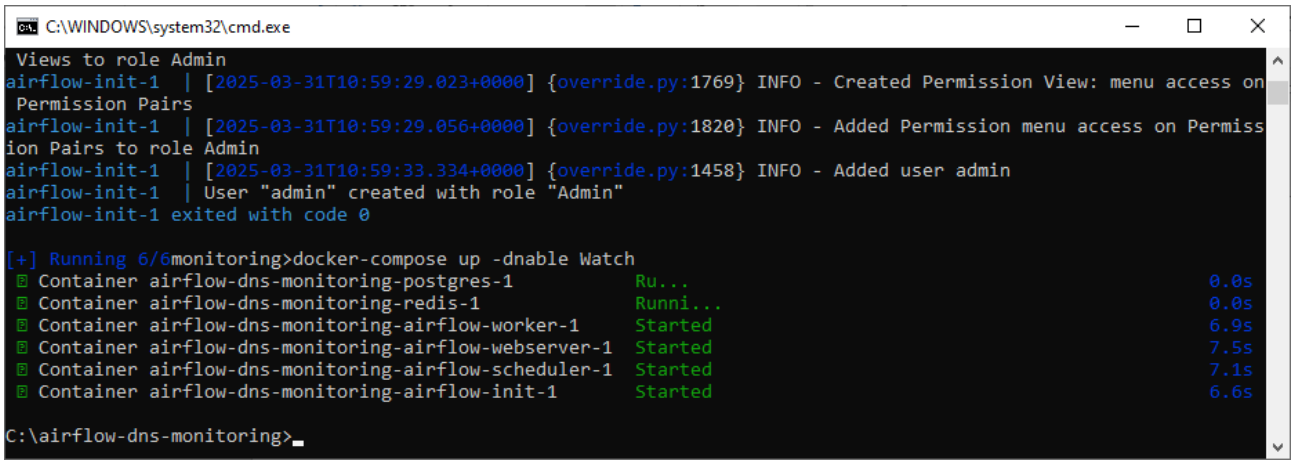


Рисунок 3.2 – Консольне повідомлення про успішний запуск середовища Apache Airflow за допомогою Docker Compose

Далі веб-інтерфейс Apache Airflow стає доступним за адресою http://localhost:8080, де за допомогою облікових даних (admin/admin) користувач отримує доступ до системи керування задачами (див. рисунок 3.3).

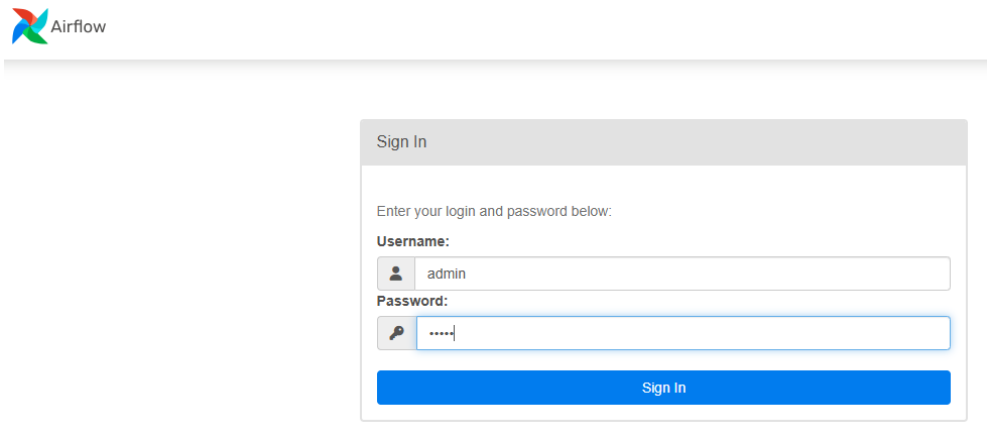


Рисунок 3.3 – Сторінка входу у веб-інтерфейс Apache Airflow

На першому етапі тестування проводилася перевірка коректності виконання DAG без помилок. Для цього були змодельовані кілька сценаріїв, у яких перевірялись різні типи DNS-записів, зокрема A, MX та TXT. Завдяки параметризації PythonOperator у самому DAG-файлі, перевірка окремих доменів відбувалася незалежно, що дозволяло створювати декілька задач у межах одного DAG, не змінюючи при цьому основну логіку. DAG було активовано через веб-інтерфейс Apache Airflow, після чого спостерігався процес виконання: етапи

збирання даних, вставки в базу та обробки логів. Завдяки наявності логування всередині Airflow, перевірка правильності обробки відбувалась безпосередньо через вкладку Task Logs (див. рисунок 3.4).

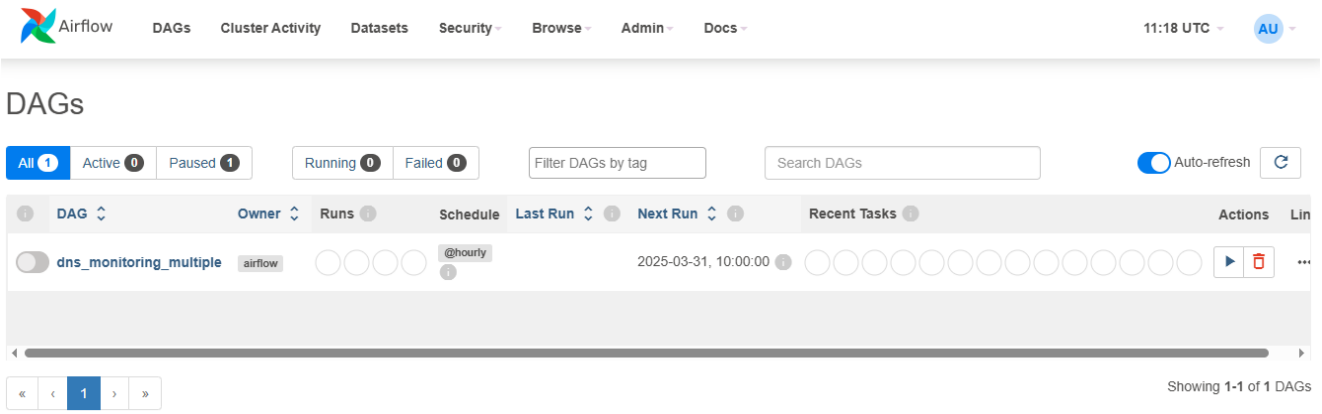


Рисунок 3.4 – DAG у списку Apache Airflow

Після підтвердження коректної роботи DAG із одиничним доменом було проведено тестування системи із розширеним набором доменів. Для цього DAG був модифікований таким чином, щоб створювати задачі циклічно з використанням списку доменів. Це дозволило автоматизовано протестувати систему на вибірці з 100 доменних імен, серед яких були як популярні (google.com, amazon.com), так і спеціально створені тестові піддомени з контрольованими DNS-записами. Метою цього етапу було перевірити, як система поводиться в умовах навантаження, коли одночасно обробляється велика кількість запитів (див. рисунок 3.5).

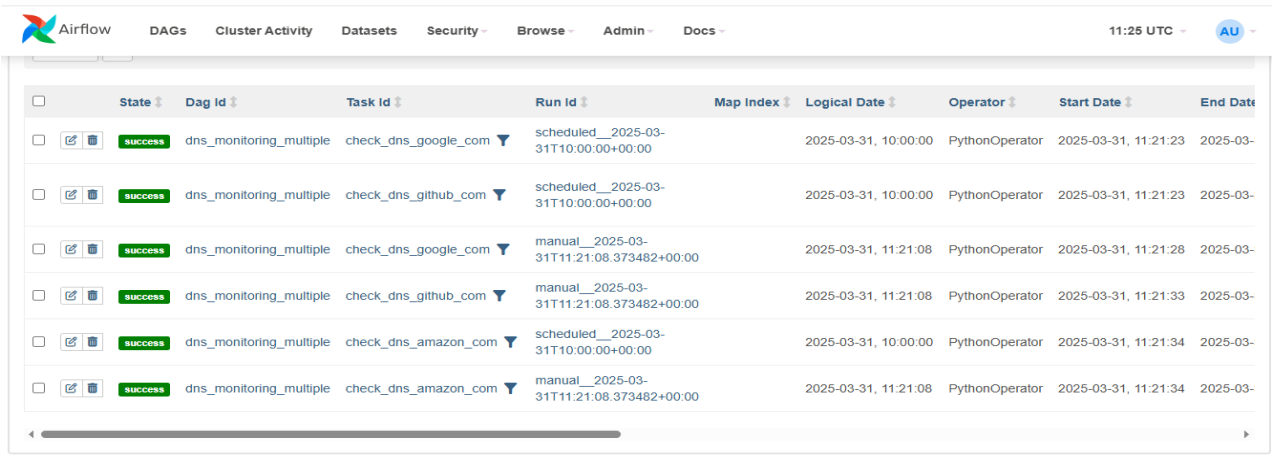


Рисунок 3.5 – Виконанням DAG

Спостереження за виконанням DAG при великій кількості задач дало змогу виявити, що Apache Airflow із Celery Executor коректно розподіляє задачі між воркерами. При використанні одного worker весь DAG виконувався протягом приблизно 3,5 хвилини. Після запуску другого воркера час зменшився вдвічі, що свідчить про лінійну масштабованість розробленої архітектури. Це є надзвичайно важливим аспектом, адже в реальних умовах кількість доменів, які потрібно моніторити, може досягати кількох тисяч.

Окремо проводилось тестування ситуацій з відсутністю відповіді DNS-серверів або з затримками. Для цього частина доменів була спеціально недоступна, а DNS-запити повертали помилки типу NXDOMAIN або SERVFAIL. У таких випадках система не виходила з ладу, а коректно обробляла виняткову ситуацію. Це свідчить про стійкість реалізації функції check_dns, яка була доповнена блоком обробки виключень, що запобігав аварійному завершенню задач. У логах з’являлось відповідне повідомлення про тип помилки, що значно спрощує її діагностику (див. рисунок 3.6).

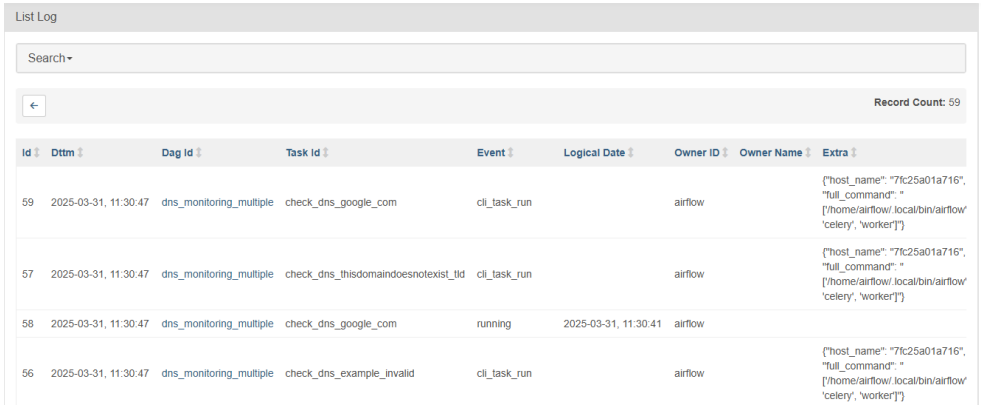


Рисунок 3.6 – Лог задачі DAG з обробкою помилки NXDOMAIN під час запиту до неіснуючого домену

Ще одним важливим аспектом тестування стало перевірка записів у базі даних. Після кожного запуску DAG перевірялась наявність нових рядків у таблиці dns_log, відповідність даних полям, коректність типу запису, дати і часу. Дані виводилися у pgAdmin, що дало змогу візуально впевнитись у правильності

структури і змісту збережених записів. Також здійснювалася перевірка на дублікати, що дозволило переконатись у тому, що система не дублює результати перевірок, якщо записи не змінювались.

Для оцінки продуктивності було виконано серію замірів часу виконання DAG при різних обсягах задач: 10, 50, 100, 200 доменів. Час виконання зростав лінійно, що підтвердило ефективність обраної архітектури з підтримкою Celery Worker. Зі зростанням кількості доменів збільшувався лише час збирання, при цьому інші етапи (вставка в базу, логування) залишались приблизно сталими. Це дозволяє робити висновок про можливість масштабування системи без істотних втрат у продуктивності.

Крім основного сценарію моніторингу, було проведено тестування вручну доданих змін у DNS-записи тестових доменів. Наприклад, для піддомену test.mydomain.com спочатку був виставлений запис типу A з IP-адресою 1.1.1.1, а згодом – змінений на 8.8.8.8. DAG фіксував цю зміну, що відображалось у логах, а також могло використовуватись для ініціації сповіщення, якщо цей функціонал буде реалізовано повністю. Таким чином було доведено здатність системи до виявлення змін у реальному часі, що є ключовим елементом безпеки та надійності моніторингу.

Ще один напрям тестування стосувався поведінки Airflow у випадку перезапуску сервісів. Після примусового зупинення контейнерів webserver та scheduler всі процеси успішно відновлювались без втрати інформації. Журнали задач зберігались у папці logs, а всі дані у PostgreSQL залишались доступними. Це свідчить про те, що система є не лише стійкою до помилок під час виконання задач, але й здатна до відновлення у випадках аварій.

Отже, результати тестування показали, що запропоноване рішення є стабільним, гнучким і масштабованим. Воно демонструє надійність при обробці великої кількості одночасних DNS-запитів, а також правильну логіку роботи з базою даних. Архітектура побудована таким чином, що її можна розширювати як кількісно (додати нові домени), так і функціонально (додати логіку сповіщень, порівняння історичних даних, тощо). Продуктивність системи залишається

високою навіть при збільшенні обсягів задач, а модульність реалізації дозволяє без значних змін адаптувати її під різні типи інфраструктури.

У сукупності всі ці фактори свідчать про практичну придатність розробленої системи для реального застосування в організаціях, де критично важливо мати надійний контроль над станом DNS-записів і оперативно реагувати на будь-які зміни. Система може слугувати базовим компонентом у ширшій платформі інформаційної безпеки або бути використаною як самостійне рішення для моніторингу й аудиту DNS.

3.4 Реальні сценарії використання та реагування

Після розгортання та тестування системи моніторингу DNS-записів важливим етапом стало моделювання та опис практичних сценаріїв її використання в умовах реальної інфраструктури. Потреба у постійному контролі за станом DNS-записів виникає у компаній, що мають складну цифрову інфраструктуру, активно використовують власні домени, піддомени, публічні IP-адреси, а також розміщують частину сервісів у хмарних середовищах або в окремих ізольованих зонах. У таких умовах стабільність і безперервність доменної розв'язки безпосередньо впливають на доступність цифрових сервісів, що обслуговують як клієнтів, так і внутрішні бізнес-процеси.

Один із поширених сценаріїв використання системи – це моніторинг DNS-записів компаній, які використовують власні DNS-сервери для гнучкого управління записами з кодової бази. У таких випадках процес налаштування доменної зони може здійснюватися через інфраструктурний код, зокрема за допомогою інструментів на кшталт Terraform. Це дає можливість програмно керувати створенням, оновленням або видаленням записів без ручного втручання через панель керування провайдера. Щоб реалізувати таку архітектуру, компанія вказує у реєстрі доменів записи типу NS, які вказують на IP-адресу її власного DNS-сервера. Таким чином, запити до домену, які

надходять від клієнтів, перенаправляються на внутрішній DNS-сервер для подальшої розв'язки.

Однак такий підхід має певні ризики, пов’язані з можливістю зміни або втрати NS-запису у реєстратора. Провайдер, який обслуговує домен, має право або технічну можливість періодично перегенеровувати DNS-записи, особливо у випадку автоматичного оновлення записів або конфлікту у внутрішніх політиках безпеки. У таких випадках запис типу NS може бути перезаписаний на стандартний або інший IP-адрес, що призведе до втрати зв'язку між клієнтом і внутрішнім DNS-сервером. Результатом є недоступність ресурсів компанії для користувачів, а також збої у внутрішній взаємодії між сервісами.

Ще одним типовим кейсом є ситуація, коли компанія має великий парк мікросервісів, кожен з яких прив'язаний до окремого піддомену. Такі сервіси можуть бути розгорнуті у різних середовищах – локальному дата-центрі, AWS, Google Cloud або навіть у Kubernetes-кластерах. У таких умовах оновлення або перерозподіл записів типу A, CNAME чи TXT між сервісами часто відбувається автоматично, наприклад, при деплойменті нової версії. У разі помилки або затримки у синхронізації записів система може продовжити видавати застарілий IP-адрес, або навпаки – не мати відповідного запису взагалі. Розроблене рішення дозволяє швидко виявити таку проблему, порівнюючи поточний стан запису з попереднім, і сигналізувати про зміну, яка не відповідає очікуваній конфігурації.

У сфері кібербезпеки моніторинг DNS також грає критичну роль. Один із реальних сценаріїв – це виявлення атак типу DNS hijacking, коли зловмисники змінюють запис домену на свій IP-адрес, перенаправляючи користувачів на підроблений сайт. Подібна атака може бути реалізована як через компрометацію облікових даних у панелі керування провайдера, так і через вразливість у самому DNS-сервері. Завдяки регулярному моніторингу таких змін система дозволяє оперативно виявити появу невідомого IP у записах типу A або підозрілих значень у MX чи TXT записах. Така автоматична перевірка є особливо цінною для організацій, які працюють у фінансовій, державній або критичній інфраструктурі, де швидкість реагування напряму впливає на безпеку.

Ще одна категорія сценаріїв стосується контролю за втратою доступу до піддоменів через помилки в ланцюжку делегацій. Наприклад, якщо основний домен делегований на інший DNS-сервер, а той, у свою чергу, посилається на ще один сервер, при збоях у будь-якій ланці ланцюжка розв'язка може припинитися. Такі проблеми складно виявити вручну, особливо при великій кількості піддоменів. Впроваджена система дозволяє визначити, на якому саме етапі відбувається збій, та надає логування з даними про час останньої відповіді, тип помилки, а також IP-адресу, що повертається у відповідь.

Крім реактивного сценарію – виявлення змін і аномалій – система може використовуватись і в проактивному режимі. Наприклад, при плановому оновленні DNS-зон або переведенні інфраструктури з одного провайдера на іншого (наприклад, із Cloudflare на AWS Route53) компанія може налаштувати додатковий DAG, який протягом певного періоду буде перевіряти доступність записів одночасно з кількох локацій або серверів. Це дозволяє переконатись, що зміни вже відбулись глобально (усі DNS-кеші оновлені), а також уникнути ситуацій, коли один з іменних серверів ще повертає застарілу інформацію.

У межах внутрішньої мережі компанії моніторинг DNS може допомогти виявити технічні помилки конфігурації, наприклад, ситуацію, коли DNS-запис типу MX вказує на сервер пошти, який не відповідає або налаштований з порушенням SPF-записів. У такому випадку зовнішні поштові сервіси можуть відмовити у доставці повідомлень, а клієнти – не отримати відповіді від служби підтримки. Автоматизоване рішення дає можливість швидко виявити такі помилки ще до того, як вони позначаться на користувачах.

Загалом, запропонована система моніторингу може бути гнучко адаптована під потреби конкретної організації – як невеликої команди з кількома сервісами, так і великої інфраструктурної компанії, що оперує сотнями доменів та піддоменів. Наявність механізму автоматичного збору та збереження DNS-записів дозволяє не лише реагувати на зміни, а й формувати аналітику: визначати частоту змін, середній час життя записів, а також створювати звіти для відділів безпеки або адміністраторів.

Отже, розроблена система демонструє високу ефективність у вирішенні практичних задач – від виявлення критичних змін до попередження атак і виявлення технічних проблем у конфігурації DNS-інфраструктури. Її застосування дозволяє значно скоротити час простоїв, зменшити ризики втрати користувачів або клієнтів та підвищити рівень технічної зрілості компанії у сфері управління мережею.

3.5 Висновки до третього розділу

У третьому розділі було здійснено безпосередню реалізацію та перевірку працездатності запропонованої системи моніторингу DNS-записів з використанням платформи Apache Airflow. Розробка розпочалась із детального налаштування середовища розгортання, що включало контейнеризацію компонентів за допомогою Docker та створення ізольованого сервісного оточення, до якого увійшли веб-інтерфейс Apache Airflow, база даних PostgreSQL, брокер повідомлень Redis, а також необхідні бібліотеки для виконання DNS-запитів.

На базі створеного середовища було реалізовано DAG-процес, який здійснює регулярну перевірку заданого списку доменів, аналізує відповідь DNS-серверів, обробляє винятки у випадку збоїв та зберігає результати моніторингу у базі даних. Завдяки використанню Celery Executor вдалося досягти масштабованості та ефективного паралельного виконання задач, що дозволяє обробляти значну кількість доменів у межах одного циклу DAG.

Під час етапу тестування система показала стабільну роботу, стійкість до непередбачених ситуацій, таких як відсутність відповіді від DNS-серверів, некоректні або неіснуючі домени. Всі подібні випадки були успішно оброблені без аварійного завершення задач. Було перевірено збереження результатів у базі даних, що дало змогу сформувати історію змін, а також оцінити працездатність системи при збільшенні обсягів даних.

У розділі також було змодельовано декілька реальних сценаріїв використання системи – від стандартного моніторингу публічних доменів до виявлення критичних змін у записах NS, що можуть впливати на доступність сервісів компанії. Окрему увагу було приділено кейсам, пов’язаним з інформаційною безпекою, втручанням у DNS-записи, а також виявленням помилок конфігурації у складних інфраструктурах.

Загалом, результати реалізації підтверджують технічну доцільність і практичну ефективність обраного підходу. Побудована система є надійною, гнучкою до масштабування та придатною для інтеграції в існуючу інфраструктуру будь-якої організації, яка має потребу в постійному контролі DNS-записів. Вона забезпечує як оперативне реагування на зміни, так і можливість для подальшого розвитку – наприклад, шляхом додавання функцій сповіщення, порівняння історичних даних, побудови звітів та інтеграції з системами інформаційної безпеки.

РОЗДІЛ 4. ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ ТА ОХОРОНИ ПРАЦІ

4.1 Основи інформаційної безпеки DNS-систем

DNS – одна з фундаментальних технологій інтернет-інфраструктури, яка забезпечує трансляцію доменних імен у IP-адреси. Уразливість цієї системи є критичною, адже вона може призвести до значних збоїв, втрати довіри користувачів та витоку конфіденційної інформації. З огляду на це, забезпечення інформаційної безпеки DNS-систем є одним із найважливіших напрямів у сфері кіберзахисту.

Уразливості в системах DNS становлять серйозну загрозу для функціонування як окремих сервісів, так і цілих інформаційних систем. Серед основних типів атак, що спрямовані на DNS, виділяють: DNS spoofing, DNS hijacking, DNS tunneling, cache poisoning, DDoS-атаки на DNS-сервери. Кожна з них має свої особливості реалізації, наслідки та шляхи протидії.

DNS spoofing – підміна відповіді від DNS-сервера з метою спрямування користувача на фальшивий ресурс, який візуально імітує легітимний сайт. Метою такої атаки часто є викрадення облікових даних, розповсюдження шкідливого програмного забезпечення або компрометація особистої інформації. DNS hijacking передбачає зміну маршруту DNS-запиту, часто через зміну записів NS у панелі керування доменом. Наслідком є направлення запитів на інший, потенційно контрольований зловмисником сервер.

Щоб ефективно протидіяти цим загрозам, використовуються різноманітні технічні та організаційні засоби. Одним із ключових механізмів захисту є DNSSEC – розширення протоколу DNS, що дозволяє верифікувати цілісність і достовірність DNS-записів за допомогою криптографічних підписів. Завдяки цьому користувач може бути впевнений, що відповідь на запит дійсно надійшла від авторитетного джерела і не була підмінена.

Іншим напрямом захисту є реалізація багаторівневої системи моніторингу змін DNS-записів. Це дозволяє виявляти спроби несанкціонованої модифікації

записів у реальному часі. Таке рішення було реалізоване в межах даної дипломної роботи. Система моніторингу на базі Apache Airflow дозволяє регулярно опитувати DNS-сервери, зберігати історію змін, фіксувати аномалії та потенційні інциденти безпеки.

Також важливо обмежити доступ до DNS-зон. Управління записами має здійснюватися лише авторизованими користувачами через захищені канали зв'язку. У середовищах підвищеної чутливості рекомендується розділяти зони зовнішнього та внутрішнього DNS, що унеможливлює витік інформації про внутрішню структуру організації [26].

Крім технічних засобів, важливою складовою є політики безпеки та підготовка персоналу. Регулярне оновлення паролів, використання двофакторної автентифікації, контроль прав доступу та аудит змін – усе це підвищує загальний рівень захищеності інфраструктури.

4.2 Охорона праці при роботі з серверним обладнанням

Робота з серверним обладнанням та інфраструктурними елементами вимагає чіткого дотримання норм охорони праці та техніки безпеки. У сучасних ІТ-організаціях сервери можуть розміщуватися як у спеціалізованих дата-центрах, так і безпосередньо в офісних приміщеннях. У будь-якому випадку, обслуговування серверного устаткування супроводжується низкою потенційних небезпек: електричні ураження, перегрів обладнання, пожежі, травмування при монтажі або демонтажі системних блоків, а також вплив електромагнітного випромінювання.

Головним аспектом безпечної роботи є дотримання електротехнічних норм. Перед проведенням робіт обладнання повинно бути відключене від електромережі, а сам працівник зобов’язаний використовувати захисні засоби – антистатичні браслети, гумові килимки, діелектричні рукавички. При підключенні чи обслуговуванні джерел безперебійного живлення (UPS),

особливо високопотужних, необхідно дотримуватись вимог щодо заземлення та уникати роботи у вологому середовищі.

Серверні приміщення мають відповідати нормативам температурного режиму – зазвичай це 18 – 22°C при вологості 40 – 60%. Вентиляційна система повинна бути розрахована на безперервну роботу і регулярно очищуватись. Наявність димових датчиків, вогнегасників та систем автоматичного пожежогасіння є обов'язковою умовою. Крім того, важливо, щоб усі кабелі були розміщені таким чином, щоб уникнути перешкоджання руху персоналу, запобігти спотиканню або замиканню внаслідок натягу [27].

До обслуговування серверного обладнання мають допускатись лише співробітники, що пройшли відповідний інструктаж. Це передбачає знання не лише правил техніки безпеки, а й основ пожежної безпеки, дій у разі аварійного відключення електроенергії, евакуації персоналу та перших дій при ураженні струмом.

Окремо варто відзначити ергономічні аспекти. Працівники, що працюють з моніторами, клавіатурами та сервісними інструментами, повинні мати обладнані робочі місця, які не призводять до перенапруги очей або опорно-рухового апарату. Дотримання режиму праці та відпочинку, наявність природного освітлення або якісного штучного освітлення також позитивно впливають на зниження ризиків [28].

Отже, охорона праці при роботі з серверним обладнанням – це комплекс заходів, які спрямовані на збереження життя і здоров'я працівників, запобігання нещасним випадкам, а також забезпечення надійності функціонування технічної інфраструктури. Правильна організація таких процесів підвищує загальну ефективність IT-відділу та знижує рівень операційних ризиків.

4.3 Захист персональних даних у процесі моніторингу

У межах реалізації систем моніторингу DNS-записів, особливо у корпоративному середовищі, постає питання обробки персональних та

конфіденційних даних. У сучасних умовах цифрової трансформації практично будь-яка інформаційна система потенційно взаємодіє з даними, що можуть ідентифікувати особу або організацію. У разі систем, що контролюють DNS-записи, це можуть бути як прямі, так і непрямі ідентифікатори – IP-адреси, поштові сервери, доменні імена з прив'язкою до користувачів або внутрішніх сервісів.

Законодавство України, а також європейське регулювання в межах GDPR визначають чіткі вимоги до захисту персональних даних. Будь-яка система, що здійснює збирання, обробку, зберігання або передавання таких даних, повинна забезпечувати їхню конфіденційність, цілісність і доступність. В умовах моніторингу DNS це означає, що результати перевірки не мають містити зайвих відомостей, які можуть бути використані для деанонімізації користувача або компрометації сервісу [29].

Для дотримання вимог необхідно впроваджувати заходи технічного та організаційного характеру. Зокрема, усі з’єднання між компонентами системи мають бути захищені за допомогою шифрування (наприклад, TLS). База даних, де зберігаються записи моніторингу, повинна мати контроль доступу та механізми авторизації. Усі операції зі зчитування, запису та видалення мають логуватись для забезпечення прозорості та підзвітності.

Також слід розробити політику збереження даних. Інформація, що не є критичною, повинна автоматично видалятись через визначений термін – наприклад, через 30 або 90 днів. Для зниження ризиків витоку дозволено застосування псевдонімізації – тобто заміни конкретних імен або ідентифікаторів на технічні маркери, які не мають значення поза межами системи [30].

Крім технічного захисту, важливим є навчання персоналу, який працює з системою моніторингу. Співробітники повинні бути поінформовані про правила поводження з персональними даними, можливі юридичні наслідки порушення та алгоритми дій у разі виявлення витоку або інциденту.

4.4 Висновки до четвертого розділу

У результаті аналізу ключових аспектів, пов’язаних з безпекою та охороною праці в рамках реалізації системи моніторингу DNS, можна зробити висновок, що успішне впровадження таких систем потребує багатогранного підходу. Питання безпеки не обмежуються лише захистом від кіберзагроз – вони охоплюють також фізичний захист персоналу, дотримання законодавства, впровадження політик управління ризиками.

Забезпечення інформаційної безпеки DNS є критично важливим у зв’язку з поширенням атак на DNS-інфраструктуру, зростанням складності мережевих конфігурацій та необхідністю гарантованої стабільності обслуговування користувачів. При цьому система, реалізована в межах даного проєкту, дозволяє не лише виявляти проблеми, а й виступати активним інструментом управління ризиками.

Так само важливо забезпечити безпечні умови праці для спеціалістів, що займаються налаштуванням і обслуговуванням технічної інфраструктури. Правильна організація робочого простору, дотримання техніки безпеки та забезпечення ергономічних умов позитивно впливають на ефективність і безпеку експлуатації ІТ-систем.

Не менш важливою є відповідальність за збереження та обробку персональних і конфіденційних даних. Захист такої інформації – це не лише технічна задача, а й юридичне зобов’язання. Рішення, що інтегрується у реальне середовище, повинно враховувати ці вимоги з самого початку проєктування.

Загалом, дотримання вимог безпеки й охорони праці створює передумови для довготривалої, стабільної та безпечної роботи систем моніторингу в організаціях різного масштабу. Це підвищує не лише технічний рівень рішень, а й сприяє формуванню культури відповідального поводження з даними та інфраструктурою.

ВИСНОВКИ

У результаті виконання даного дослідження було реалізовано повноцінну автоматизовану систему моніторингу DNS-записів на базі платформи Apache Airflow, яка дозволяє виявляти критичні зміни у доменних зонах, здійснювати збереження історії записів, аналізувати стабільність інфраструктури та своєчасно інформувати відповідальних осіб про потенційні ризики або відхилення. Проведене дослідження охопило теоретичне обґрунтування, архітектурне проектування, реалізацію, тестування системи, а також аналіз питань інформаційної безпеки та охорони праці в умовах сучасної цифрової інфраструктури.

У першому розділі було виконано аналіз предметної області, розкрито важливість надійного функціонування системи доменних імен, обґрунтовано актуальність проблеми моніторингу DNS-записів у контексті забезпечення безперервної роботи онлайн-сервісів та захисту інформації. Було проведено дослідження існуючих рішень, серед яких як хмарні сервіси, так і локальні системи моніторингу, що дозволило виявити їхні переваги та недоліки. Основна увага була приділена обґрунтуванню доцільності створення власного кастомного рішення, здатного задовольнити специфічні вимоги інфраструктури, забезпечити гнучкість та глибоку інтеграцію з внутрішніми процесами. У межах розділу також сформульовано вимоги до системи та визначено Apache Airflow як базову платформу для реалізації автоматизованих процесів перевірки DNS.

У другому розділі зосереджено увагу на проєктуванні архітектури системи. Побудовано модель взаємодії основних компонентів, у тому числі модулів збору даних, обробки, порівняння, збереження та сповіщення. Архітектура реалізована із застосуванням принципів ETL (Extract, Transform, Load) та орієнтована на гнучке масштабування. Запропоновано структуру бази даних, здатну зберігати як поточні значення записів, так і їхні історичні зміни, що дозволяє виконувати аналітичну обробку даних. Завдяки використанню Apache Airflow у поєднанні з контейнеризацією через Docker, система отримала високу стабільність та

портативність. Запропонований підхід дозволяє адаптувати систему до будь-якого середовища – локального чи хмарного.

Третій розділ було присвячено практичній реалізації розробленої архітектури. Було виконано повноцінне розгортання системи, налаштовано робоче середовище, створено та запущено DAG-процеси, що виконують перевірку DNS-записів для вибраного списку доменів. Завдяки використанню Celery Executor досягнуто паралельності виконання задач, що істотно підвищує продуктивність та дозволяє масштабувати систему в залежності від навантаження. Проведено тестування системи за різними сценаріями, включаючи навмисні збої, відсутність відповіді DNS-серверів, помилки у записах тощо. Було підтверджено, що система коректно обробляє винятки, зберігає результати перевірок у базі даних, забезпечує високу стабільність та дозволяє швидко реагувати на зміни. Додатково змодельовано реальні сценарії застосування системи в умовах організаційної інфраструктури – як у сфері обслуговування зовнішніх клієнтів, так і для внутрішніх потреб безпеки.

У четвертому розділі висвітлено аспекти безпеки та охорони праці, що супроводжують впровадження подібних систем. Було охарактеризовано основні загрози для DNS-інфраструктури, обґрунтовано заходи для їх запобігання, зокрема впровадження DNSSEC, обмеження доступу, криптографічний захист даних та регулярний аудит змін. Визначено умови безпечної експлуатації серверного обладнання, включно з фізичним захистом, вентиляцією, пожежною безпекою та інструктажем персоналу. Розглянуто правові та етичні аспекти обробки персональних даних, сформульовано вимоги до їх зберігання, знеособлення, логування доступу та дотримання законодавства, зокрема в контексті українського Закону «Про захист персональних даних» та європейського регламенту GDPR.

Загалом, результати дослідження засвідчують доцільність обраного підходу та високу практичну цінність реалізованої системи. Побудоване рішення є адаптивним, стійким до збоїв, легко налаштовується та масштабується. Воно може бути впроваджене як у невеликих командах, так і в масштабних

організаціях, що мають розгалужену інфраструктуру. Запропоноване програмне рішення дозволяє істотно підвищити рівень контролю за станом DNS-записів, зменшити ризики втручання, скоротити час простоїв та покращити загальну інформаційну безпеку. Водночас гнучкість та відкритість архітектури забезпечують можливість подальшого розвитку проєкту, його інтеграції з системами сповіщень, SIEM-рішеннями, платформами інформаційної аналітики, а також розширення функціональності відповідно до зростаючих потреб бізнесу або організації.

ПЕРЕЛІК ДЖЕРЕЛ

1. Що таке DNS і DNS-записи? [Електронний ресурс]. – URL: https://hyperhost.ua/uk/wiki/chto-takoe-dns-i-dns-zapisi (дата звернення 08.05.2025)

2. Як працює DNS [Електронний ресурс]. – URL: https://hostiq.ua/blog/ukr/how-does-dns-work/ (дата звернення 08.05.2025)

3. Як перевірити та змінити свій DNS [Електронний ресурс]. – URL: https://trustytech.io/ua/blog/yak-perevirity-ta-zminyty-svij-dns/ (дата звернення 08.05.2025)

4. Apache Airflow [Електронний ресурс]. – URL: https://airflow.apache.org/ (дата звернення 10.05.2025)

5. ТОП сервісів для моніторингу доступності веб-сайтів і серверів [Електронний ресурс]. – URL: https://kr-labs.com.ua/blog/top-uptime-monitoring-tools (дата звернення 10.05.2025)

6. Сервіс перевірки записів DNS [Електронний ресурс]. – URL: https://thehost.ua/ua/domains/dnslookup (дата звернення 16.05.2025)

7. Fryz, M., & Mlynko, B. (2020). Properties of stationarity and cyclostationarity of conditional linear random processes.

8. DNSChecker [Електронний ресурс]. – URL: https://dnschecker.org/ (дата звернення 16.05.2025)

9. Pingdom [Електронний ресурс]. – URL: https://www.pingdom.com/ (дата звернення 16.05.2025)

10. UptimeRobot [Електронний ресурс]. – URL: https://uptimerobot.com/ (дата звернення 17.05.2025)

11. Zabbix [Електронний ресурс]. – URL: https://www.zabbix.com/ (дата звернення 22.05.2025)

12. Nagios [Електронний ресурс]. – URL: https://www.nagios.org/ (дата звернення 22.05.2025)

13. PRTG Network Monitor [Електронний ресурс]. – URL: https://www.paessler.com/prtg (дата звернення 23.05.2025)

14. SecurityTrails [Електронний ресурс]. – URL: https://securitytrails.com/ (дата звернення 23.05.2025)

15. Farsight DNSDB [Електронний ресурс]. – URL: https://www.farsightsecurity.com/ (дата звернення 23.05.2025)

16. Stadnyk, M., Fryz, M., Zagorodna, N., Muzh, V., Kochan, R., & Nikodem, J. (2022). Steady state visual evoked potential classification by modified KNN method.

17. AWS Lambda [Електронний ресурс]. – URL: https://aws.amazon.com/lambda/ (дата звернення 23.05.2025)

18. Google Cloud Functions [Електронний ресурс]. – URL: https://cloud.google.com/functions (дата звернення 23.05.2025)

19. Apache Airflow: стандартизація, автоматизація та моніторинг процесу ETL [Електронний ресурс]. – URL: https://surl.li/bslqrw (дата звернення 23.05.2025)

20. Directed Acyclic Graph (DAG) [Електронний ресурс]. – URL: https://hazelcast.com/foundations/distributed-computing/directed-acyclic-graph/ (дата звернення 23.05.2025)

21. Типи DNS-записів домена: NS, A, AAAA, CNAME, TXT, ALIAS, SRV [Електронний ресурс]. – URL: https://cityhost.ua/uk/blog/tipi-dns-zapisiv-domena-ns-a-aaaa-cname-txt-alias-srv.html (дата звернення 23.05.2025)

22. Transfer data from Big query to Amazon S3 using Airflow [Електронний ресурс]. – URL: https://stackoverflow.com/questions/69056826/transfer-data-from-big-query-to-amazon-s3-using-airflow (дата звернення 24.05.2025)

23. Поширення DNS: Чому виникають затримки і як їх вирішити [Електронний ресурс]. – URL: https://www.spaceship.com/uk/blog/dns-propagation-delays/ (дата звернення 24.05.2025)

24. Інтеграція Slack + Телеграм [Електронний ресурс]. – URL: https://apix-drive.com/ua/slack/telegram (дата звернення 25.05.2025)

25. Розуміння DNS TTL [Електронний ресурс]. – URL: https://www.vpnunlimited.com/ua/help/cybersecurity/dns-

ttl?srsltid=AfmBOoowAa6g0P3T6rrUadk9c7D9CjJdBBNZcuWMcIKiVx9jzpK3luGX (дата звернення 25.05.2025)

26. How to set up A-record and rDNS record [Електронний ресурс]. – URL: https://snov.io/knowledgebase/how-to-set-up-a-record-and-rdns-record/ (дата звернення 25.05.2025)

27. Шимчук, Г. В., Маєвський, О. В., & Назаревич, О. Б. (2016). Методичні вказівки до виконаня практичних робіт з дисципліни Історія науки і техніки для студентів освітнього рівня бакалавр спеціальності 125– Кібербезпека.

28. Lupenko, S., Lytvynenko, I., & Stadnyk, N. (2020). Method for reducing the computational complexity of processing discrete cyclic random processes in digital data analysis systems.

29. Duda, O., Matsiuk, O., Kunanets, N., Pasichnyk, V., & Veretennikova, N. (2020). Selection of effective methods of big data analytical processing in information systems of smart cities. CEUR Workshop Proceedings, 2643, 68–78.

30. PostgreSQL: The World's Most Advanced Open Source Relational Database [Електронний ресурс]. – URL: (дата звернення 26.05.2025)

31. Федчук Т.Б. ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ БЕЗПЕЧНОГО ДОСТУПУ ДО РЕСУРСІВ DNS НА БАЗІ ML-ТРЕНОВАНИХ МОДЕЛЕЙ ІДЕНТИФІКАЦІЇ ТРАФІКУ [Електронний ресурс]. – URL: https://www.sworldjournal.com/index.php/swj/article/view/swj21-01-015 (дата звернення 26.05.2025)

32. Вимоги до серверної (серверному приміщенню, апаратної) [Електронний ресурс]. – URL: https://shop.hypernet.com.ua/trebovaniya-k-servernoy-komnate/ (дата звернення 28.05.2025)

33. Інструкція з охорони праці для фахівця з інформаційних технологій [Електронний ресурс]. – URL: https://pro-op.com.ua/article/1065-nstruktsya-z-ohoroni-prats-dlya-fahvtsya-z-nformatsynih-tehnology (дата звернення 02.06.2025)

34. ЗАХИСТ ПЕРСОНАЛЬНИХ ДАНИХ: ПРАВОВЕ РЕГУЛЮВАННЯ ТА ПРАКТИЧНІ АСПЕКТИ [Електронний ресурс]. – URL:

https://ombudsman.gov.ua/storage/app/media/uploaded-files/Handbook_Pers_Data_Protect_2021.pdf (дата звернення 04.06.2025)

35. Головацький Н.Т. ПРАВОВЕ РЕГУЛЮВАННЯ ЗАХИСТУ ПЕРСОНАЛЬНИХ ДАНИХ: GDPR ТА ЗАКОНОДАВСТВО США, КАНАДИ Й УКРАЇНИ [Електронний ресурс]. – URL: https://doi.org/10.24144/2307-3322.2024.85.2.42 (дата звернення 05.06.2025)

ДОДАТКИ

Додаток А

Лістинг повного коду DAG для моніторингу DNS-записів

```
    # Імпорт основних бібліотек Airflow
    from airflow import DAG
    from airflow.operators.python import PythonOperator
    from datetime import datetime
    import dns.resolver
    import psycopg2

    # Основна функція для перевірки DNS-запису та збереження в базу
даних
    def check_dns(domain, record_type):
        try:
            # Виконання DNS-запиту
            result = dns.resolver.resolve(domain, record_type)
            values = [r.to_text() for r in result]

            # З'єднання з базою PostgreSQL, вставка результату у
таблицю
            conn = psycopg2.connect(
                dbname="airflow",
                user="airflow",
                password="airflow",
                host="postgres"
            )
            cur = conn.cursor()
            cur.execute(
                "INSERT INTO dns_log (domain, type, values,
checked_at) VALUES (%s, %s, %s, %s)",
                (domain, record_type, str(values), datetime.now())
            )
            conn.commit()
            cur.close()
            conn.close()
        except Exception as e:
            # Обробка винятків (наприклад, NXDOMAIN, SERVFAIL)
            print(f"Error checking {domain}: {e}")

    # Параметри DAG: дата початку, кількість спроб тощо
    default_args = {
        'start_date': datetime(2024, 1, 1),
        'retries': 1,
    }

    # Оголошення DAG – визначення ідентифікатора, розкладу та опису
    with DAG(
        dag_id='dns_monitoring_multiple',
```

```
default_args=default_args,
         schedule_interval='@hourly',  # Запуск щогодини
         catchup=False,
         description='Моніторинг кількох доменів',
     ) as dag:

         # Список доменів для перевірки
         domains = ['google.com', 'amazon.com', 'github.com']
         record_type = 'A'

         # Для кожного домену створюється окрема задача в DAG
         for domain in domains:
             PythonOperator(
                 task_id=f'check_dns_{domain.replace(".", "_")}',
                 python_callable=check_dns,
                 op_kwargs={'domain':    domain,    'record_type':
record_type}
             )
```

Додаток Б

Файл конфігурації

```
version: '3'

services:
  postgres:
    image: postgres:13
    environment:
      POSTGRES_USER: airflow
      POSTGRES_PASSWORD: airflow
      POSTGRES_DB: airflow
    volumes:
      - ./pgdata:/var/lib/postgresql/data
    restart: always

  redis:
    image: redis:latest
    restart: always

  airflow-webserver:
    image: apache/airflow:2.8.1
    command: webserver
    depends_on:
      - postgres
      - redis
    ports:
      - "8080:8080"
    environment:
      AIRFLOW__CORE__EXECUTOR: CeleryExecutor
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__RESULT_BACKEND:
db+postgresql://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__BROKER_URL: redis://redis:6379/0
      AIRFLOW__CORE__FERNET_KEY: ''
      AIRFLOW__CORE__LOAD_EXAMPLES: 'False'
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
    restart: always

  airflow-scheduler:
    image: apache/airflow:2.8.1
    command: scheduler
    depends_on:
      - airflow-webserver
    environment:
      AIRFLOW__CORE__EXECUTOR: CeleryExecutor
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
```

```
    AIRFLOW__CELERY__RESULT_BACKEND:
db+postgresql://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__BROKER_URL: redis://redis:6379/0
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
    restart: always

  airflow-worker:
    image: apache/airflow:2.8.1
    command: celery worker
    depends_on:
      - airflow-scheduler
    environment:
      AIRFLOW__CORE__EXECUTOR: CeleryExecutor
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__RESULT_BACKEND:
db+postgresql://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__BROKER_URL: redis://redis:6379/0
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
    restart: always

  airflow-init:
    image: apache/airflow:2.8.1
    depends_on:
      - postgres
      - redis
    entrypoint: /bin/bash -c "airflow db upgrade && airflow
users create --username admin --password admin --firstname admin --
lastname admin --role Admin --email admin@example.com"
    environment:
      AIRFLOW__CORE__EXECUTOR: CeleryExecutor
      AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__RESULT_BACKEND:
db+postgresql://airflow:airflow@postgres/airflow
      AIRFLOW__CELERY__BROKER_URL: redis://redis:6379/0
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
```