Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

Ткачука Тимура Сергійовича

на здобуття ступеня вищої освіти Бакалавра

Система захисту мікросервісних застосунків від “атаки на відмову”

Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітня програма Кібербезпека

Шифр КРБКБ.200117.20.01.17 ПЗ

Виконав студент 4 курсу група КБ-20-1 Тимур ТКАЧУК
Керівник ст. викладач, доктор філософії Микола СТЕЦЮК
Нормоконтролер старший викладач Сергій МОСТОВИЙ

До захисту допускаю:
Завідувач кафедри кібербезпеки Юрій КЛЬОЦ

19 06 2024 р.

Хмельницький 2024

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Кібербезпеки

Рівень вищої освіти Бакалавр

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека

Освітня програма Кібербезпека

ЗАТВЕРДЖУЮ

Завідувач кафедри кібербезпеки

Юрій КЛЬОЦ

15 лютого 2024 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Ткачуку Тимуру Сергійовичу

1 Тема роботи Система захисту мікросервісних застосунків від "атаки на відмову"

Керівник роботи Микола СТЕЦЮК

Затверджено наказом ректора університету від 15 лютого 2024 № 8

2 Строк подання студентом кваліфікаційної роботи на кафедру

3 Вихідні дані до роботи Метою даної кваліфікаційної роботи є розробка та впровадження системи захисту мікросервісних застосунків від атак на відмову. Така система повинна забезпечувати виявлення, запобігання та мінімізацію впливу DoS атак на мікросервіси. Перше Я маю зробити дослідження особливостей атак на мікросервіси, оглянути сучасні технології та інструменти захисту від DoS атак, після розробити модель загроз для мікросервісних застосунків, спроектувати систему захисту та реалізувати її.

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1 Теоретична частина; 2 Розробка та впровадження системи захисту мікросервісів від DDoS атак; 3 Реалізація системи захисту від DDoS-атаки.

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Мікросервісна архітектура (КРБКБ.200117.20.01.17 Е8), розроблена мікросервісна архітектура(КРБКБ.200117.20.01.17 Е8),алгоритм роботи(КРБКБ.200117.20.01.17 Е8)

6 Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Мостовий С.В., старший викладач кафедри кібербезпеки		

7 Дата видачі завдання 16 лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
Вибір і затвердження теми кваліфікаційної роботи	17.02.2024	Виконав
Ознайомлення з предметною областю	17.02.2024	
Дослідження існуючих рішень	24.02.2024	
Постановка задачі	04.03.2024	
Визначення загальних принципів рішення задачі	11.03.2024	
Деталізація принципів рішення задачі	16.04.2024	
Розробка проєктних рішень	25.04.2024	
Апробація проєктних рішень	15.05.2024	
Оформлення пояснювальної записки згідно вимог	24.05.2024	
Оформлення графічної частини	03.06.2024	
Захист КР	20.09.2024	

Судент Тимур ТКАЧУК

Керівник кваліфікаційної роботи Микола СТЕЦЮК

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Система захисту мікросервісних застосунків від атаки на відмову».

Автор роботи: Ткачук Тимур Сергійович

Керівник роботи: Стецюк Микола Васильович

Пояснювальна записка: 72с., 11 рис., 40 джерел.

Графічна частина: 3 плакати, 10 презентаційних слайдів.

Мета дипломної роботи: Метою дипломної роботи є розроблення ефективної системи захисту мікросервісних застосунків, здатної проактивно виявляти та нейтралізувати потенційні кіберзагрози, забезпечуючи стабільну та безпечну роботу мікросервісної архітектури.

Ключові слова: кібербезпека, мікросервіси, DDoS-атаки, система захисту, шифрування даних, веб-брандмауер, моніторинг мережевого трафіку, ліцензування, конфіденційні дані.

Об'єктом дослідження є мікросервісна архітектура.

Предметом дослідження є оцінка роботи системи мікросервісних застосунків та розробка власної системи захисту від атаки на відмову.

Під час проведення даного дослідження був використаний метод систематичного огляду літератури для вивчення і аналізу предметної області данного дослідження з текстових джерел інформації та для розробки власного проекту.

20.06.2024 Ткачук

ANNOTATION

Theme of the qualification work: "A system for protecting microservice applications from denial of service attacks".

Author of the work: Tkachuk Timur

Supervisor of the work: Stetsiuk Mykola

Explanatory note: 72 p., 11 figures, 40 sources.

Graphic part: 3 posters, 10 presentation slides.

Purpose of the thesis: The purpose of the thesis is to develop an effective system for protecting microservice applications that can proactively detect and neutralize potential cyber threats, ensuring stable and secure operation of the microservice architecture.

Keywords: cybersecurity, microservices, DDoS attacks, protection system, data encryption, web firewall, network traffic monitoring, licensing, confidential data.

The object of research is microservice architecture.

The subject of the study is to evaluate the operation of a microservice application system and develop its own denial-of-service attack protection system.

During this study, the method of systematic literature review was used to study and analyze the subject area of this study from textual sources of information and to develop its own project.

20.06.2024 Ткачук

ЗМІСТ

ВСТУП

У сучасному цифровому світі безпека інформації стає дедалі важливішою у забезпеченні надійності та цілісності систем. Одним із найпоширеніших видів кібератак є атака на відмову (DDoS), що має на меті перекриття доступу до ресурсів шляхом перенавантаження системи запитами. У зв'язку з цим, розробка та впровадження ефективних систем захисту від DDoS-атак є актуальною проблемою. У даній кваліфікаційній роботі досліджується та аналізується ефективність різних методів захисту від атак на відмову. Проводиться порівняльний аналіз різних підходів до виявлення та мінімізації негативних наслідків DDoS-атак на інформаційні системи. Розглядаються як традиційні методи захисту, так і інноваційні технології, спрямовані на запобігання великим масштабам атак та забезпечення стійкості систем у умовах постійної кіберзагрози. Обрана тема дозволить детально розглянути ключові аспекти захисту від DDoS-атак, виявити сильні та слабкі сторони різних методів, а також розробити рекомендації щодо покращення сучасних систем захисту від кіберзагроз. Кваліфікаційна робота зосереджена на пошуку оптимальних рішень для забезпечення безпеки та стабільності інформаційних систем у сучасному цифровому середовищі.

Метою даної кваліфікаційної роботи є розробка та впровадження системи захисту мікросервісних застосунків від атак на відмову. Така система повинна забезпечувати виявлення, запобігання та мінімізацію впливу DoS атак на мікросервіси. Перше Я маю зробити дослідження особливостей атак на мікросервіси, оглянути сучасні технології та інструменти захисту від DoS атак, після розробити модель загроз для мікросервісних застосунків, спроектувати систему захисту та реалізувати її, після зробити оцінку захисту.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Опис мікросервісної архітектури

Мікросервіси також відомі як «архітектура мікросервісів» – це архітектурний стиль, який структурує програму як набір сервісів, які:

– розгортається самостійно;

– слабко зчеплені.

Послуги зазвичай організовуються навколо бізнес-можливостей. Кожна служба часто належить одній невеликій команді.

Архітектура мікросервісів - це альтернативний патерн, який усуває обмеження монолітної архітектури. Вона передбачає розбиття великого додатку на менші, незалежні сервіси, які можна розробляти, розгортати і масштабувати незалежно. Одним з найпоширеніших прикладів архітектури мікросервісів є патерн API Gateway / Backends for Frontends, який передбачає використання шлюзу API для обробки клієнтських запитів і перенаправлення їх до відповідних мікросервісів. Цей патерн дозволяє створювати відокремлену та масштабовану архітектуру. Іншим важливим патерном в архітектурі мікросервісів є використання асинхронного обміну повідомленнями для міжсервісної комунікації. Цей патерн передбачає, що сервіси взаємодіють, обмінюючись повідомленнями через канали обміну повідомленнями. Це забезпечує вільний зв'язок між сервісами і дозволяє покращити масштабованість та відмовостійкість. Існує також мова патернів для мікросервісів, яка надає набір рекомендацій та найкращих практик для проектування та реалізації архітектур мікросервісів. Вона включає такі патерни, як патерн монолітної архітектури, коли додаток будується як єдиний розгорнутий модуль, і патерн мікросервісної архітектури, коли додаток будується як набір невеликих, незалежних сервісів. Загалом, архітектура мікросервісів пропонує кілька переваг, таких як покращена масштабованість, гнучкість та відмовостійкість. Розбиваючи великий додаток на менші сервіси, організації можуть досягти кращої гнучкості та ремонтопридатності. Однак це також створює проблеми, такі як управління міжсервісною комунікацією та забезпечення узгодженості даних між сервісами.

Плюси мікросервісів:

– прості служби;

– автономність команди;

– конвеєр швидкого розгортання;

– підтримка кількох технологічних стеків;

– розділіть субдомени за їхніми характеристиками.

Прості служби кожна послуга складається з невеликої кількості субдоменів можливо, лише з одного, і тому їх легше зрозуміти та підтримувати.

Автоманомність команди, команда може розробляти, тестувати та розгортати свій сервіс незалежно від інших команд.

Конвеєр швидкого розгортання, кожна служба швидко тестується, оскільки вона відносно невелика та може бути розгорнута незалежно.

Підтримка кількох технологічних стеків, різні служби можуть використовувати різні технологічні стеки та оновлюватися незалежно.

Субдомени можна розділити за своїми характеристиками на окремі служби, щоб покращити масштабованість, доступність, безпеку.

Недоліки мікросервісів:

– деякі розподілені операції можуть бути складними, їх важко зрозуміти та усунути неполадки;

– деякі розподілені операції можуть бути потенційно неефективними;

– деякі операції може знадобитися реалізувати за допомогою складного, зрештою узгодженого (не ACID) керування транзакціями, оскільки слабкий зв’язок вимагає від кожної служби мати власну базу даних;

– деякі розподілені операції можуть передбачати тісний зв’язок часу виконання між службами, що знижує їх доступність;

– ризик тісного зв’язку часу проектування між службами, що вимагає тривалих покрокових змін [1].

Подивимось основні характеристики самостійного розгортання. Спрощене визначення «незалежного розгортання» означає, що послуга упакована як

розгортається або виконувана одиниця. Приклади розгорнутого або виконуваного блоку включають: виконуваний файл JAR, файл WAR, виконуваний файл операційної системи, образ контейнера Docker або файл Zip, який визначає функцію AWS Lambda. І навпаки, звичайний JAR-файл, який потрібно запакувати разом з іншими JAR-файлами, не є розгортаним або виконуваним модулем. Хоча це відповідає буквальному визначенню «самостійного розгортання», це вкрай неадекватно. Давайте подивимося на краще визначення.

Набагато більш змістовне визначення незалежного розгортання – це служба, яка упакована як розгортається або виконувана одиниця та готова до виробництва після того, як її було протестовано окремо. Така служба має власне сховище вихідного коду та конвеєр розгортання. Конвеєр розгортання тестує службу ізольовано, використовуючи дублі тестів для своїх співавторів разом із контрактним тестуванням, керованим споживачем. З конвеєра розгортання виходить служба, яку можна і потрібно розгортати у виробництві. Якщо вам потрібно протестувати свою службу з іншими службами, щоб переконатися, що вона готова до виробництва, тоді її не можна розгортати окремо. Крім того, ви можете розглянути можливість розміщення цих служб в єдиному сховищі. Це гарантує, що результат вашого єдиного конвеєра розгортання фактично готовий до виробництва. Це також усуне складність розробки в кількох сховищах. Важливою перевагою незалежного розгортання служби є те, що вона прискорює процес розгортання. Це усуває необхідність у повільних, крихких і складних наскрізних тестах кількох служб. Це також позбавляє команд від необхідності координувати роботу та потенційно перешкоджати одна одній. Однією з перешкод для самостійного розгортання служб є тести прийнятності користувача на системному рівні. Приймальні тести зазвичай пишуться з точки зору користувача та часто охоплюють кілька служб. Пряме впровадження таких тестів потребує спільного тестування кількох служб. Для незалежного розгортання служб необхідно замінити тести прийнятності користувача на системному рівні тестами прийнятності користувача на рівні сервісу [2].

Ще одна істотна характеристика полягає в тому, що послуги слабо пов’язані.

Фактично існує два різних типи зчеплення:

– зв'язок часу виконання;

– розрахунково-часовий зв'язок.

Зв'язок часу виконання знижує доступність, наприклад, уявімо, що createOrder()системна операція реалізована HTTP POST /orderskінцевою точкою в Order Service. Він Order Serviceобробляє, HTTP POSTвикликаючи інші служби, чекаючи їх відповіді, а потім надсилаючи відповідь своєму клієнту приклад на рисунку 1.1 [2].

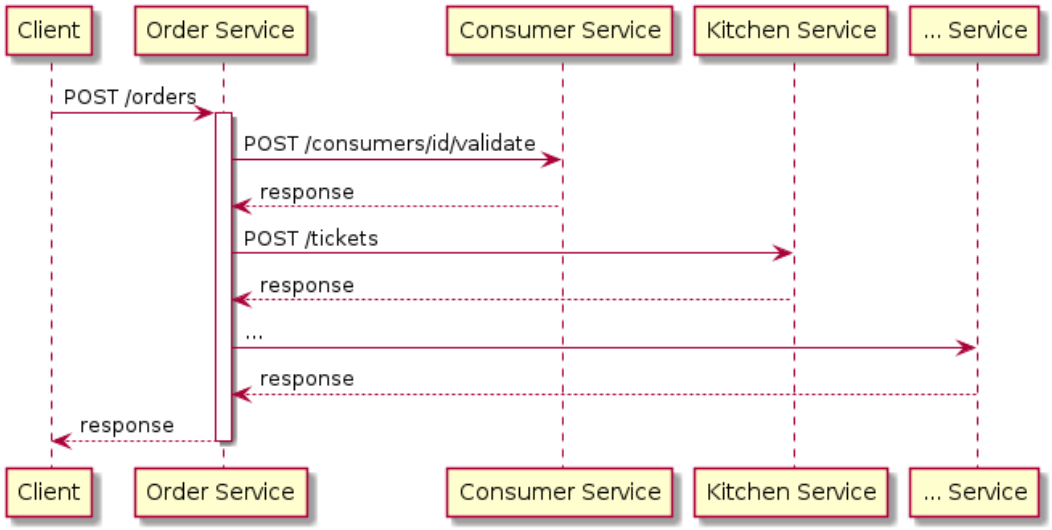


Рисунок 1.1 – слабо повязані [2]

У цьому дизайні Order Servicене можна відповісти на POSTзапит, доки на нього не дадуть відповіді інші служби.

Кажуть, що Order Service(або операція createOrder()) є середовищем виконання, пов’язаним з цими іншими службами. У результаті доступність операції createOrder() знижується, оскільки всі послуги мають бути доступними.

Зменшення часу роботи, зведення до мінімуму зв’язку під час роботи є однією із сил тяжіння темної матерії, яка протистоїть розкладанню. Один із способів зменшити зв’язок часу виконання операції – зменшити кількість служб, які її реалізують. Фактично, ми можемо повністю усунути зв’язок часу

виконання, зробивши операцію локальною для однієї служби. Однак не завжди можливо створити архітектуру мікросервісу, де всі операції виконуються локально. Це, ймовірно, порушує сили темної енергії , які сприяють розкладанню.

Інший спосіб зменшити зв’язок часу виконання, задовольнивши сили темної енергії – це розробити автономні служби. Автономна служба відповідає на синхронний запит частковим результатом, а потім асинхронно завершує операцію. Наприклад, Order Serviceможна відповісти на HTTP POST /orderszапит відповіддю 202 Accepted, а потім ініціювати Create Order Saga завершення операції. Цей підхід покращує доступність Order Service. Недоліком є те, що це робить клієнта більш складним, оскільки він повинен мати можливість обробляти часткові результати та якимось чином визначати кінцевий результат операції [3].

Взаємозв'язок між продовженням проекту та темпами розробки, ступінь зв'язку дека в дизайні між парами програмних елементів (клас-сервіс) – це ймовірність того, що її доведеться замінити з тієї ж причини. Зв'язок між службами в рамках розробки архітектури мікросервісів є особливо проблематичним.

З'єднання під час проектування знижує швидкість розробки, якщо дві служби підключені слабо, навряд потрібно буде змінити ще 1 службу, щоб замінити 1 службу. Однак, якщо 2 служби тісно пов'язані, зазвичай потрібно змінити іншу, щоб змінити 1 службу. Такі поступові зміни перекривають шлях, оскільки вони, як правило, передбачають внесення змін до API.

Кожного разу, коли потрібно внести будь-які суттєві зміни, кроки обслуговування клієнтів полягають тому, щоб додати нову основну версію API, змініть службу підтримки клієнтів. Служба повинна реалізовувати як старі, так і нові версії Api, поки всі клієнти не будуть переміщені. Видалення застарілої версії API зі служби підтримки клієнтів. Що ще гірше, служба часто належить різним командам, тому ці команди повинні координувати зміни. Іншими

словами, декомунізація часу розробки між службами підриває автономію команди.

Скорочений час проектування, мінімізація злиття під час проектування є однією з переваг темної матерії, яка протистоїть дисоціації. Існує кілька способів декомунізувати взаємозв'язок часу розробки між службами. Слабо пов'язані субдомени проекту слабо пов'язані субдомени можуть бути упаковані як різні служби. Слабка ланка під час проектування зазвичай досягається тим, що кожен субдомен має стабільний API, що охоплює його реалізацію.

Масові субдомени, тісно пов'язані з однією, і тією ж службою. Якщо два субдомени тісно декомунізовані, об'єднання їх в першу службу запобігає зв'язок між розроблюваними службами [3].

1.2 Опис атак на відмову

Атака на відмову в обслуговуванні, розподілена атака на відмову в обслуговуванні (англ. DoS attack, DDoS attack, (distributed) denial-of-service attack) – напад на комп'ютерну систему з наміром зробити комп'ютерні ресурси недоступними користувачам, для яких комп'ютерна система була призначена.

Одним із найпоширеніших методів нападу є насичення атакованого комп'ютера або мережевого устаткування великою кількістю зовнішніх запитів (часто безглуздих або неправильно сформульованих) таким чином атаковане устаткування не може відповісти користувачам, або відповідає настільки повільно, що стає фактично недоступним. Взагалі відмова сервісу здійснюється,

Перше, примусом атакованого устаткування до зупинки роботи програмного забезпечення/устаткування або до витрат наявних ресурсів, внаслідок чого устаткування не може продовжувати роботу.

Друге, заняттям комунікаційних каналів між користувачами і атакованим устаткуванням, внаслідок чого якість сполучення перестає відповідати вимогам.

Якщо атака відбувається одночасно з великої кількості IP-адрес, то її називають розподіленою (англ. distributed denial-of-service — DDoS).

Анатомія DoS-атак, DoS-атаки поділяються на локальні та віддалені. До локальних відносяться різні експлойти: форк-бомби і програми, що відкривають по мільйону файлів або запускають якийсь циклічний алгоритм, який «з'їдає» пам'ять та процесорні ресурси. Для локальної DoS атаки необхідно мати, або якимось чином отримати доступ до атакованої машини на рівні, що буде достатнім для захоплення ресурсів.

Flood (UDP-флуд, ICMP-флуд, MAC-флуд) – надсилання на адресу жертви величезної кількості безглуздих (рідше – осмислених) пакетів. Метою флуду може бути канал зв'язку або ресурси машини. У першому випадку потік пакетів займає весь пропускний канал і не дає машині, що атакується, можливості обробляти легальні запити. У другому – ресурси машини захоплюються за допомогою багаторазового і дуже частого звернення до якого-небудь сервісу, що виконує складну, ресурсоємну операцію. Це може бути, наприклад, тривале звернення до одного з активних компонентів (скрипту) веб-сервера. Сервер витрачає всі ресурси машини на обробку запитів, що атакують, а користувачам доводиться чекати.

У традиційному виконанні (один нападник – одна жертва) зараз залишається ефективним лише перший вид атак. Класичний флуд - марний. Просто тому що при сьогоднішній ширині каналу серверів, рівні обчислювальних потужностей і повсюдному використанні різних анти-DoS прийомів в ПЗ (наприклад, затримки при багаторазовому виконанні тих самих дій одним клієнтом), нападник перетворюється на докучливого комара, не здатного завдати будь-якого збитку. Але якщо цих «комарів» наберуться сотні, тисячі або навіть сотні тисяч, вони легко покладуть сервер на лопатки. Розподілена атака типу «відмова в обслуговуванні» (DDoS), зазвичай здійснювана за допомогою безлічі «зазомбованих» хостів, може відрізати від зовнішнього світу навіть найстійкіший сервер, і єдиним ефективним захистом при цьому є організація розподіленої системи серверів (кластера).

Є два варіанти організації DDoS атак:

– ботнет;

– флешмоб.

Ботнет – зараження певного числа комп'ютерів програмами, які в певний момент починають здійснювати запити до атакованого сервера.

Флешмоб – домовленість великого числа користувачів інтернету почати здійснювати певні типи запитів до атакованого сервера [3].

Найбільш популярні типи DDoS-атак, перше – це об'ємні атаки є одним із найстаріших типів DDoS-атак. Вони використовують великі обсяги трафіку, щоб заповнити пропускну здатність жертви або пропускну здатність між мережею та Інтернетом. Сьогодні найбільші об'ємні атаки вимірюються в терабітах на секунду (тбіт/с), що еквівалентно приблизно 9000 середніх підключень до Інтернету. Наприклад, під час флуд-атаки за протоколом UDP (User Datagram Protocol) зловмисники перевантажують цільовий віддалений сервер, посилаючи запити програмі, яка прослуховує певний порт. Через те, що сервер перевіряє і відповідає на кожен запит, його пропускна спроможність швидко закінчується і він стає недоступним. Друге – це атаки прикладного рівня (7 рівень моделі OSI) здійснюються на загальнодоступні програми за допомогою великого обсягу підробленого або фіктивного трафіку. Прикладом є HTTP-флуд, що заповнює певний веб-сервер легітимними запитами HTTP GET і HTTP POST. Незважаючи на те, що сервер може мати достатню пропускну здатність, він змушений обробляти велику кількість фіктивних запитів, тому його можливості обробки вичерпуються. Атаки прикладного рівня вимірюються десятками мільйонів запитів на секунду (RPS). Трете – під час атак на рівні протоколу зловмисники використовують уразливості мережевих протоколів, як-от TCP, UDP, ICMP (3 і 4 рівня OSI), щоб вичерпати ресурси системи жертви. Одним із прикладів є SYN-флуд, який надсилає велику кількість запитів на сервер жертви, але залишає відповіді на ці запити без подальших дій, тому так зване «тристороннє рукостискання» (Three-way Handshake) залишається неповним. Коли кількість незавершених з'єднань вичерпує потужність сервера, він стає недоступним.

Атаки на рівні протоколу використовують спеціально створені пакети для досягнення своїх шкідливих цілей і вимірюються в пакетах на секунду (PPS). Під час найбільших зафіксованих атак такого типу кількість пакетів сягала сотень мільйонів [4].

Техніки, що використовують в DDoS-атаках:

– підробка;

– відображення;

– посилення.

Підробка – це коли зловмисник підробляє IP-пакет, коли він змінює або маскує інформацію в його заголовку, яка повинна вказувати вам, звідки він надходить. Оскільки жертва не може бачити справжнє джерело пакета, вона не може блокувати атаки, що надходять із цього джерела.

Відображення – це коли зловмисник може створити підроблену IP-адресу, щоб виглядало так, ніби вона справді походить від передбачуваної жертви, а потім надіслати цей пакет сторонній системі, яка «відповідає» жертві. Через це цілі ще важче зрозуміти, звідки насправді походить атака.

Посилення – певні онлайн-сервіси можна обманом змусити відповісти на пакети дуже великими пакетами або кількома пакетами.

Розпізнання DDoS-атаки може бути важко діагностувати. Зрештою, атаки зовні нагадують потік трафіку від законних запитів від законних користувачів. Але є способи, за допомогою яких можна відрізнити штучний трафік від DDoS-атаки від більш «природного» трафіку, який ви очікуєте отримати від реальних користувачів.

Симптоми DDoS-атаки, на які слід звернути увагу

1. Незважаючи на методи спуфінгу чи розповсюдження, багато DDoS-атак відбуватимуться з обмеженого діапазону IP-адрес або з однієї країни чи регіону можливо, з якого ви зазвичай не бачите багато трафіку.

2. Подібним чином ви можете помітити, що весь трафік надходить від одного типу клієнта, з тією самою ОС і веб-браузером, які відображаються в його

HTTP-запитах, замість того, щоб показувати різноманіття, якого ви очікуєте від реальних відвідувачів.

3. Трафік може збиватися на один сервер, мережевий порт або веб-сторінку, а не рівномірно розподілятися на вашому сайті.

4. Трафік може надходити регулярними хвилями або шаблонами.

У чому різниця між атаками на відмову в обслуговуванні (DoS), і розподіленими атаками на відмову в обслуговуванні (DDoS)

Основна різниця полягає в кількості атакуючих машин. У разі DoS-атак використовується скрипт або інструмент, що запускається з одного пристрою і націлений на один конкретний сервер або робочу станцію. Тоді як DDoS-атаки здійснюються великою мережею скомпрометованих пристроїв, також відомою як ботнет, яку можна використовувати для перевантаження окремих пристроїв, програм, веб-сайтів, служб або навіть цілих мереж жертв.

Сім причин, чому варто подбати про захист від DDoS-атак:

1. Унаслідок атаки організація може втратити частину доходу через те, що її веб-сайт, сервіс або система не реагують на запити користувачів. Крім того, усунення наслідків інциденту також вимагає додаткових витрат.

2. За даними кількох відомих організацій, що займаються відстеженням DDoS-атак, за останні три роки кількість інцидентів стрімко зросла.

3. З кожним роком DDoS-атаки стають потужнішими. Зокрема, 2020 року найбільші атаки (на мережевому рівні) перевищували показник 1 Тбіт/с, 2021 року під час кількох наймасштабніших інцидентів було зафіксовано 2-3 Тбіт/с. Крім того, під час щонайменше двох DDoS-атак у 2021 році фіксували 15 мільйонів запитів на секунду (RPS).

4. Організаціям не завжди потрібно бути безпосередніми цілями DDoS-атак, щоб відчути їхній вплив. Зокрема, якщо зловмисники порушують роботу важливих елементів Інтернет-інфраструктури, наприклад місцевих або регіональних провайдерів. У 2016 році внаслідок атаки на провайдера DNS Dyn стали недоступні популярні онлайн-сервіси Twitter, Reddit, Netflix і Spotify.

5. Кіберзлочинці часто шантажують організації тим, що використовують ботнети для DDoS-атаки на них, якщо жертви не заплатять викуп. Для цього зловмисники не обов'язково можуть мати доступ до мереж цілей.

6. Починаючи з 2020 року також було зафіксовано випадки, коли відомі групи кіберзлочинців, які поширюють програми-вимагачі, використовували DDoS-атаки для додаткового шантажу жертв.

7. У даркнеті існують послуги з найму зловмисників, які займаються DDoS-атаками [4].

1.3 Існуючі методи захисту від DDoS атак

Щоб знизити руйнівні ризики DDoS-атак, організаціям необхідно використовувати комплексні заходи, включно з базовим аналізом і моніторингом мережевого трафіку, плануванням DDoS-атак, заходами щодо пом'якшення наслідків DDoS-атак, а також розгортанням інструментів захисту від DDoS-атак і розвідки загроз. Наведені нижче практики можуть стати основою ефективної стратегії запобігання DDoS-атакам.

Знати, на що звертати увагу, і стежити за цим, щоб виявити DDoS-атаку до того, як стане занадто пізно, потрібно знати, як виглядає звичайний мережевий трафік. Створивши базовий шаблон вашого звичайного трафіку, ви зможете легше виявити ознаки DDoS-атаки, такі як незрозуміло низька продуктивність мережі, нестабільне з'єднання, періодичні збої в роботі мережі, незвичні джерела трафіку або сплеск спаму. Пильний моніторинг має вирішальне значення, як мережевий трафік, так і трафік додатків; навіть невелика аномалія може сигналізувати про спроби кіберзлочинців перед більш масштабною атакою. Чим раніше ви виявите подію, тим швидше та ефективніше ви зможете активувати плани протидії DDoS-атакам. Водночас дуже важливо звести до мінімуму помилкові спрацьовування, щоб уникнути непотрібних операційних перебоїв.

Ключеві аспекти – це створення базового шаблону трафіу, ознаки DDoS-атаки, пильний моніторинг, швидка реакція, Мінімізація помилкових спрацьовувань. Основний акцент робиться на важливості створення базового шаблону нормального трафіку і пильного стеження за будь-якими відхиленнями від нього.

Рекомендую цей спосіб бо для ефективного виявлення DDoS-атак та захисту мережі від них. Пильне спостереження за мережевим трафіком та додатковим трафіком є важливим елементом безпеки в інформаційній сфері. Виявлення аномалій, які можуть вказувати на потенційні атаки, дозволяє забезпечити швидке реагування та мінімізувати можливі наслідки для системи. Створення базового шаблону звичайного трафіку допомагає зрозуміти, як повинен виглядати нормальний режим роботи мережі. Це дозволяє вчасно виявляти відхилення від стандарту, які можуть бути ознаками DDoS-атаки. Моніторинг мережі та додаткового трафіку на предмет будь-яких аномалій допомагає попередити можливі загрози та вчасно реагувати на них. Швидке виявлення DDoS-атаки та активування планів протидії є ключовими уникнення серйозних наслідків для мережі. Реагування на атаку в найкоротший термін допомагає зменшити можливі збитки та зберегти безпеку системи. Мінімізація помилкових спрацьовувань також важлива, оскільки дозволяє уникнути непотрібних перебоїв та зберегти ефективність заходів безпеки.

Скласти план реагування на атаки на відмову в обслуговуванні, коли ви визначили, що відбувається ймовірна DDoS-атака, ваша організація повинна мати можливість швидко та ефективно реагувати на неї. Детальне планування дозволить уникнути необхідності імпровізувати під тиском обставин. При розробці плану реагування на DDoS-атаку важливо враховувати кілька важливих аспектів.

По-перше, вам потрібно проаналізувати потенційні загрози та визначити сценарії атак, які можуть допомогти вам підготуватися до різних ситуацій. Поглиблене вивчення системи та мережі може допомогти виявити слабкі місця, які зловмисники можуть використовувати для здійснення DDoS-атак.

Другим важливим кроком є розробка конкретних кроків та визначення відповідальних за кожен етап плану реагування. Забезпечення чіткого розподілу

обов'язків сприяло б швидкій координації дій під час кризових явищ. Крім того, важливо підготувати інструменти для моніторингу мережі та виявлення аномалій у режимі реального часу. Швидко виявляючи DDoS-атаки, ви можете швидко реагувати і ініціювати заходи захисту.Також не менш важливим є регулярне навчання персоналу плануванню та реалізації планів реагування. Тільки навчений персонал може ефективно діяти в кризових ситуаціях і мінімізувати можливі наслідки DDoS-атак.

Таким чином, створення детального та добре продуманого плану реагування на атаки DDoS є важливою частиною вашої загальної стратегії кібербезпеки. Тільки завдяки попередньому плануванню та підготовці Ви зможете ефективно захистити свою мережу від потенційних загроз.

Ваш план повинен включати в собі.

1. контрольний список систем, активів та сучасних засобів виявлення загроз;
2. визначена команда реагування з компетенціями протидії DDoS-атакам;
3. процедури підтримки бізнес-операцій на час проведення атаки;
4. протоколи сповіщення про інциденти та їх ескалації;
5. комунікаційний план, що охоплює як співробітників, так і зовнішні зацікавлені сторони, такі як клієнти, партнери та засоби масової інформації.

Ідея створення плану у якого основна мета полягає в тому, щоб забезпечити швидку та ефективну реацію на подібні інциденти шляхом детального планування.

Впровадження рішень для захисту від DDoS-атак та аналізу загроз. Запобігання DDoS-атакам базується на багаторівневій стратегії, що складається з передових технологій, інструментів та аналізу загроз. Рішення для захисту від DDoS-атак має включати можливості моніторингу трафіку, виявлення загроз у режимі реального часу, блокування ненормальної поведінки, розпізнавання шаблонів атак нульового дня, очищення від DDoS-атак та автоматичного реагування. Аналітика загроз необхідна для поліпшення інструментів запобігання DDoS-атак за допомогою своєчасних даних про поточну активність і тенденції в області DDoS-атак, включаючи IP-адреси ddos-ботнетів і вразливих серверів, які, як відомо, пов'язані з DDoS-атаками. У поєднанні з можливостями

виявлення загроз у режимі реального часу, штучного інтелекту (AI)/машинного навчання (ML) та автоматичного видалення підписів аналіз загроз дозволяє організаціям застосовувати проактивний підхід до протидії DDoS-атакам.

При впровадженні рішення для захисту від DDoS-атак і аналізу загроз важливо враховувати кілька важливих аспектів. Першим кроком є аналіз потенційних ризиків, які можуть виникнути внаслідок таких атак. Для цього вам необхідно детально вивчити види DDoS-атак, їх характеристики і можливі наслідки для вашої системи безпеки. Далі варто розглянути можливі варіанти захисту від DDoS-атак.

Одним з ефективних способів є використання програмного і апаратного забезпечення, призначеного для виявлення і блокування атак. Також варто розглянути можливість використання хмарних рішень для забезпечення високого рівня захисту від DDoS-атак. Після введення захисних заходів важливо проводити систематичний аналіз загроз для своєчасного виявлення нових методів атаки. Це допомагає підтримувати високий рівень безпеки системи та уникати серйозних наслідків потенційних DDoS-атак.

Також важливо враховувати важливість постійного навчання персоналу з кібербезпеки та його здатності реагувати на можливі загрози. Тільки поєднання технічних заходів і своєчасних аналітичних дій може забезпечити ефективний захист від DDoS-атак та інших кіберзагроз.

Забезпечення відмовостійкість інфраструктури. Враховуючи, що в якийсь момент можуть бути зроблені спроби DDoS-атак, слід вжити заходів для мінімізації наслідків. Проектування мереж і систем з трафіком, що в 2-5 разів перевищує очікувані базові потреби, може допомогти нейтралізувати атаки при достатньому часі відгуку. Розподіл ресурсів може обмежити масштаби атак (наприклад, розміщення серверів в окремих центрах обробки даних або в різних мережах або в різних місцях). (Або розмістити центр обробки даних.) Пристрої резервного копіювання та архітектури високої доступності (ha-архітектури) можуть прискорити відновлення системи після DDoS-атак (зверніть увагу, що їх слід запускати тільки після завершення атаки, щоб не піддаватися безперервним

атакам). Уникайте або усувайте вузькі місця та окремі точки збою, які можуть зробити вас особливо вразливими до припливу трафіку.

Забезпечення відмовостійкості інфраструктури є критично важливим аспектом будь-якої сучасної організації. Підтримка безперебійної роботи систем та мереж має вирішальне значення для забезпечення ефективності бізнесу та задоволення потреб клієнтів. Планування та реалізація відмовостійких рішень можуть значно зменшити ризики від відмов у роботі технологічних систем. У процесі забезпечення відмовостійкості інфраструктури слід удосконалювати архітектуру систем, використовувати дублювання обладнання та мережеві резервування, а також запроваджувати автоматичне відновлення послуг у разі відмов. Важливо також регулярно проводити аудит інфраструктури для виявлення слабких місць та запобігання можливим проблемам у майбутньому. Підтримка відмовостійкості інфраструктури вимагає не лише технічних знань, але й глибокого розуміння потреб бізнесу та користувачів. Важливо враховувати специфіку діяльності організації, її потенційні ризики та можливі наслідки в разі відмови систем. Комплексний підхід до забезпечення відмовостійкості дозволяє підвищити надійність інфраструктури та забезпечити стабільну роботу бізнесу в умовах зростаючих вимог до технологічних систем.

Усвідомлення важливості відмовостійкості та прийняття відповідних заходів є ключовими для успішної діяльності будь-якої компанії. Інвестування в забезпечення високої доступності систем дозволяє уникнути серйозних проблем у майбутньому та забезпечити стабільну роботу бізнесу навіть у найскладніших умовах.

Знайти укриття у хмарі. Хмара має кілька можливостей зменшити ризик DDoS-атак. Хмарні провайдери мають набагато більшу пропускну здатність, ніж традиційні компанії, а розсіювання хмари може сприяти підвищенню відмовостійкості. Аналогічно, безпечне резервне копіювання даних у хмарі може сприяти швидкому відновленню у разі пошкодження системи.

З іншого боку, багатокористувацьке хмарне середовище має певні ризики. Постачальник хмарних послуг, хостинг-провайдер або колокейшн-провайдер,

який виявив DDoS-атаку на одного клієнта, запобігає побічні ефекти для інших клієнтів аналогічним чином, атака на іншого клієнта хмарного провайдера може вплинути на вашу компанію, навіть якщо ви не є основною метою. Це буде. У зв'язку з цим важливо працювати з хмарними, хостинговими і колокейшн-провайдерами, які надають захист від DDoS-атак в якості послуги для своїх клієнтів.

Знайти укриття в хмарі може бути важливою задачею в ситуаціях надзвичайних ситуацій або погіршення погодних умов. Ось кілька порад, які можуть вам допомогти:

Державна служба з надзвичайних ситуацій (ДСНС) може надати інформацію про найближчі укриття в хмарі. Ви можете звернутися до них або перевірити їх веб-сайт для отримання актуальної інформації. Місцеві органи влади, такі як міська рада або сільська рада, також можуть мати інформацію про укриття в хмарі у вашому регіоні. Зверніться до них або перевірте їхній веб-сайт для отримання деталей. Інтернет-карти можуть бути корисним інструментом для пошуку укриття в хмарі. Ви можете використовувати пошукові системи або спеціалізовані веб-сайти, щоб знайти місця, де розташовані укриття в хмарі у вашому районі. Зверніться до місцевих експертів з питань безпеки або надзвичайних ситуацій. Вони можуть мати інформацію про укриття в хмарі, яка може бути недоступна в інших джерелах. Не забувайте про власну безпеку. Перед тим, як шукати укриття в хмарі, оцініть поточну ситуацію і ризики. Дотримуйтесь рекомендацій та інструкцій від органів влади та експертів з питань безпеки. Важливо мати на увазі, що рекомендації та доступні укриття в хмарі можуть змінюватися залежно від конкретної ситуації. Тому рекомендується періодично оновлювати інформацію та слідкувати за новинами та оголошеннями від органів влади.

1.4 Принципи захисту від DDoS атак

Пом’якшити DDoS-атаку складно, оскільки, як зазначалося раніше, атака має форму веб-трафіку того самого типу, який використовують ваші законні клієнти. Було б легко «зупинити» DDoS-атаку на ваш веб-сайт, просто заблокувавши всі HTTP-запити, і справді це може бути необхідним, щоб захистити ваш сервер від збою. Але це також блокує будь-кого іншого з відвідування вашого веб-сайту, що означає, що ваші зловмисники досягли своїх цілей.

Якщо ви можете відрізнити трафік DDoS від легального трафіку, як описано в попередньому розділі, це може допомогти пом’якшити атаку, зберігаючи ваші служби принаймні частково онлайн: наприклад, якщо ви знаєте, що трафік атаки надходить зі східноєвропейських джерел, ви можете заблокувати IP-адреси з цього географічного регіону. Хорошим профілактичним методом є закриття будь-яких загальнодоступних служб, якими ви не користуєтеся. Служби, які можуть бути вразливими до атак на прикладному рівні, можна вимкнути, не впливаючи на вашу здатність обслуговувати веб-сторінки. Загалом, однак, найкращий спосіб пом’якшити DDoS-атаки – просто мати здатність протистояти великій кількості вхідного трафіку. Залежно від вашої ситуації це може означати посилення власної мережі або використання мережі доставки вмісту (CDN), послуги, призначеної для обслуговування величезних обсягів трафіку. Ваш постачальник мережевих послуг може мати власні служби пом’якшення, якими ви можете скористатися [7].

Візьмемо вісім найпоширеніших способів зупини DDoS-Атаки.

Перший спосіб – Підвищення кібервідмовостійкість за допомогою передового рішення ZTNA. Згідно з BDIR Verizon за 2022 рік , DDoS був найпоширенішою формою атаки. Якщо доступ до мережі з нульовою довірою (ZTNA) прийнято, це може бути ефективним пом’якшенням цих катаклізмичних атак. Рідне хмарне рішення ZTNA, яке включає потужні можливості захисту кінцевих точок, як-от CylanceGATEWAY™ , може забезпечити захист, виявлення та запобігання DDoS-атак.

Захист мережі належне рішення ZTNA для пом’якшення DDoS-атак захищає мережу, оскільки не потребує відкриття будь-яких портів, оскільки воно передає трафік до корпоративної мережі, тож організації фундаментально захищені від DDoS.

Виявлення загроз рішення ZTNA використовує системи виявлення вторгнень для виявлення зловмисного трафіку на основі моделей мережевих потоків на трьох незалежних рівнях: система доменних імен (DNS), протокол контрольних повідомлень Інтернету (ICMP) і безпека транспортного рівня (TLS). Крім того, мережевий трафік постійно оцінюється, а фактори ризику обчислюються за кількома векторами. Удосконалені рішення поєднують машинне навчання, репутацію IP-адреси та підрахунок ризиків, щоб створити динамічний чорний список інтернет-напрямків, які будуть і активно блокуються.

Запобігання спроби зловмисного вторгнення, такі як впровадження SQL, підробка протоколу розпізнавання адрес (ARP), Man-In-The-Middle (MiTM) і зловмисні точки доступу Wi-Fi, свідчать про DDoS-атаки. На додаток до багаторівневого тунелю з підтримкою ідентифікаційних даних із безперервною автентифікацією та авторизацією, правильне рішення ZTNA для DDoS також сприяє реалізації сегментованого контролю доступу до мережі, що разом запобігає ARP-спуфінгу. Спуфінг ARP є звичайним переходом до MiTM, тому його також запобігають. Нарешті, зв’язок рівня 3 має бути повністю зашифрованим, що зменшує ймовірність успішної тунельної спроби зловмисного вторгнення, наприклад SQL-ін’єкції, зловмисних точок доступу Wi-Fi тощо.

Другий спосіб – Blackhole Routing.

Хоча іноді це вважається надлишковим, якщо ви використовуєте розширене рішення ZTNA, залежно від бюджетних обмежень альтернативою для розгляду є маршрутизація через чорну діру. За допомогою такої тактики мережевий трафік направляється в «чорну діру» і втрачається. Недоліком цього методу є те, що без належних критеріїв обмеження як легітимний, так і

нелегітимний трафік видаляється з мережі. Це фактично робить DDoS-атаку успішною, оскільки мережа тепер недоступна.

Щоб блокувати DDoS-атаки "чорних дір", адміністратори налаштовують маршрутизатори на перенаправлення трафіку на нульовий інтерфейс і ефективне відкидання трафіку. Зазвичай це відбувається тому, що мережа не має достатньої пропускної здатності інтернет-портів і не може впоратися з атакою без шкоди для пропускної здатності виробничих служб.

Інтернет-провайдери та мережеві адміністратори можуть використовувати кілька методів запобігання чорних дірок у маршрутизації, щоб блокувати шкідливий DDoS-трафік від досягнення наміченої мети. Ці методи IP-маршрутизації можуть перенаправляти трафік на основі IP-адрес джерела або призначення.

1. Статична чорна маршрутизація передбачає ручне налаштування маршрутизаторів для відсікання трафіку, що надходить із певної IP-адреси або призначений для неї. Ця техніка статичного маршруту ефективно блокує трафік з IP-адрес, які, як відомо, є шкідливими, від цільових серверів.

2. Чорна маршрутизація BGP — це технологія фільтрації пакетів на основі призначення, яка використовує протокол BGP (Border Gateway Protocol) для «оголошення» або передачі чорного маршруту для певної IP-адреси іншим маршрутизаторам у мережі BGP.

3. Віддалена фільтрація чорної діри або фільтрація RTBH зазвичай блокує трафік на основі IP-адрес призначення. Фільтрація RTBH також використовує BGP, але забезпечує більш контрольований і цілеспрямований підхід, що дозволяє мережевим адміністраторам пом’якшувати атаки на певні хости або підмережі, не впливаючи на всю мережу.

4. Чорна маршрутизація Flowspec BGP забезпечує ще більш детальний підхід до фільтрації трафіку. За допомогою цієї методики адміністратори можуть вказати додаткові параметри, які допомагають точніше націлюватися на зловмисний трафік, одночасно дозволяючи законному мережевому трафіку досягти місця призначення.

5. Списки блокування для фільтрації спаму також можна використовувати для чорного холінгу. Чорні списки містять IP-адреси, які, як відомо, надсилають спам або шкідливий трафік. Коли трафік надходить на сервер електронної пошти з IP-адреси зі списку блокування, він автоматично відхиляється або поміщається в карантин для перевірки.

6. IP-фільтрація на основі адресатів зазвичай використовується для відкидання небажаного трафіку з мережі. Цього можна досягти за допомогою одного хост-маршрутизатора або «всіх» маршрутизаторів у мережі. RTBH також можна застосувати до напрямку, у якому пакет проходить мережею (вхідний прийом або вихідна передача). Приклад BlackHole рисунок 1.2.

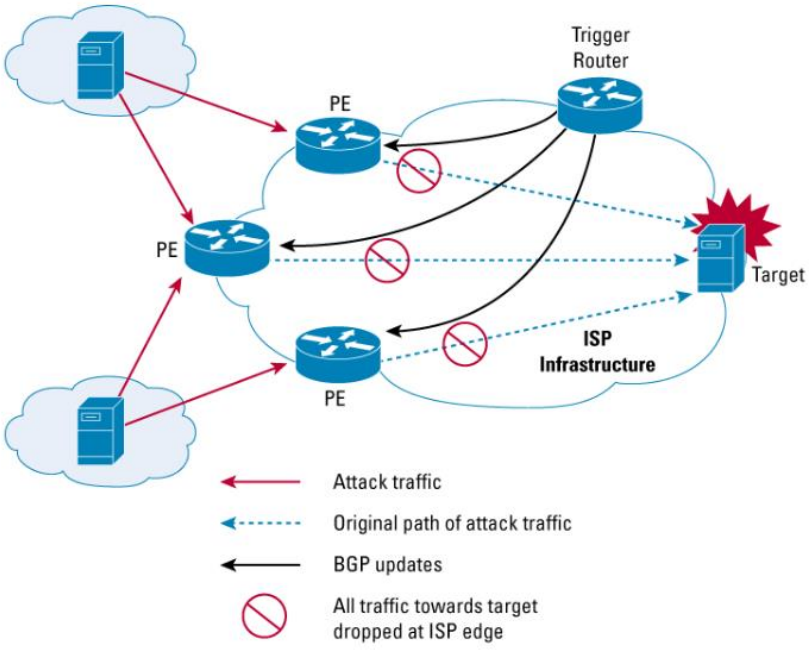


Рисунок 1.2 – приклад Blackhole Routing

Третій спосіб – розвідка соціальних мереж.

Слідкуйте за соціальними мережами, зокрема Twitter, на наявність погроз, розмов і хвастощів, які можуть означати, що ви стали мішенню. Ось безкоштовний ресурс, який може бути вам корисним:Інструменти та бібліотеки v2, створені Twitter.

Четвертий спосіб – обмеження швидкості.

Обмежте кількість запитів, які сервер прийматиме протягом певного періоду часу. Як правило, цього недостатньо для захисту від більш складних атак, але є хорошим компонентом для багатосторонньої стратегії пом'якшення.

П'ятий спосіб – брандмауер веб-додатків (WAF)

Брандмауер веб-додатків (WAF) перевіряє семантику XML/SOAP потокового трафіку і виявляє різні атаки на рівні додатків, перевіряючи HTTP/HTTPS-трафік для виявлення веб-порталів і пристроїв безпеки (апаратних або віртуальних), основним завданням яких є захист веб-додатків. Брандмауер веб-додатків діє як проксі-сервер, але завдяки своїй здатності аналізувати HTTPS-трафік (імпортуючи сертифікати безпеки цільового сервера), термінації SSL-трафіку і балансуванню навантаження на сервер, можуть виконуватися й інші функції. Крім того, WAF підтримує кластеризацію для прискорення роботи веб-додатків.

Міжмережевий екран для веб-додатків (WAF) функціонує на основі двох загальноприйнятих моделей безпеки:

– negative;

– positive.

Negative – негативна модель або чорний список (заперечує те, що є свідомо встановленим). Для надання базового захисту, аналогічного IPS, але з більш високим рівнем оцінювання безпеки застосунків, WAF може використовувати як загальновідомі сигнатури для запобігання найпоширенішим атакам, так і специфічні сигнатури для атак, які використовують уразливості окремих web-додатків. Наприклад: заперечувати певний потенційно небезпечний HTTP запит GET і дозволити все інше.

Positive – позитивна модель або білий список (дозволяє тільки те, що є свідомо встановленим). Для поліпшеного захисту, на додаток до сигнатур, використовується ще один тип логіки: правила, які визначають, що явно дозволено. Наприклад: дозволити тільки HTTP GET запити для певного URL і заборонити все інше.

Ключові можливості WAF:

1. Підтримка всіх застосовних до web-додатків PCI DSS Requirements, пов'язаних із компонентами системи в середовищі обробки даних за платіжними картками.

2. Оперативна реакція (визначається активною політикою та/або набором правил) на загрози й атаки, визначені, як мінімум, в OWASP Top 10.

3. Перевірка вхідного HTTP/HTTPS трафіку і запитів до web-додатків і вжиття захисних заходів на основі активних політик і правил (дозволити, блокувати, попередити).

4. Підтримка і дотримання коректного функціонування позитивної та негативної моделі безпеки.

5. Вивчення і перевірка web-контенту, створеного за допомогою Hypertext Markup Language (HTML), Dynamic HTML (DHTML), Cascading Style Sheets (CSS) і основних протоколів доставки web-контенту, таких як Hypertext Transport Protocol (HTTP) і Transport Protocol Hypertext over SSL (HTTPS).

6. Запобігання витоку даних - перевірка вихідного HTTP/HTTPS-трафіку і запитів до web-додатків і вжиття захисних заходів на основі активних політик і правил, а також своєчасний запис подій, що відбулися, в журнал подій.

7. Аналіз повідомлень web-сервісів, особливо публічних. Як правило, включає перевірку Simple Object Access Protocol (SOAP) і eXtensible Markup Language (XML), а також Remote Procedure Call (RPC) орієнтовані моделі взаємодії з web-сервісами, засновані на базі HTTP.

8. Перевірка будь-якого протоколу або конструкції даних (пропрієтарних або стандартизованих), які використовуються для передавання даних у/з web-додатку.

9. Захист від загроз, спрямованих безпосередньо на WAF;

10. Термінація SSL і/або TLS (розшифровка і перевірка трафіку перед відправленням до web-додатку).

Шостий спосіб – тестування на проникнення.

Розгляньте можливість використання сторонньої служби проникнення або тестування пера для моделювання атаки на вашу IT-інфраструктуру за

допомогою реальних сценаріїв, щоб бути готовими до реальних подій. Регулярне відпрацювання плану реагування на DDoS вашої організації з усіма внутрішніми та зовнішніми зацікавленими сторонами допоможе виявити прогалини та проблеми, переконатися, що всі учасники розуміють свої ролі та обов'язки під час DDoS-атаки, і зміцнити довіру до плану реагування на DDoS рисунок 1.3.

Рисунок 1.3 – план реагування на DDoS.

Сьомий спосіб – метод мережевої дифузії Anycast.

Anycast — це метод мережевої маршрутизації, який розподіляє вхідні запити між різними серверами. Ідея полягає в тому, що у разі DDoS-атаки доданий трафік розподіляється та поглинається мережею. Ефективність цього підходу залежить від розміру DDoS-атаки та розміру та компетенції мережі.

Мережева маршрутизація Anycast дозволяє маршрутизувати вхідні запити на з'єднання між кількома центрами обробки даних. Коли запити надходять на одну IP-адресу , пов'язану з мережею Anycast, мережа розподіляє дані на основі певної методології визначення пріоритетів. Процес відбору, що стоїть за вибором конкретного центру обробки даних, зазвичай оптимізується для зменшення затримки шляхом вибору центру обробки даних із найменшою відстанню від запитувача. Anycast характеризується асоціацією 1-до-1 із багатьох і є одним із 5 основних методів мережевого протоколу, які використовуються в Інтернет-протоколі.

Цей спосіб використовують Якщо багато запитів надсилаються одночасно до одного вихідного сервера, сервер може бути перевантажений трафіком, і не зможе ефективно відповідати на додаткові вхідні запити. За допомогою мережі

Anycast замість того, щоб один вихідний сервер брав на себе основний трафік, навантаження також можна розподілити між іншими доступними центрами обробки даних, кожен з яких матиме сервери, здатні обробляти вхідні запити та відповідати на них. Цей метод маршрутизації може запобігти збільшенню пропускної здатності вихідного сервера та уникнути перерв у обслуговуванні клієнтів, які запитують вміст із вихідного сервера.

Мережа Anycast помякшує атуку на відмову інструменти пом'якшення DDoS відфільтрують частину трафіку атак, Anycast розподіляє трафік, що залишився, між кількома центрами обробки даних, запобігаючи перевантаженню будь-якого місця запитами. Якщо пропускна здатність мережі Anycast перевищує трафік атаки, атаку ефективно пом'якшують. У більшості DDoS-атак багато скомпрометованих комп'ютерів-«зомбі» або «ботів» використовуються для створення так званого ботнету . Ці машини можуть бути розкидані по мережі та генерувати стільки трафіку, що можуть перевантажити типову машину, підключену до Unicast.

Належним чином Anycasted CDN збільшує площу поверхні приймаючої мережі, щоб нефільтрований трафік відмови в обслуговуванні з розподіленої бот-мережі поглинався кожним із центрів обробки даних CDN. Як наслідок, оскільки мережа продовжує зростати в розмірах і місткості, стає все важче і важче запускати ефективний DDoS проти будь-кого, хто використовує CDN.

Справжню мережу Anycasted налаштувати непросто. Належне впровадження вимагає, щоб постачальник CDN підтримував власне мережеве обладнання, встановлював прямі зв'язки зі своїми операторами зв'язку та налаштовував свої мережеві маршрути, щоб гарантувати, що трафік не «перекидається» між кількома місцями. У цій публікації в блозі Cloudflare пояснюється, як Cloudflare використовує Anycast для балансування навантаження без балансувальників навантаження. Приклад цього методу на рисунку 1.4.

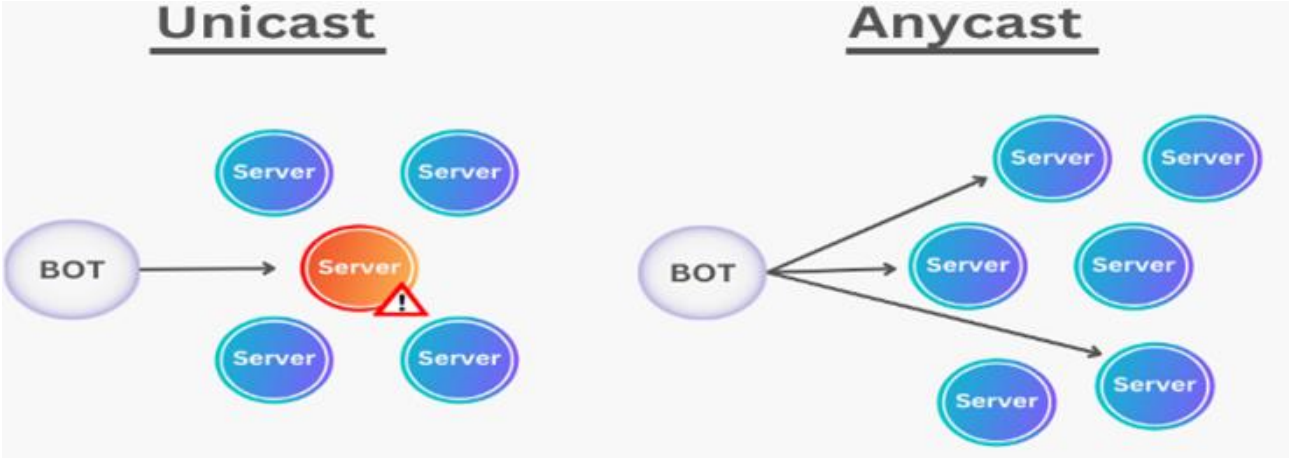


Рисунок 1.4 – метод Anycast.

Восьмий спосіб – підпишіться на службу захисту від DdoS.

Спільне керівництво CISA, Федерального бюро розслідувань (ФБР) і Міждержавного центру обміну та аналізу інформації (MS-ISAC) рекомендує організаціям зареєструватися в спеціальній службі захисту від DDoS. Хоча багато постачальників послуг Інтернету (ISP) мають засоби захисту від DDoS, їх може бути недостатньо, щоб протистояти широкомасштабним або розширеним атакам DDoS. Служба захисту від DDoS, наприклад AWS Shield , може контролювати трафік, підтверджувати атаку, визначати джерело та пом’якшувати ситуацію, перенаправляючи зловмисний трафік із вашої мережі. CylanceGATEWAY містить AWS Shield як додатковий рівень захисту.

Організаціям також рекомендується поговорити з постачальником керованих послуг (MSP) про конкретні керовані служби, які захищають від атак DDoS. MSP, що пропонують різні технології на «краї», можуть допомогти з налаштуванням граничного захисту.

Сервіси Edge Defense можуть скоротити час простою, спричинений DDoS-атаками. Служби периферійного захисту, виявлення та пом’якшення зменшують ризик того, що зловмисний трафік досягне своєї мети, і значно підвищують шанси законних користувачів досягти ваших веб-сайтів/веб-програм.[6]

1.5 Висновки розділу

У цьому розділі я докладно розглянув теоретичні аспекти захисту мікросервісних застосунків. На початку я визначив основні характеристики мікросервісної архітектури, яка включає розділення системи на окремі, незалежні сервіси. Це спрощує масштабування, розвиток і підтримку системи, що є важливим для сучасних технологічних рішень. Також я проаналізував загрози, з якими стикаються мікросервіси, зокрема атаки на відмову (DDoS) і їхні наслідки. Я розглянув різні типи атак, такі як SQL-ін'єкції, XSS, CSRF та різні методи DDoS-атак. Це дозволило мені краще зрозуміти, які загрози можуть виникнути і які слабкі місця можуть бути виявлені в системах. Я звернув увагу на методи захисту від цих загроз. Зокрема, я проаналізував впровадження рішень для захисту від DDoS-атак та аналізу загроз. Цей метод дозволяє проактивно виявляти та блокувати загрози до того, як вони зможуть вплинути на інфраструктуру. Це забезпечує оперативний моніторинг трафіку в реальному часі та постійне оновлення алгоритмів, що є критично важливим для захисту від нових типів загроз.

Я також розглянув забезпечення відмовостійкості інфраструктури з використанням брандмауера веб-додатків (WAF). Брандмауери веб-додатків є ключовим елементом захисту, що дозволяє ефективно фільтрувати HTTP/HTTPS-трафік, виявляючи і блокуючи загрози на рівні запитів до веб-додатків. Такий підхід значно знижує ризик злому і забезпечує безперебійну роботу системи, що є критично важливим для підтримки високого рівня доступності та надійності. Використання цих методів у створенні системи захисту мікросервісних застосунків дозволить мені забезпечити комплексний підхід до безпеки, який включає проактивне виявлення загроз, належну фільтрацію трафіку і забезпечення високої доступності систем. Такий підхід не тільки мінімізує ризики атаки, але й підтримує безперебійну роботу системи, що є основою для надійної і ефективної інфраструктури мікросервісів.

Таким чином, розробка і впровадження цих методів стане важливою складовою частиною моєї стратегії забезпечення кібербезпеки, що дозволить

підтримувати високу безпеку і стабільність мікросервісних застосунків, відповідаючи сучасним вимогам до захисту інформації.

2 РОЗРОБКА ТА ВПРОВАДЖЕННЯ СИСТЕМИ ЗАХИСТУ МІКРОСЕРВІСНИХ ЗАСТОСУНКІВ ВІД DDOS АТАК

2.1 Порівняння ефективності різних підходів

Метою цього розділу є виявлення найбільш оптимальних методів захисту від DDoS-атак, що дозволяють забезпечити високу надійність та ефективність захисту інформаційної інфраструктури. Аналіз кожного з методів включає розгляд їхніх переваг та недоліків, а також оцінку їхньої ефективності в умовах різних типів атак. Цей аналіз дозволить визначити найкращі практики та рекомендації для впровадження надійних систем захисту, що є критично важливим для забезпечення стабільної роботи онлайн-сервісів та безпеки даних. Порівняння ефективності різних підходів до захисту від DDoS-атак допоможе зрозуміти, які методи є найбільш дієвими у певних умовах, і на основі цього зробити обґрунтовані висновки для подальшої розробки захисних заходів.

Перший підхід для захисту Anycast. Він дозволяє використовувати одну IP-адресу на декількох серверах в різних місцях. Коли запит надходить до мережі, він автоматично перенаправляється на найближчий сервер, що дозволяє ефективно розподілити навантаження та мінімізувати затримки. Переваги цього методу:

- розподіл навантаження;
- зменшення затримок;
- підвищена стійкість.

Розподіл навантаження є однією з основних технологій у сфері управління інтернет-ресурсами, що дозволяє забезпечити високу доступність, надійність і продуктивність веб-додатків. Його основна функція полягає у рівномірному розподілі вхідного трафіку між кількома серверами, що дозволяє уникнути перевантаження окремих машин і знижує ризик їх виходу з ладу.

Також він оптимізовує продуктивність. Розподіл навантаження також сприяє оптимізації продуктивності системи. Він дозволяє використовувати

ресурси декількох серверів, що забезпечує ефективне використання обчислювальної потужності, пам'яті та інших ресурсів. Це дозволяє системам справлятися з великим обсягом трафіку, не втрачаючи в швидкодії та якості обробки запитів. Крім того, сучасні системи розподілу навантаження можуть автоматично масштабуватися, додаючи нові сервери в мережу у відповідь на зростання навантаження, що забезпечує гнучкість і масштабованість системи.

Невід'ємна частина є захист від DDoS-атак. Однією з важливих функцій розподілу навантаження є його здатність захищати веб-додатки від атак, таких як DDoS (Distributed Denial of Service). Розподіляючи трафік між кількома серверами, цей метод ускладнює для зловмисників зосередження всієї своєї потужності на одному вузлі, що робить атаки менш ефективними. Додатково, багато систем розподілу навантаження мають вбудовані механізми захисту, такі як фільтрація трафіку, блокування шкідливих IP-адрес і автоматичне виявлення аномалій, що підвищує загальний рівень безпеки системи.

Хоча метод Anycast має безліч переваг, таких як підвищення надійності, оптимізація продуктивності та захист від атак, важливо враховувати і її недоліки:

– висока вартість;

– складність налаштування.

Одним з основних недоліків системи розподілу навантаження є її висока вартість. Це обумовлено декількома факторами, що включають як початкові витрати на апаратне забезпечення, так і постійні витрати на програмне забезпечення та інфраструктуру.

По-перше, апаратне забезпечення для балансування навантаження часто коштує дорого. Балансувальники навантаження високої продуктивності, які можуть забезпечити необхідну надійність і швидкість обробки трафіку, зазвичай мають високу ціну. Це може включати спеціалізоване обладнання з високою продуктивністю, що підтримує різноманітні алгоритми балансування і забезпечує високу надійність роботи. Ці пристрої часто потребують складної конфігурації і можуть вимагати значних інвестицій у встановлення і підтримку.

По-друге, витрати на програмне забезпечення також значні. Багато рішень для розподілу навантаження потребують ліцензійного програмного забезпечення, яке може бути досить дорогим, особливо для великих підприємств. Ліцензійні збори на програмне забезпечення для балансування навантаження часто включають не лише початкову покупку, а й щорічні підписки або оновлення, що додає до загальних витрат.

Крім того, необхідно враховувати витрати на інфраструктуру, що включають мережеве обладнання, сервери для резервування та масштабування, а також програмне забезпечення для моніторингу і управління. Всі ці компоненти разом утворюють значні витрати, які можуть бути непомірними для малих і середніх компаній.

Іншим значним недоліком є складність налаштування системи розподілу навантаження. Це питання включає як технічні складнощі, так і потребу в висококваліфікованих спеціалістах.

Налаштування балансувальника навантаження часто вимагає глибоких знань в області мережевих технологій і серверного адміністрування. Система повинна бути налаштована таким чином, щоб ефективно обробляти трафік, забезпечуючи оптимальне використання ресурсів і мінімізацію затримок. Це включає вибір правильного алгоритму балансування, налаштування політик безпеки, інтеграцію з іншими системами і забезпечення високої доступності. Це може бути дуже складним завданням, що вимагає від фахівців не лише технічних знань, а й досвіду роботи з подібними системами.

Крім того, інтеграція з іншими компонентами інфраструктури може бути складною. Системи розподілу навантаження часто потребують інтеграції з базами даних, системами кешування, системами моніторингу і безпеки. Це створює додаткові виклики і може значно ускладнити процес налаштування та тестування. Неправильна конфігурація може призвести до збоїв у роботі системи, що вимагає додаткового часу і ресурсів для виправлення помилок.

Також, процес масштабування системи може бути складним і потребує ретельного планування. Додавання нових серверів і балансувальників може

вимагати перепланування всієї інфраструктури, що включає зміну мережевих налаштувань, оновлення конфігурацій і тестування нових компонентів. Це може бути трудомістким процесом, який потребує значного часу і зусиль.

Другий підхід використання хмарних сервісів для захисту від DDoS-атак. Хмарні сервіси пропонують масштабовані ресурси, які можуть динамічно адаптуватися до зростаючого навантаження під час атаки.

Хмарні ресурси мають вражаючу здатність швидко масштабуватися, коли навантаження зростає, забезпечуючи додаткові потужності для відбиття атак. Це дозволяє системам залишатися стабільними навіть під час пікових навантажень, значно підвищуючи стійкість до DDoS-атак. Крім того, хмарні рішення відзначаються своєю гнучкістю, вони легко адаптуються до різних типів атак, змінюючи свої налаштування в реальному часі. Це означає, що вони можуть автоматично оновлювати правила файрволу, змінювати параметри обробки трафіку та вносити інші необхідні налаштування, що дозволяє оперативно реагувати на нові загрози.

Хмарні провайдери також славляться своєю високою доступністю та надійністю, що гарантує стабільну роботу навіть під час найінтенсивніших атак. Вони мають багаторівневу систему захисту, яка включає фільтри трафіку, системи виявлення аномалій і інші механізми, що надають додатковий рівень безпеки. Крім того, хмарні сервіси забезпечують постійний моніторинг і автоматизоване реагування на атаки, що значно скорочує час реагування на інциденти і підвищує ефективність захисту.

Однак, варто зважати на деякі нюанси. Використання хмарних сервісів пов'язане з певною залежністю від третьої сторони, що може стати проблемою у випадку збоїв або технічних неполадок у провайдера. Це може призвести до непередбачуваних відключень сервісів, проблем з доступом до даних або збоїв у системах обробки трафіку, що може викликати серйозні перерви у роботі. Крім того, передача даних до хмарного провайдера піднімає питання конфіденційності та безпеки даних, оскільки вони можуть зберігатися на віддалених серверах, що підвищує ризик несанкціонованого доступу або витоку

інформації. Тому важливо ретельно управляти політикою доступу, використовувати шифрування даних і впроваджувати додаткові заходи безпеки для захисту від можливих загроз.

Таким чином, хмарні сервіси для захисту від DDoS-атак мають значні переваги в плані масштабованості, гнучкості та доступності, але також варто пам'ятати про можливі проблеми з залежністю від стороннього провайдера і питаннями конфіденційності даних, які потребують особливої уваги при їх використанні.

Третій підхід полягає в забезпеченні відмовостійкості інфраструктури. Це означає створення систем, які можуть продовжувати функціонувати навіть при виході з ладу окремих компонентів.

Цей підхід забезпечує високу доступність і надійність системи, дозволяючи їй працювати безперебійно навіть під час технічних неполадок чи атак. Відмовостійка інфраструктура зазвичай включає дублювання критичних компонентів, використання резервних каналів зв'язку і автоматичне перемикання на резервні системи, якщо основні компоненти виходять з ладу. Це значно знижує ризик простоїв і гарантує безперебійну роботу системи в будь-який час. Однією з головних переваг відмовостійкої інфраструктури є її висока надійність. Дублювання критичних компонентів і резервні системи забезпечують безперебійну роботу, навіть коли основні системи виходять з ладу. Це особливо важливо для підтримки безперебійної роботи бізнес-процесів, що критично для будь-якої компанії. Такий підхід мінімізує ризик втрати даних або збоїв у роботі системи, що може призвести до значних фінансових втрат.

Ще однією важливою перевагою є мінімізація простоїв. Автоматичне перемикання на резервні системи значно знижує ризик простоїв, що особливо важливо під час атак або технічних неполадок. Це дозволяє системам залишатися доступними для користувачів і зберігати свою функціональність, що зменшує ризик втрати доходів і зниження продуктивності. Таке автоматичне перемикання забезпечує швидке відновлення після збоїв, що критично важливо для підтримки високого рівня обслуговування та задоволеності клієнтів.

Підвищена доступність — ще одна важлива перевага відмовостійкої інфраструктури. Резервні канали зв'язку та дублювання систем гарантують високу доступність, що критично для підтримки бізнес-процесів і сервісів. Це забезпечує користувачам стабільний доступ до необхідних ресурсів і послуг, знижуючи ризик втрати доступу до важливої інформації або інструментів. Така доступність допомагає підтримувати безперебійну роботу системи, що важливо для збереження конкурентоспроможності і ефективності бізнесу.

Однак, варто враховувати, що забезпечення відмовостійкості інфраструктури має і свої недоліки. Налаштування відмовостійкої інфраструктури може потребувати значних фінансових витрат на додаткове обладнання, програмне забезпечення і налаштування. Це включає витрати на придбання резервного обладнання, ліцензії на програмне забезпечення та необхідні сервіси для підтримки системи в робочому стані. Крім того, створення і підтримка таких систем вимагає високого рівня технічної експертизи і постійного моніторингу, що може бути досить складним і ресурсомістким процесом. Також важливо регулярно тестувати резервні системи і оновлювати програмне забезпечення для забезпечення їх ефективності, що додає додаткове навантаження на IT-підрозділ. Це створює потребу в постійному навчанні співробітників, оновленні обладнання та програмного забезпечення, що вимагає значних ресурсів і часу. Тому підтримка відмовостійкої інфраструктури є не тільки технічним, але й фінансовим викликом для будь-якої організації.

Четвертий підхід – це Впровадження рішень для захисту від DDoS-атак і аналізу загроз є важливою складовою сучасної кібербезпеки. Ці рішення включають моніторинг мережевого трафіку, виявлення і блокування аномальних дій, а також застосування штучного інтелекту і машинного навчання (AI/ML) для аналізу загроз.

Однією з основних переваг є проактивний підхід, який забезпечують AI/ML. Ці системи можуть виявляти і блокувати загрози на ранніх стадіях, ще до того, як вони встигнуть завдати шкоди інфраструктурі. Це значно знижує ризик

і дозволяє скоротити час на реагування на потенційні атаки, забезпечуючи більш ефективний захист від нових і невідомих загроз.

Ще однією важливою перевагою є моніторинг мережевого трафіку в режимі реального часу. Такий підхід дозволяє швидко реагувати на будь-які зміни в трафіку та оперативно відбивати атаки. Це забезпечує високу швидкість обробки даних і мінімізує час від виявлення загрози до її блокування, що критично важливо для збереження стабільності і безпеки системи.

Не менш важливою є адаптивність таких систем. Використання AI/ML дозволяє їм навчатися на нових даних і змінювати свої алгоритми для виявлення нових типів атак. Це постійне вдосконалення захисних заходів дозволяє системам ефективно адаптуватися до нових загроз, знижуючи ймовірність успішних атак на інфраструктуру.

Однак є і деякі недоліки, про які слід пам’ятати. Вартість таких рішень може бути значною перепоною. Інструменти та сервіси для аналізу загроз часто мають високу вартість, включаючи витрати на придбання, ліцензії і обслуговування. Це може стати серйозною проблемою для малих і середніх компаній, які не мають достатніх ресурсів для інвестицій у такі високотехнологічні рішення.

Складність інтеграції також є важливим недоліком. Впровадження нових рішень у вже існуючу інфраструктуру може вимагати значних зусиль, часу і ресурсів. Це включає не тільки технічні аспекти, а й організаційні зміни, навчання персоналу та адаптацію бізнес-процесів. Інтеграція нових систем може бути складною і трудомісткою, що вимагає ретельного планування і координації між різними підрозділами організації.

Постійний моніторинг трафіку є важливою складовою системи захисту від DDoS-атак і аналізу загроз. Цей підхід дозволяє виявляти аномалії та потенційні загрози в режимі реального часу, що значно підвищує ефективність реагування на атаки.

Постійний моніторинг трафіку дозволяє оперативно виявляти будь-які аномалії або підозрілі активності в мережі. Це забезпечує швидке реагування на

потенційні загрози, що значно знижує ризик успішної атаки. Такий підхід є особливо важливим у випадках DDoS-атак, коли швидкість виявлення та блокування шкідливого трафіку може визначати успішність захисту системи.

Моніторинг у режимі реального часу дозволяє не лише виявляти загрози, але й аналізувати їхню природу та масштаб. Це дає змогу адаптувати захисні заходи відповідно до конкретної загрози, що значно підвищує ефективність системи безпеки. Оперативне виявлення аномальних зразків трафіку дозволяє уникнути значних простоїв і збоїв у роботі системи, що є критично важливим для підтримки стабільності бізнес-процесів.

Одним з основних недоліків постійного моніторингу є висока складність і ресурсомісткість такого підходу. Налагодження систем моніторингу, які можуть обробляти великі обсяги даних у режимі реального часу, вимагає значних технічних і фінансових витрат. Це включає не лише закупівлю відповідного обладнання та програмного забезпечення, але й необхідність у кваліфікованих спеціалістах для налаштування і підтримки таких систем.

Також варто зазначити, що постійний моніторинг може створювати додаткове навантаження на мережу і ресурси системи. Це може призвести до зниження продуктивності і збільшення витрат на обробку даних, особливо в умовах високого трафіку. Крім того, складність інтерпретації великих обсягів даних може ускладнити виявлення справжніх загроз серед великої кількості нормальних подій.

Таким чином, хоча постійний моніторинг трафіку забезпечує значні переваги у сфері захисту від DDoS-атак і аналізу загроз, він також має ряд недоліків, які потребують уваги при плануванні і впровадженні системи безпеки.

Брандмауери веб-додатків (WAF) є важливою складовою системи захисту від DDoS-атак та інших видів кібератак, що спрямовані на веб-додатки. WAF забезпечує фільтрацію, моніторинг і блокування HTTP/HTTPS-трафіку, що направляється до веб-серверів, з метою виявлення та запобігання атакам, таким як SQL-ін’єкції, XSS (межсайтовий скриптинг) та CSRF (використання міжсайтових запитів).

Брандмауери веб-додатків пропонують широкий спектр переваг, що роблять їх важливими для захисту веб-інфраструктури. Однією з головних переваг є спеціалізоване фільтрування трафіку. WAF аналізує кожен запит і відповідь, використовуючи набір правил і підписів, що дозволяє ефективно виявляти та блокувати спроби злому ще до того, як вони досягнуть сервера. Це значно знижує ймовірність успішних атак і мінімізує ризик витоку даних.

Ще однією важливою перевагою є гнучкість у налаштуванні правил. WAF дозволяє створювати та налаштовувати правила, що відповідають конкретним вимогам і загрозам вашого веб-додатку. Це дозволяє адаптувати захист до змін у структуруванні додатка або нових типів атак, що з'являються з часом. Таким чином, WAF забезпечує динамічне оновлення та вдосконалення захисних заходів, що дозволяє підтримувати високий рівень безпеки.

Ще однією важливою перевагою є реальний час моніторинг і реагування. WAF постійно відстежує трафік і миттєво реагує на підозрілі активності, забезпечуючи швидке блокування атакуючих запитів. Це дозволяє зменшити час, необхідний для виявлення і відбиття атак, що є критично важливим для підтримки стабільності і безпеки веб-додатку.

Однак, варто звернути увагу і на деякі недоліки, що можуть бути пов'язані з використанням WAF. Одним з основних недоліків є залежність від правил і підписів. Ефективність WAF значною мірою залежить від актуальності і точності правил, які можуть не завжди враховувати нові або нехарактерні для типових атак методи. Це може призвести до пропуску загроз або надмірного блокування легітимного трафіку, що в свою чергу може негативно вплинути на користувацький досвід.

Складність налаштування та інтеграції є ще одним важливим недоліком. Налаштування WAF може вимагати значних зусиль і глибоких знань у галузі безпеки. Крім того, інтеграція WAF з існуючою інфраструктурою та веб-додатками може бути складною і потребує ретельного планування та тестування. Це включає не лише технічні аспекти, але й необхідність у навчанні персоналу та постійному моніторингу системи.

Таким чином, хоча брандмауери веб-додатків мають значні переваги у забезпеченні захисту від різноманітних загроз, їх впровадження та експлуатація також мають свої недоліки, які потрібно враховувати при розробці стратегії кібербезпеки.

При розробці системи захисту інформації для мікросервісних застосунків, я зупинився на двох методах, які, на мою думку, є найбільш ефективними та сучасними. Ці методи включають впровадження рішень для захисту від DDoS-атак та аналізу загроз, а також забезпечення відмовостійкості інфраструктури з використанням брандмауера веб-додатків (WAF).

Використання сучасних технологій для аналізу загроз дозволяє проактивно виявляти та блокувати загрози до того, як вони досягнуть інфраструктури. Цей метод забезпечує швидкий моніторинг трафіку в реальному часі, що значно знижує ризик атак і збоїв, а також дозволяє системам адаптуватися до нових типів загроз завдяки постійному оновленню алгоритмів. Цей підхід є важливим для забезпечення високого рівня захисту і стабільності мікросервісних застосунків.

Забезпечення відмовостійкості інфраструктури з використанням брандмауера веб-додатків (WAF). Брандмауери веб-додатків є ключовим елементом захисту, який дозволяє ефективно фільтрувати HTTP/HTTPS-трафік, виявляючи і блокуючи загрози, такі як SQL-ін’єкції, XSS та CSRF. Цей метод забезпечує високу гнучкість у налаштуванні правил і дозволяє системам оперативно реагувати на підозрілі активності, що значно знижує ризик злому і забезпечує стабільну роботу веб-додатків. Крім того, WAF забезпечує безперервний моніторинг трафіку в реальному часі, що є критично важливим для захисту мікросервісних систем.

Інтеграція цих методів у систему захисту дозволить створити комплексний підхід до безпеки мікросервісів. Використання сучасних технологій для аналізу загроз забезпечить проактивний захист, що мінімізує ризики і знижує час реагування на атаки. Водночас впровадження WAF гарантує, що всі запити до

веб-додатків будуть ретельно перевірені і захищені від найбільш поширених типів атак.

Такий підхід дозволить захистити від атаки на відмову, але й забезпечить безперебійну роботу системи, знижуючи ризики простоїв і втрат. Це допоможе підтримати високий рівень безпеки, що є критично важливим для будь-якої сучасної організації, яка працює з мікросервісними застосунками.

2.2 Аналіз існуючих рішень для захисту мікросервісних застосунків

Застосування міжмережевих екранів та систем виявлення вторгнень. Міжмережеві екрани (firewalls) та системи виявлення вторгнень (IDS) є одними з найпоширеніших засобів захисту мереж. Вони забезпечують фільтрацію трафіку, блокування підозрілих запитів та виявлення аномалій у мережевому трафіку. Основні переваги цих технологій:

– фільтрація трафіку;

– аналіз аномалій;

– реагування в реальному часі.

Фільтрація трафіку – можливість блокувати небажаний трафік на основі різних критеріїв (IP–адреса, порт, протокол).

Аналіз аномалій – виявлення нетипових патернів у мережевому трафіку, які можуть свідчити про DDoS атаку.

Реагування в реальному часі – можливість автоматично блокувати підозрілі запити, зменшуючи навантаження на мікросервіси. Проте, ці технології мають і недоліки:

– обмежена масштабованість;

– висока вартість.

Обмежена масштабованість – Міжмережеві екрани можуть стати вузьким місцем у системі, якщо кількість трафіку значно збільшується.

Висока вартість – Професійні системи IDS/IPS можуть бути досить дорогими. Використання мережевих пристроїв та сервісів розподілу навантаження. Load balancers (балансувальник навантаження) – це рішення, яке діє як проксі-сервер трафіку та розподіляє мережевий або додатковий трафік між кінцевими точками на кількох серверах. Балансувальники навантаження використовуються для розподілу потужності під час пікового трафіку та для підвищення надійності програм. Вони покращують загальну продуктивність додатків, зменшуючи навантаження на окремі служби чи хмари, і розподіляють попит між різними обчислювальними поверхнями, щоб підтримувати сеанси додатків і мережі. Також воно є важливим компонентом для розподілу вхідного трафіку між різними серверами. Вони забезпечують рівномірний розподіл запитів, що підвищує стійкість системи до перевантажень та атак.

Переваги балансування навантаження. Користувачі та клієнти залежать від здатності майже в реальному часі знаходити інформацію та проводити транзакції. Час затримки або ненадійні та суперечливі відповіді – навіть під час пікового попиту та використання – можуть назавжди відвернути клієнта. І високі стрибки потреби в обчисленнях можуть спричинити хаос для внутрішнього сервера або серверної системи, якщо вхідний попит або «навантаження» є занадто високим, щоб його легко задовольнити.

Переваги використання балансувальника навантаження включають: доступність пргограми;

– масштабність;

– безпека програм;

– продуктивність програми.

Доступність програми – як внутрішні, так і зовнішні користувачі повинні мати можливість покладатися на доступність програми. Якщо програма чи функція не працюють, відстають або зависають, втрачається дорогоцінний час і з’являється потенційне джерело тертя, яке може підштовхнути клієнта до конкурента.

Масштабованість додатка – уявіть, що ви керуєте квитковою компанією, і квитки на популярну виставу оголошують доступними в певну дату та час. Можуть бути тисячі або навіть більше людей, які намагаються отримати доступ до вашого сайту, щоб купити квитки. Без балансувальника навантаження ваш сайт був би обмежений тим, що може вмістити ваш єдиний/перший сервер – що, ймовірно, буде невеликим за такого великого попиту. Натомість ви можете спланувати цей великий сплеск трафіку, встановивши балансувальник навантаження, щоб спрямовувати запити та трафік на інші доступні обчислювальні поверхні. А це означає, що більше клієнтів зможуть отримати бажані квитки.

Безпека програм – балансування навантаження також дозволяє організаціям масштабувати свої рішення безпеки. Одним із основних способів є розподіл трафіку між кількома серверними системами, що допомагає мінімізувати поверхню атаки та ускладнює виснаження ресурсів і перенасичення посилань. Балансувальники навантаження також можуть перенаправляти трафік на інші системи, якщо одна система вразлива або скомпрометована. Крім того, балансувальники навантаження можуть запропонувати додатковий рівень захисту від атак DDoS, перенаправляючи трафік між серверами, якщо певний сервер стає вразливим.

Продуктивність програми – виконуючи все вищезазначене, балансувальник навантаження підвищує продуктивність програми. Підвищуючи безпеку, оптимізуючи час безвідмовної роботи та забезпечуючи масштабованість у разі різкого зростання попиту, балансувальники навантаження забезпечують роботу ваших додатків відповідно до плану — і так, як цього бажаєте ви та ваші клієнти.

Недоліки:

– складність налаштування;
– вразливість до атак;
– використання кешування.

Кешування дозволяє зберігати часто запитувані дані в пам'яті, що значно зменшує навантаження на бекенд–сервіси та підвищує швидкість відповіді на запити. Основні інструменти для кешування:

– redis;

– memcached.

Redis – це надшвидкий і надійний інструмент, який допомагає швидко зберігати та отримувати доступ до великої кількості інформації. Він був створений у 2009 році Сальваторе Санфіліппо. Він дійсно добре керує даними та може виконувати багато різних завдань, оскільки він дуже швидкий і надійний. Redis як спеціальний інструмент, який може виконувати багато різних завдань. Він може зберігати інформацію, надсилати повідомлення, допомагати іншим програмам запам’ятовувати речі та допомагати різним системам працювати разом.

Redis – це супершвидкий і потужний інструмент, який допомагає людям працювати з даними швидко та без проблем. Багато людей і компаній використовують його, тому що він дуже хороший для швидкого отримання інформації, чи то для зберігання інформації, надсилання сповіщень чи будь-чого іншого. Він завжди поруч, щоб гарантувати легкий доступ до даних і правильну роботу.

Memcached – це зручне високопродуктивне сховище даних у пам'яті. Продумане масштабоване рішення з відкритим вихідним кодом рішення з відкритим вихідним кодом забезпечує час відгуку на рівні часток мілісекунди, що дає змогу використовувати його як кеш або сховище сесій. Memcached широко застосовується для підтримки рекламних технологій, майданчиків інтернет-комерції, ігрових, мобільних і інтернет-додатків, а також інших додатків, що працюють у режимі реального часу.

На відміну від баз даних, що зберігають дані на дисках або твердотільних накопичувачах дисках або твердотільних накопичувачах, Memcached зберігає дані в оперативній пам'яті. Оскільки Memcached, як і інші сховища даних типу "ключ-значення" у пам'яті, не потребує доступу до диска, це виключає затримки,

пов'язані з пошуком, і забезпечує доступ до даних за мікросекунди. Крім того, сховище Memcached є розподіленим, тому його можна просто масштабувати шляхом додавання нових вузлів. Багатопоточність Memcached дає змогу швидко нарощувати обчислювальну потужність. Завдяки високій швидкості, масштабованості, простоті, ефективності управління пам'яттю та підтримці API для більшості поширених мов програмування Memcached застосовують для створення масштабного кешу з високою продуктивністю.

Переваги кешування:

– зниження навантаження на бекен;

– підвищення швидкості відповіді;

– масштабованість.

Зниження навантаження на бекенд – це процес, який спрямований на зменшення кількості запитів до основних серверів і баз даних, які обслуговують додаток або вебсайт. Це досягається шляхом зберігання і обробки часто використовуваних даних в кеші, що знаходиться ближче до користувачів або на спеціалізованих серверах.

Підвищення швидкості відповіді означає зменшення часу, необхідного для обробки та повернення запиту користувача. Це досягається шляхом оптимізації процесу доступу до даних і їх обробки. Одним із найефективніших методів для цього є кешування, яке дозволяє зберігати та повторно використовувати часто запитувані дані, зменшуючи необхідність у зверненні до повільніших систем, таких як бази даних або зовнішні API.

Недоліки:

– складність управління;

– вразливість до атак.

Технології аутентифікації та авторизації. Забезпечення надійної аутентифікації та авторизації користувачів є ключовим аспектом безпеки мікросервісів. Для цього можуть використовуватися різні технології.

Oauth це відкритий стандарт авторизації, який дозволяє користувачам відкривати доступ до своїх приватних даних (фотографії, відео, списки контактів), що зберігаються на одному сайті, іншому сайту, без необхідності вводу імені користувача та паролю.

OAuth дозволяє користувачам роздавати сайтам маркери доступу, до даних що розміщуються на сайтах-сервісах. Кожен маркер доступу надає доступ конкретному сайту (наприклад, сайту редагування відео) до конкретних ресурсів (наприклад, тільки відео від конкретного альбому) та на визначений термін (наприклад, на наступні 2 години). Це дозволяє користувачам надавати доступ третім сайтам до їх інформації, що зберігається на інших сайтах — постачальниках послуг, не передаючи повною мірою самих даних та без застосування імені/паролю.

JWT (JSON Web Tokens – JSON web token (JWT), вимовляється як "jot", є відкритим стандартом (RFC 7519), який визначає компактний і самодостатній спосіб безпечної передачі інформації між сторонами як об’єкт JSON. Знову ж таки, JWT є стандартом, що означає, що всі JWT є маркерами, але не всі маркери є JWT.

Через відносно невеликий розмір JWT можна надіслати через URL-адресу, через параметр POST або всередині заголовка HTTP, і він швидко передається. JWT містить всю необхідну інформацію про сутність, щоб уникнути повторного запиту до бази даних. Одержувачу JWT також не потрібно викликати сервер для перевірки маркера.

Переваги:

– безпечний доступ;

– масштабованість;

– гнучкість.

Недоліки:

– складність реалізації;

– залежність від сторонніх сервісів.

Моніторинг мережевого трафіку дозволяє виявляти аномальні сплески трафіку та своєчасно реагувати на потенційні загрози. Основні інструменти для моніторингу. Grafana: Платформа для візуалізації даних, зібраних Prometheus та іншими системами.

Grafana – це багатоплатформний веб-додаток для аналітики та інтерактивної візуалізації з відкритим вихідним кодом. Він може створювати діаграми, графіки та сповіщення для Інтернету при підключенні до підтримуваних джерел даних.

Існує також ліцензійна версія Grafana Enterprise з додатковими можливостями, яка продається як самостійна інсталяція або через обліковий запис на хмарному сервісі Grafana Labs. Її можна розширювати за допомогою системи плагінів. Кінцеві користувачі можуть створювати складні панелі моніторингу за допомогою інтерактивних конструкторів запитів. Продукт поділяється на фронт-енд і бек-енд, написані на TypeScript і Go відповідно. Як інструмент візуалізації, Grafana може використовуватися як компонент у стеках моніторингу, часто в поєднанні з базами даних часових рядів, такими як InfluxDB, Prometheus і Graphite; платформами моніторингу, такими як Sensu, Icinga, Checkmk, Zabbix, Netdata і PRTG; SIEM, такими як Elasticsearch, OpenSearch і Splunk; та іншими джерелами даних. Інтерфейс користувача Grafana спочатку базувався на версії 3 Kibana.

Переваги моніторингу:

– виявлення аномалій;

– реагування в реальному часі;

– аналітика та звітність.

Недоліки:

– велика кількість даних;

– складність налаштування.

2.3 Вибір інструментів та підходів для розробки системи захисту

На основі проведеного аналізу було обрано наступні інструменти та підходи для розробки системи захисту мікросервісних застосунків від атак на відмову:

- API Gateway (Шлюз API);
- rate Limiting;
- кешування;
- моніторинг та виявлення аномалій.

API Gateway буде використовуватися для централізованого управління трафіком та забезпечення безпеки мікросервісів. Його основні функції включають маршрутизацію запитів, аутентифікацію та авторизацію користувачів, а також застосування політик безпеки. Вибір API Gateway обґрунтований наступними перевагами:

Для централізованого управління трафіком, аутентифікації та авторизації користувачів, а також застосування політик безпеки. Шлюз API приймає запити API від клієнта, обробляє їх на основі визначених політик, спрямовує їх до відповідних служб і комбінує відповіді для спрощеної взаємодії з користувачем. Як правило, він обробляє запит, викликаючи кілька мікросервісів і агрегуючи результати. Він також може перекладати між протоколами в застарілих розгортаннях. Шлюз API приймає запити API від клієнта, обробляє їх на основі визначених політик, спрямовує їх до відповідних служб і комбінує відповіді для спрощеної взаємодії з користувачем. Як правило, він обробляє запит, викликаючи кілька мікросервісів і агрегуючи результати. Він також може перекладати між протоколами в застарілих розгортаннях.

Для додатків на основі мікросервісів шлюз API діє як єдина точка входу в систему. Він знаходиться перед мікросервісами та спрощує клієнтські реалізації та програму мікросервісів, відокремлюючи складність програми від її клієнтів. В архітектурі мікросервісів шлюз API відповідає за маршрутизацію запитів,

композицію та застосування політики. Він обробляє деякі запити, просто направляючи їх до відповідної серверної служби, і обробляє інші, викликаючи кілька серверних служб і агрегуючи результати. Шлюз API може надавати інші можливості для мікросервісів, наприклад автентифікацію, авторизацію, моніторинг, балансування навантаження та обробку відповідей, розвантажуючи впровадження нефункціональних вимог на рівень інфраструктури та допомагаючи розробникам зосередитися на основній бізнес-логіці, прискорюючи випуски програм.

Переваги:

– централізоване управління трафіком;

– гнучкість налаштувань;

– захист від атак.

Rate Limiting дозволить обмежити кількість запитів від окремих користувачів або IP–адрес, що допоможе запобігти перевантаженню системи та захистити її від DDoS атак.

Для обмеження кількості запитів, які можуть бути зроблені певним користувачем або IP–адресою за визначений проміжок часу. Обмеження швидкості – це метод обмеження мережевого трафіку, щоб запобігти виснаженню системних ресурсів користувачами. Обмеження швидкості ускладнює зловмисникам перевантажувати систему та спричиняти атаки, такі як відмова в обслуговуванні (DoS). Це означає, що зловмисники заповнюють цільову систему запитами та споживають занадто багато мережевої ємності, пам’яті та пам’яті.

API, які використовують обмеження швидкості, можуть гальмувати або тимчасово блокувати будь-якого клієнта, який намагається зробити занадто багато викликів API. Це може уповільнити запити обмеженого користувача на певний час або взагалі відхилити їх. Обмеження швидкості гарантує, що законні запити можуть досягати системи та отримувати доступ до інформації без впливу на загальну продуктивність програми. Впровадження частотного обмеження відіграє вирішальну роль у зміцненні заходів безпеки сучасного кіберзахисту.

Досліджуючи різні методи атак, які маніпулюють швидкістю вхідних запитів, ця проактивна стратегія служить надійним стримуючим засобом проти зловмисних дій. Однією з таких загроз, яка ефективно нейтралізується за допомогою обмеження частоти, є атака розподіленої відмови в обслуговуванні (DDoS), метою якої є перевантаження системи надмірним потоком трафіку, що робить її недоступною для законних користувачів. Встановивши обмеження на обсяг запитів з будь-якого джерела, потенційний вплив загроз DDoS можна значно пом’якшити.

Крім того, загрозі збору та крадіжки даних, коли конфіденційна інформація збирається для незаконних цілей, можна ефективно протистояти за допомогою впровадження обмеження швидкості. Виявляючи та блокуючи зловмисних ботів, які займаються збиранням даних, можна зберегти цілісність цільових веб-сайтів, запобігаючи потраплянню критичної інформації в чужі руки. Крім того, практика заповнення облікових даних через автоматичних ботів становить серйозний ризик, оскільки дозволяє зловмисникам отримати несанкціонований доступ до облікових записів шляхом систематичного тестування викраденої інформації користувача.

Застосовуючи обмеження на кількість спроб входу, можна виявити ненормальні шаблони та запобігти потенційним порушенням до того, як станеться серйозна шкода. По суті, стратегічне застосування частотного обмеження служить життєво важливим компонентом комплексної системи кібербезпеки, захищаючи цифрові активи та зміцнюючи захист від безлічі потенційних загроз. Крім того, проблема DDoS-атак полягає в їх децентралізованому характері, коли запити розподіляються між численними IP-адресами, щоб уникнути виявлення. У цьому сценарії потрібне розширене рішення безпеки для ідентифікації та агрегування запитів із різних місць, розглядаючи їх як частину скоординованої атаки, а не як окремі джерела.

Подібним чином атаки грубої сили, які передбачають невпинне вгадування облікових даних для входу, можна запобігти за допомогою заходів з обмеження частоти. Завдяки обмеженню кількості спроб випадкових облікових даних

мережеві ресурси захищені від надмірного споживання широкомасштабними атаками.

Основні переваги:

– зменшення навантаження на мікросервіси;

– захист від зловживань.

Кешування

Використання Redis для кешування дозволить знизити навантаження на бекенд–сервіси та підвищити швидкість відповіді на запити.

Більш швидкий доступ і скорочення операцій введення-виведення – Redis використовує оперативну пам'ять для отримання даних, що значно швидше, ніж отримання даних з дискового сховища. Крім того, якщо дані вже зберігаються в оперативній пам'яті, кількість операцій введення-виведення значно скорочується.

Ключові міркування:

– обмежений обсяг даних;

– збереження даних;

– динамічні дані.

Ідеальний варіант використання Бази даних і структури даних in-memory ефективні в застосунках, де швидкий доступ до даних і час відгуку мають першочергове значення, наприклад під час обробки даних у реальному часі, кешування, аналізу та написання сценаріїв Redis - ідеальний вибір для застосунків, які потребують продуктивності в реальному часі та швидких даних Redis - ідеальне рішення для застосунків, які потребують продуктивності в реальному часі та швидкого доступу до даних.

Моніторинг та виявлення аномалій

Інтеграція з Prometheus та Grafana дозволить відслідковувати аномалії у трафіку та своєчасно реагувати на потенційні загрози. Основні переваги:

– реагування в реальному часі;

– аналітика та звітність.

Автоматичне масштабування ресурсів

Використання Kubernetes для автоматичного масштабування мікросервісів дозволить динамічно реагувати на зміну навантаження та забезпечити безперебійну роботу системи.

Використання Kubernetes є Підтримка роботи контейнерних програм може бути складною, оскільки вони часто включають багато контейнерів, розгорнутих на різних машинах. Kubernetes надає спосіб планувати та розгортати ці контейнери, а також масштабувати їх до бажаного стану та керувати їхніми життєвими циклами. Використовуйте Kubernetes, щоб реалізувати свої програми на основі контейнерів портативним, масштабованим і розширюваним способом.

Оскільки додатки складаються з декількох контейнерів, розгорнутих на декількох серверах, управління ними стає дедалі складнішим. Щоб впоратися з цією складністю, Kubernetes надає API з відкритим вихідним кодом для управління тим, де і як запускаються ці контейнери.

Kubernetes організовує кластери віртуальних машин і планує запуск контейнерів на цих віртуальних машинах, ґрунтуючись на доступних обчислювальних ресурсах і вимогах до ресурсів кожного контейнера. Контейнери об'єднуються в модулі, які є основною одиницею роботи Kubernetes, і ці контейнери масштабуються до потрібного стану.

Kubernetes також автоматично керує виявленням сервісів, забезпечує балансування навантаження, стежить за розподілом ресурсів і масштабується залежно від використання обчислень. Крім того, він перевіряє стан окремих ресурсів і забезпечує самовідновлення додатків шляхом автоматичного перезапуску або реплікації контейнерів.

Основні переваги:

– гнучкість масштабування;

– підвищення стійкості системи.

2.4 Висновок розділу

Ми розглянули такі інструменти, як API Gateway, який керує трафіком і допомагає автентифікувати та маршрутизувати запити, щоб зменшити навантаження на систему. Цей інструмент також може обмежити частоту надсилання запитів, щоб запобігти атакам. Підсумовуючи, ми виявили, що використання сучасних інструментів і методів має вирішальне значення для забезпечення безпеки та ефективності додатків мікросервісів у сучасному швидкому цифровому світі.

Розглядаючи способи захисту додатків мікросервісів від кібератак, ми виявили, що традиційні методи не завжди можуть працювати добре. Мікросервіси потребують спеціальних методів, щоб залишатися в безпеці та добре працювати, оскільки вони розкидані й мають працювати швидко. Використовуючи комбінацію цих інструментів, ми можемо переконатися, що програми мікросервісів захищені від атак. Це не тільки робить їх більш безпечними, але й допомагає їм працювати безперебійно та легко розвиватися.

Це важливо для компаній, які покладаються на безперебійну роботу своїх цифрових послуг. Інструменти моніторингу, такі як Prometheus і Grafana, допомагають стежити за справністю системи та швидко виявляти все незвичайне, що може становити загрозу. Кешування — ще один метод, який ми розглянули. Це допомагає прискорити обробку та покращити роботу системи, зберігаючи дані, які часто використовуються.

3 РЕАЛІЗАЦІЯ СИСТЕМИ ЗАХИСТУ ВІД DDoS-АТАКИ

3.1 Проектування архітектури системи захисту

У сучасному технологічному світі, де цифрові сервіси відіграють вирішальну роль у повсякденному житті та бізнесі, захист від атак на відмову обслуговування (DDoS) стає надзвичайно важливим. Мікросервісна архітектура, яка забезпечує масштабованість і гнучкість додатків, також потребує особливої уваги до безпеки через її розподілений характер.

Моя потреба в розробці власного програмного забезпечення виникла ще на етапі огляду рішень для захисту від DDoS-атак від інших компаній та розробників. Якщо поглянути на них, то кожне з них має свої сильні та слабкі сторони. Деякі з них пропонують кращі функції за доступною ціною, тоді як інші вимагають додаткових витрат для підвищення рівня захисту. Зібравши всю інформацію, ми вирішили створити власне програмне забезпечення для захисту від DDoS-атак з базовими та розширеними функціями інших рішень.

Мета полягає в тому, щоб створити систему, яка забезпечить користувачам безкоштовний та ефективний захист від DDoS-атак і підвищить рівень безпеки без додаткових витрат. Наше програмне забезпечення включає можливість автоматичного виявлення та відключення атак, забезпечуючи безперервну доступність сервісів та захист від зловмисних дій кіберзлочинців.

Такі рішення необхідні для забезпечення стабільної роботи мікросервісів, які є основою багатьох сучасних цифрових додатків. Цей підхід ефективно захищає інфраструктуру та знижує ризик фінансових втрат і репутаційних збитків. Крім того, вони також відповідають вимогам новітнього законодавства щодо захисту даних та безпеки системи. Таким чином, продукти поєднують в собі найкращі риси існуючих рішень, а також пропонують інноваційні функції, які роблять захист від DDoS доступним для всіх користувачів без додаткових витрат. Враховуючи важливість захисту мікросервісів від DDoS-атак, важливо розглянути, які рішення вже існують на ринку та як наше програмне забезпечення виділяється серед них.

Звичайна система захисту від DDoS-Атаки є собою виявленя якихось аномалій в мережевому трафіку, або вимкнути програму приклад на схемі роботи звичайного захисту Рисунок 3.1.

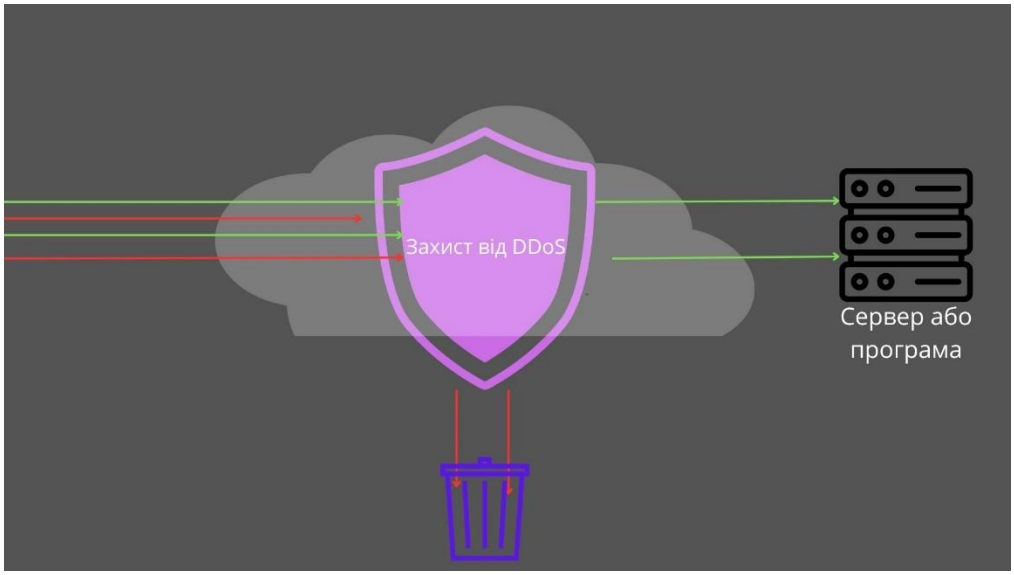


Рисунок 3.1 – Схема захисту від DDoS-атак

Розглянемо мою блок схему програми захисту від атаки на відмову Рисунок 3.2.

Рисунок 3.2 – блок схема захисту від DDoS-атки

На рисунку зображено роботу звичайного захисту від DDoS-атаки, зазвичай вони дивляться та трафіком і шукають якісь дивні зблішення трафіку або аномалії.

3.2 Оцінка ефективності

Розглянемо мою блок-схему в ній ми бачимо щось типу радару інакше кажучі аналіз мережевого трафіку, якщо він виявляє якусь анамалію або підозрілу активність він перевіряє чи це дійсно DDoS-атака чи просто наплив людей, якщо це атакато він запускає захисну систему від атаки записує інформацію про неї та повідомляєе про атаку адміністратору.

Як ми можемо побачити, в коді присутньо багато функцій, які забезпечують різноманітні операції захисту та роботи з даними. Розглянемо їх детальніше. Спочатку ми бачимо, що основними функціями є ініціалізація та завантаження бібліотеки. Функції pyarmor_init та pyarmor_runtime відповідають за ініціалізацію бібліотеки _pytransform, яка є ядром всього механізму захисту. Вони налаштовують параметри роботи бібліотеки, включаючи шлях до файлів, режим роботи (runtime чи ні), платформу, суфікс та інші необхідні параметри. Це забезпечує коректну роботу всієї системи захисту. Ще одна важлива частина коду стосується роботи з ліцензіями. Функції generate_license_file, generate_license_key, get_registration_code та get_expired_days використовуються для генерації та отримання ліцензійних файлів і ключів, а також для перевірки терміну дії ліцензій. Це дозволяє контролювати доступ до захищених ресурсів і забезпечує, що лише авторизовані користувачі можуть користуватися програмою. Код також забезпечує високий рівень безпеки завдяки функціям шифрування та захисту даних. Функції encrypt_code_object, clean_obj та clean_str відповідають за шифрування об'єктів коду та очищення різних типів об'єктів від конфіденційних даних. Це запобігає несанкціонованому доступу до чутливих даних і підвищує загальний рівень безпеки системи.

Для визначення поточної платформи та архітектури системи використовуються функції format_platform, _match_features та _gnu_get_libc_version. Вони дозволяють правильно завантажувати відповідні бібліотеки та налаштовувати параметри роботи залежно від особливостей платформи. Це забезпечує сумісність програми з різними операційними системами та апаратним забезпеченням. Щодо захисту та безпеки, то ініціалізація та завантаження бібліотеки, завдяки функціям pyarmor_init та _load_library, забезпечує правильне завантаження та ініціалізацію бібліотеки _pytransform. Це включає налаштування шляху до бібліотеки, перевірку підтримуваних платформ та архітектур, і встановлення необхідних параметрів безпеки. Використання функцій encrypt_code_object, clean_obj, та clean_str дозволяє безпечно шифрувати дані та очищати конфіденційну інформацію з пам'яті, що запобігає несанкціонованому доступу до чутливих даних. Функції generate_license_file та generate_license_key забезпечують генерацію унікальних ліцензійних файлів та ключів, що дозволяє контролювати доступ до захищених ресурсів. Функції get_registration_code та get_expired_days дозволяють перевіряти стан ліцензії та її термін дії. Використання функцій get_hd_info та show_hd_info дозволяє отримувати інформацію про апаратне забезпечення, що може бути використано для додаткової перевірки ліцензій та забезпечення захисту. Зручність використання та інтерфейс також є важливими аспектами даного коду. Завдяки зрозумілому інтерфейсу, функції get_license_info, get_license_code та get_user_data дозволяють легко отримати необхідну інформацію про ліцензії та користувацькі дані. Це забезпечує зручність у використанні і дозволяє швидко знайти необхідні дані без складних налаштувань. Загалом, цей код забезпечує надійний механізм захисту та роботи з даними завдяки широкому спектру функцій для шифрування, ліцензування та отримання інформації про систему. Він також має зрозумілий інтерфейс, що робить його легким у використанні навіть для недосвідчених користувачів.

Оцінка ефективності створеного додатка комплексного захисту від DDoS-атак була проведена за допомогою експерименту, метою якого було порівняння

працездатності програми, що перебуває під DDoS-атакою, з підключеним захистом і без нього. Для цього було організовано два стрес-тести з однаковими параметрами атаки на заздалегідь підготовлену програму на захищену і незахищену IP-адреси.

Ступінь працездатності програми із захистом від DDoS-атак на фільтрацію небажаного трафіку оцінювалася за допомогою метрик завантаження процесора і мережевого пристрою. Для тестування використовувалися наступні налаштування:

Програмне середовище:

– проста програма, розгорнута за допомогою стандартного LAMP-стека;

– віртуальний сервер з 1 vCPU і 1.7 GB оперативної пам'яті на Debian;

– інструменти моніторингу;

– утиліта Netdata для перегляду відомостей про систему в реальному часі;

– інструмент для проведення стрес-тесту;

– IP Stresser, що надає можливість створення тесту з об'ємом атаки до 3 Гбіт/с.

Середовищем для розміщення програми було обрано віртуальне приватне хмара Google Cloud за можливість швидкого створення віртуальної машини і підключення публічної підмережі. Було створено сервер всередині проекту, використовуючи готовий образ Debian 9. Для встановлення необхідного програмного забезпечення було обрано готовий пакетний стек Bitnami для хмарної платформи Google Cloud, через можливість швидкого та простого встановлення. Встановлення здійснювалося через інтерфейс Google Cloud.

Після встановлення програми було перевірено її працездатність. Аналіз стандартних правил міжмережевого екрану показав, що система не зможе протистояти DDoS-атаці, оскільки відсутні правила для фільтрації потенційно злочинного трафіку.

В результаті проведених стрес-тестів було визначено ефективність захисту програми від DDoS-атак шляхом порівняння показників завантаження процесора і мережевого пристрою під час атак з увімкненим захистом і без нього.

Перейдемо до стрес-тесту системи на незахищену IP-адресу. Конфігурації атаки були визначені заздалегідь і включали інтенсивний потік даних, спрямований на новостворену програму. Під час цього тесту, адреса цільового хоста була призначена новоствореній програмі, яка при реальному використанні, без додаткових налаштувань та механізмів захисту, обслуговуватиме весь трафік, що надходить на неї.

У відсутності належних заходів захисту, система не фільтруватиме підозрілий трафік, що в результаті призведе до відмови в її працездатності. Цей тест показав, як важливо мати належні механізми захисту для забезпечення стабільної роботи програмного забезпечення під час DDoS-атак.

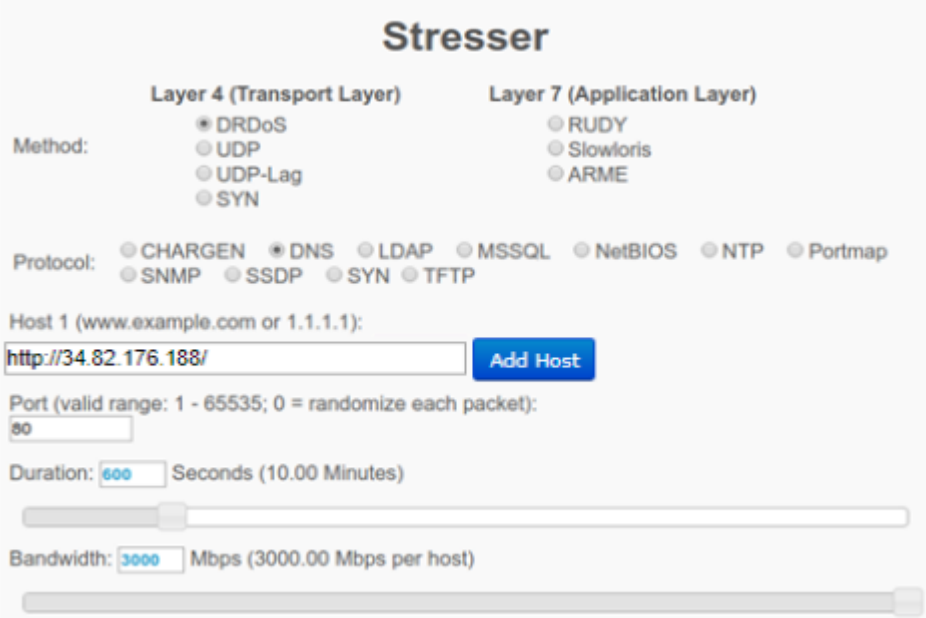


Рисунок 3.3 – Конфігурації стрес-тесту

На рисунку 3.4 видно, що майже миттєво відбувається перевантаження процесора і об'єм прийнятого трафіку досягає до 3Гбіт/с.

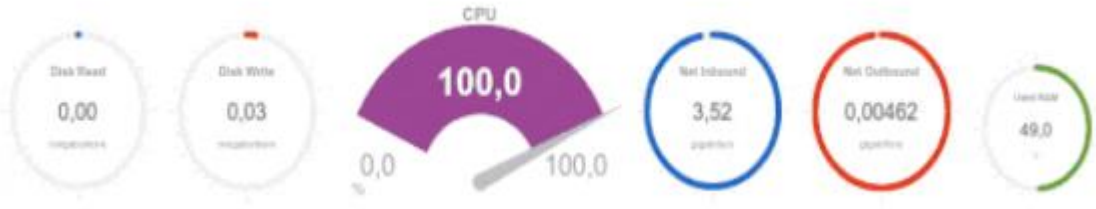


Рисунок 3.4 – Метрика веб-сайту без захисту

Резюмуючи отримані результати, очевидно, що веб-сервер повністю прийняв на себе весь обсяг тестової атаки, а в разі збільшення навантаження стався б відмова працездатності програми. Далі ми протестуємо доступність веб-сервера з підключеним захистом від DDoS атак. Для цього підключимося через SSH до нашого сервера та включимо 39 розроблений захист, використовуючи команду ddosprotect.php –a (рис. 3.6). В результаті ми бачимо список новий список правил iptables (рис. 3.7).

```
progres_drumer@wordpress-1-vm:~$ ^C
progres_drumer@wordpress-1-vm:~$ iptables -L -n -v
-bash: iptables: command not found
progres_drumer@wordpress-1-vm:~$ sudo /sbin/iptables -L -n -v
Chain INPUT (policy ACCEPT 755K packets, 502M bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 758K packets, 375M bytes)
 pkts bytes target     prot opt in     out     source               destination
progres_drumer@wordpress-1-vm:~$ ./ddosprotect.php -a
```

Рисунок 3.5 – Список правил iptables до включення захисту та застосування захисту

```
# Generated by iptables-save v1.6.0 on Mon Jun  1 15:15:40 2020
*mangle
:PREROUTING ACCEPT [161:12980]
:INPUT ACCEPT [161:12980]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [79:11948]
:POSTROUTING ACCEPT [79:11948]
-A PREROUTING -m conntrack --ctstate INVALID -j DROP
-A PREROUTING -p tcp -m tcp ! --tcp-flags FIN,SYN,RST,ACK SYN -m conntrack --ctstate NEW -j DROP
-A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,RST FIN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags ACK,URG URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags PSH,ACK PSH -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,PSH,ACK,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,PSH,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,PSH,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,ACK,URG -j DROP
-A PREROUTING -s 224.0.0.0/3 -j DROP
-A PREROUTING -s 169.254.0.0/16 -j DROP
-A PREROUTING -s 172.16.0.0/12 -j DROP
-A PREROUTING -s 192.0.2.0/24 -j DROP
-A PREROUTING -s 192.168.0.0/16 -j DROP
-A PREROUTING -s 10.0.0.0/8 -j DROP
-A PREROUTING -s 0.0.0.0/8 -j DROP
-A PREROUTING -s 240.0.0.0/5 -j DROP
-A PREROUTING -s 127.0.0.0/8 ! -i lo -j DROP
-A PREROUTING -p icmp -j DROP
-A PREROUTING -f -j DROP
COMMIT
# Completed on Mon Jun  1 15:15:40 2020
```

Рисунок 3.6 – Результат роботи команди ./ddos_protect.php -a

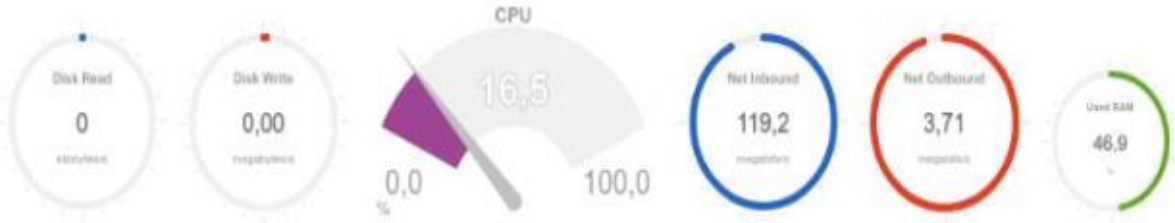


Рисунок 3.7 – Метрика програми з використанням захисту

Таким чином, можна зробити висновок, що використання подібного технічно-програмного рішення для захисту від DDoS-атак забезпечує певний рівень безпеки програми.

3.3 Висновки розділу

У цьому розділі детально розглядається й оцінюється ефективність запропонованої системи захисту та управління даними. Основна мета полягала в тому, щоб продемонструвати, як різні функції забезпечують високий рівень безпеки та ефективності.

По-перше, система забезпечує стабільну і безпечну роботу завдяки правильній ініціалізації та конфігурації бібліотеки. Це забезпечує оптимальне використання ресурсів і захист даних.

По-друге, функція управління ліцензіями забезпечує контроль доступу до захищених ресурсів. Це гарантує конфіденційність і гарантує, що тільки авторизовані користувачі можуть використовувати додаток. По-третє, функції шифрування та очищення конфіденційної інформації забезпечують високий рівень безпеки даних. Це запобігає несанкціонованому доступу до конфіденційних даних і підвищує загальну безпеку системи. Крім того, визначення поточної платформи та архітектури системи забезпечує сумісність застосунків із різними операційними системами та апаратним забезпеченням. Це сприяє широкій доступності додатків на різних пристроях. Це дуже важливо, оскільки дає змогу швидко й ефективно використовувати застосунок без складного налаштування. Таким чином, розглянута система забезпечує надійний механізм захисту і роботи з даними завдяки широкому набору функцій, таких як шифрування, ліцензування та інформація про систему. Простий і зрозумілий інтерфейс і високий рівень безпеки роблять її ефективним рішенням для захисту конфіденційної інформації.

ВИСНОВКИ

У ході виконання даної кваліфікаційної роботи було досліджено та реалізовано комплекс заходів для забезпечення захисту мікросервісних застосунків від кіберзагроз. Основною метою роботи було розроблення ефективної системи захисту, здатної проактивно виявляти та нейтралізувати потенційні загрози, забезпечуючи стабільну та безпечну роботу мікросервісної архітектури. На початку роботи було проведено детальний аналіз мікросервісної архітектури, що дозволило визначити її основні переваги та недоліки. Було виявлено, що розподілена природа мікросервісів підвищує їх гнучкість і масштабованість, але водночас збільшує кількість потенційних векторів атак. У результаті проведеного дослідження було визначено основні типи загроз для мікросервісних систем, включаючи DDoS-атаки, SQL-ін’єкції, XSS, CSRF та інші. Для кожного типу загроз було розроблено та впроваджено відповідні заходи захисту. Зокрема, було створено систему моніторингу та аналізу мережевого трафіку, яка дозволяє оперативно виявляти підозрілу активність і запускати механізми захисту. Важливим елементом захисту стала інтеграція брандмауера веб-додатків (WAF), що забезпечує ефективну фільтрацію HTTP/HTTPS-трафіку, виявлення та блокування загроз. Крім того, було реалізовано систему двоетапного шифрування даних, що забезпечує високий рівень захисту конфіденційної інформації. У процесі роботи також було приділено увагу зручності використання розробленої системи. Завдяки зрозумілому та інтуїтивно зрозумілому інтерфейсу, система може бути легко використана навіть недосвідченими користувачами, що значно підвищує її ефективність та доступність. Таким чином, результати даної кваліфікаційної роботи підтверджують, що розроблена система захисту мікросервісних застосунків забезпечує високий рівень безпеки та надійності. Вона дозволяє оперативно виявляти та нейтралізувати потенційні загрози, зберігаючи стабільну роботу системи та захищаючи конфіденційні дані користувачів. Впровадження цієї системи в реальні умови експлуатації може значно підвищити рівень безпеки мікросервісних застосунків та знизити ризики, пов’язані з кіберзагрозами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1 What are microservices?. microservices. URL: https://microservices.io/ (date of access: 20.05.2024).

2 Essential characteristics of the microservice architecture: loosely coupled. microservices. URL: https://microservices.io/post/architecture/2023/03/28/microservice-architecture-essentials-loose-coupling.html (date of access: 20.05.2024).

3 Essential characteristics of the microservice architecture: independently deployable. microservices. URL: https://microservices.io/post/architecture/2022/05/04/microservice-architecture-essentials-deployability.html (дата звернення: 20.05.2024).

4 DoS-атака. wikipedia. URL: https://uk.wikipedia.org/wiki/DoS-атака#cite_note-1 (дата звернення: 20.05.2024).

5 Розподілена атака на відмову в обслуговуванні. eset. URL: https://www.eset.com/ua-ru/support/information/entsiklopediya-ugroz/distributed-denial-of-service/ (дата звернення: 20.05.2024).

6 Захист від DDoS: 8 простих тактик. blogs.blackberry. URL: https://blogs.blackberry.com/en/2022/11/ddos-attack-8-simple-prevention-and-mitigation-strategies (дата звернення: 20.05.2024).

7 Fruhlinger J. DDoS attacks: Definition, examples, and techniques. csoonline. URL: https://www.csoonline.com/article/571981/ddos-attacks-definition-examples-and-techniques.html (date of access: 21.05.2024).

8 Document-Based Knowledge Discovery with Microservices Architecture / H. K. Gidey et al. Communications in Computer and Information Science. Cham, 2022. P. 146–161. URL: https://doi.org/10.1007/978-3-031-08277-1_13 (date of access: 21.05.2024).

9 Context-Aware and QoS Prediction-based Cross-Domain Microservice Instance Discovery / H. Liu et al. 2022 IEEE 13th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 21–23 October

2022. 2022. URL: https://doi.org/10.1109/icsess54813.2022.9930241 (date of access: 21.05.2024).

10 Bhattacharyya D. K. DDoS Attacks. Chapman and Hall/CRC, 2016. URL: https://doi.org/10.1201/b20614 (date of access: 21.05.2024).

11 Availability and Scalability Optimized Microservice Discovery from Enterprise Systems / A. A. C. De Alwis et al. Lecture Notes in Computer Science. Cham, 2019. P. 496–514. URL: https://doi.org/10.1007/978-3-030-33246-4_31 (date of access: 21.05.2024).

12 A Digital Twin Platform for Industrie 4.0 / M. Redeker et al. Data Spaces. Cham, 2022. P. 173–200. URL: https://doi.org/10.1007/978-3-030-98636-0_9 (date of access: 21.05.2024).

13 Liu H., Cao Z., Zhang X. An Efficient Algorithm of Context-Clustered Microservice Discovery. the 2nd International Conference, Hohhot, China, 22–24 October 2018. New York, New York, USA, 2018. URL: https://doi.org/10.1145/3207677.3277949 (date of access: 21.05.2024).

14 Remodularization Analysis for Microservice Discovery Using Syntactic and Semantic Clustering / A. A. C. De Alwis et al. Advanced Information Systems Engineering. Cham, 2020. P. 3–19. URL: https://doi.org/10.1007/978-3-030-49435-3_1 (date of access: 21.05.2024).

15 Theoretical and Experimental Methods for Defending Against DDOS Attacks. Elsevier, 2016. URL: https://doi.org/10.1016/c2015-0-05397-7 (date of access: 21.05.2024).

16 Cao W. Theoretical and experimental studies of surface and interfacial phenomena involving steel surfaces : doctoral thesis. 2010. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-26194 (дата звернення: 21.05.2024).

17 Demtröder W. Molecular physics: Theoretical principles and experimental methods. Weinheim : Wiley-VCH, 2005. 470 c.

18 Doh J.-H., n/a. Experimental and Theoretical Studies of Normal and High Strength Concrete Wall Panels. 2003. URL: http://www4.gu.edu.au:8080/adt-root/public/adt-QGU20030605.114125 (дата звернення: 21.05.2024).

19 Experimental Chaos Conference (2nd 1993 Arlington, Va.). Proceedings of the 2nd Experimental Chaos Conference: October 6-8, 1993, Arlington, Virginia / ред.: D. W. L, United States. Office of Naval Research. Singapore : World Scientific, 1995. 368 с.

20 Hillman E. M. C. Experimental and theoretical investigations of near infrared tomographic imaging methods and clinical applications : thesis. 2002. URL: http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.268884 (дата звернення: 21.05.2024).

21 Khoshrou S. H. Theoretical and experimental investigation of wall-control blasting methods : thesis. 1996. URL: http://digitool.Library.McGill.CA:80/R/?func=dbin-jump-full&object_id=40161 (дата звернення: 21.05.2024).

22 Logunova O., Romanov P., Il'ina E. Processing of experimental data on a computer. ru : INFRA-M Academic Publishing LLC., 2020. URL: https://doi.org/10.12737/1064882 (дата звернення: 21.05.2024).

23 North Atlantic Treaty Organization. Advisory Group for Aerospace Research and Development. Fluid Dynamics Panel. Symposium. Theoretical and experimental methods in hypersonic flows =: Les méthodes théoriques et expérimentales pour l'etude des ecoulements hypersoniques. Neuilly sur Seine, France : AGARD, 1993.

24 Rabus J. Mass Spectrometry of Carbohydrates by Experimental and Theoretical Methods. 2021. URL: http://rave.ohiolink.edu/etdc/view?acc_num=ohiou1628688928273698 (дата звернення: 21.05.2024).

25 Rosenblad B. L. Experimental and theoretical studies in support of implementing the spectral-analysis-of-surface-wave (SASW) method offshore /. 2000. URL: http://wwwlib.umi.com/cr/utexas/main (дата звернення: 21.05.2024).

26 Rostem K. Theoretical and experimental methods for the development of superconducting transition-edge sensors : thesis. 2010. URL: http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.608722 (дата звернення: 21.05.2024).

27 Sansour C. Generalized Continua and Dislocation Theory: Theoretical Concepts, Computational Methods and Experimental Verification. Vienna : Springer Vienna, 2012.

28 Sansour C., Skatulla S. Generalized continua and dislocation theory: Theoretical concepts, computational methods and experimental verification. Wien : Springer Verlag, 2012. 317 c.

29 Schamel G. C. Experimental and theoretical investigation of optimal control methods with model reduction : dissertation. 1989. URL: http://hdl.handle.net/10919/54412 (дата звернення: 21.05.2024).

30 Theoretical and Experimental Methods for Defending Against DDOS Attacks. Elsevier, 2016. URL: https://doi.org/10.1016/c2015-0-05397-7 (date of access: 21.05.2024).

31 Tunga K. R. Experimental and Theoretical Assessment of PBGA Reliability in Conjunction with Field-Use Conditions : thesis. 2004. URL: http://hdl.handle.net/1853/5266 (date of access: 21.05.2024).

32 Westberg J. Faraday modulation spectroscopy : Theoretical description and experimental realization for detection of nitric oxide : doctoral thesis. 2013. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-68649 (date of access: 21.05.2024).

33 Żochowski M. Synchrony in biological and physical systems: An experimental and theoretical study. Warszawa : Polska Akademia Nauk, Instytut Biocybernetyki i Inżynierii Biomedycznej, 2000. 104 c.

34 api-gateway. .f5.com. URL: https://www.f5.com/glossary/api-gateway (дата звернення: 21.05.2024).

35 Що таке Kubernetes?. azure.microsoft.com. URL: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-kubernetes#watch-how-kubernetes-works (дата звернення: 21.05.2024).

36 Що таке Blackhole Routing?. www.akamai.com. URL: https://www.akamai.com/glossary/what-is-blackhole-routing (дата звернення: 21.05.2024).

37 Simadiputra V., Surantha N. Rasefiberry: Secure and efficient Raspberry-Pi based gateway for smarthome IoT architecture. Bulletin of Electrical Engineering and Informatics. 2021. Vol. 10, no. 2. P. 1035–1045. URL: https://doi.org/10.11591/eei.v10i2.2741 (date of access: 21.05.2024).

38 Huang W., Zhou J., Zhang D. On-the-Fly Fusion of Remotely-Sensed Big Data Using an Elastic Computing Paradigm with a Containerized Spark Engine on Kubernetes. Sensors. 2021. Vol. 21, no. 9. P. 2971. URL: https://doi.org/10.3390/s21092971 (date of access: 21.05.2024).

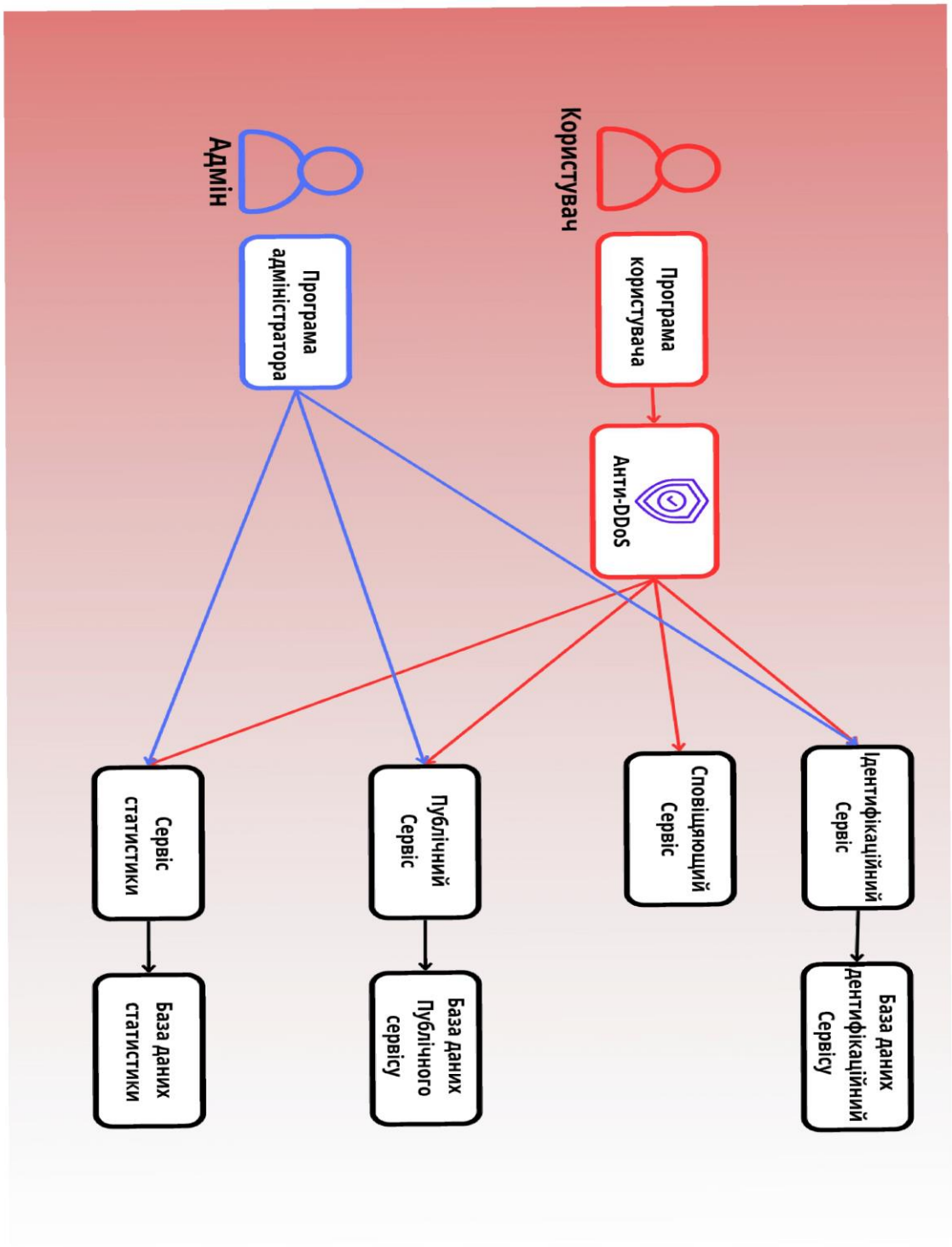
39 Experiences Modeling a OPC UA / DDS Gateway in AADL in the Context of Fog Computing / P. Denzler et al. ACM SIGAda Ada Letters. 2023. Vol. 43, no. 1. P. 58. URL: https://doi.org/10.1145/3631483.3631490 (date of access: 21.05.2024).

40 BHARDWAJ G., SHANKAR U. Load Balanced Fuzzy Control based Adaptive Gateway Discovery for Ubiquitous Internet Access in MANET. INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY. 2015. Vol. 14, no. 12. P. 6334–6342. URL: https://doi.org/10.24297/ijct.v14i12.1743 (date of access: 21.05.2024).

ДОДАТОК А

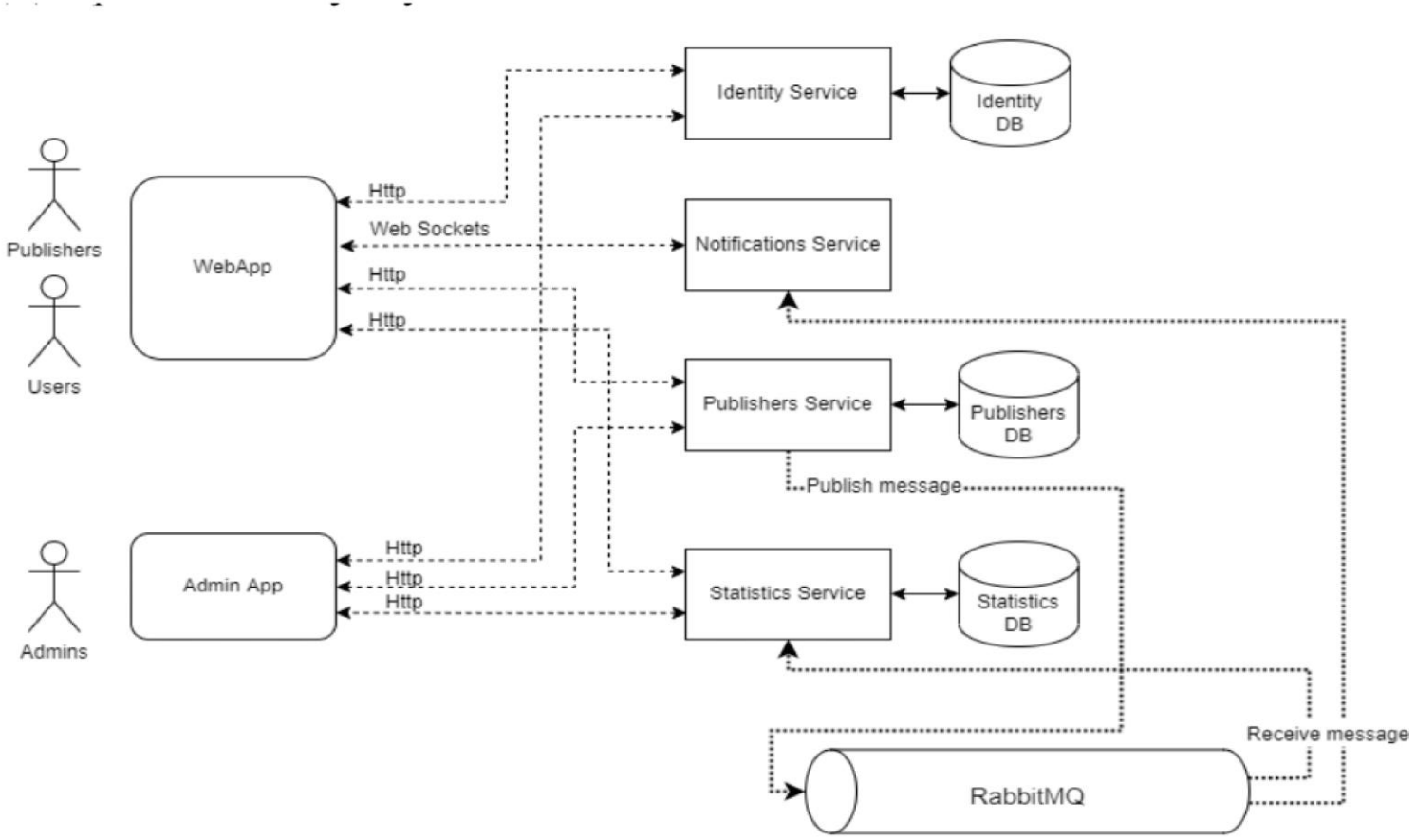
(Обов’язковий)

Копія Графічної частини

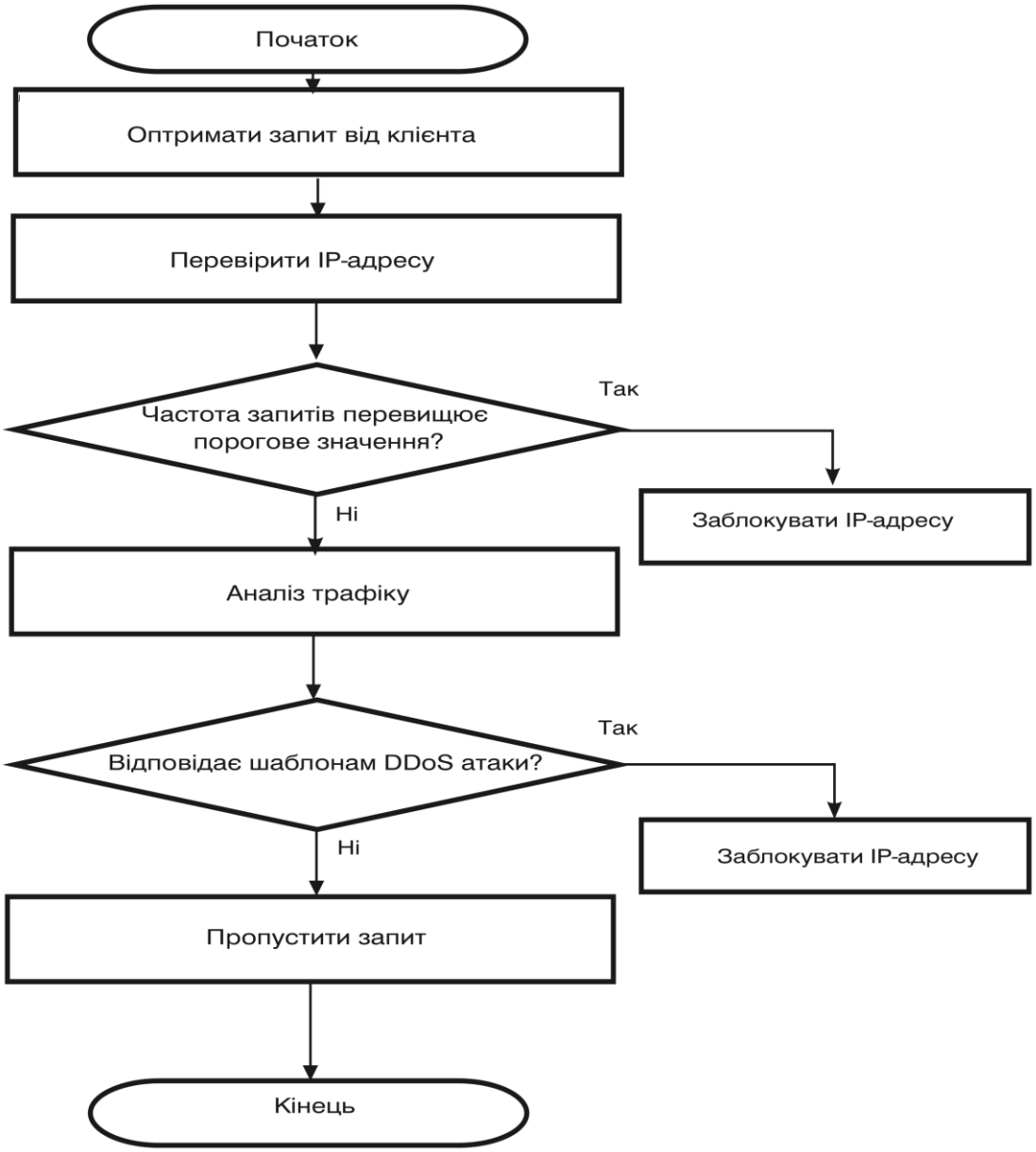


КРБКБ.200117.20.01.17 Е8

Зм	Арк.	№ докум.	Підпис	Дата	КРБКБ.200117.20.01.17 Е8			
Розроб.		Ткачук Т.С.			Система захисту мікросервісних застосунків від атаки на відмову	Літ	Маса	Масштаб
Перевір.		Стецюк М.В.				У		
Н.контр.		Мостовий С.В			Графічний вигляд програми	Аркуш	Аркушів	1
Т.контр						ХНУ, КБ-20-1		
Затверд.		Кльоц Ю.П.						



							КРБКБ.200117.20.01.17 Е8			
							Система захисту мікросервісних застосунків від атаки на відмову	Літ	Маса	Масштаб
								у		
							Графічний вигляд програми	Аркуш	Аркушів	1
Зм.	Арк.	№ докум.	Підпис	Дата				ХНУ, КБ-20-1		
Розроб.		Ткачук Т.С.								
Перевір.		Стецюк М.В.								
Н.контр.		Мостовий С.В								
Т.контр.										
Затверд.		Кльоц Ю.П.								



					КРБКБ. 200117.20.01.17 Е8	
Зм.	Арк.	№ докум.	Підпис	Дата	Система захисту мікросервісних застосунків від атаки на відмову Алгоритм роботи	Літера / Маса / Масштаб
Розроб.		Ткачук Т.С.				
Перевір.		Стецюк М.В.				Аркуш / Аркушів
Т.Контр.						
Н.Контр		Мостовий С.В.			ХНУ, КБ-20-1	
Затв.		Кльоц Ю.П.				

Завідувачу кафедри кібербезпеки
к.т.н., доц. Кльоцу Ю.П.
Ткачука Тимура Сергійовича
ПІБ здобувача вищої освіти

Студента ФІТ, 4 курсу, групи КБ-20-1

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у хмельницькому національному університеті» від 31.08.2023, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

17.06.2024
дата

підпис

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 1.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 11%**

	Документ		Сумарний збіг по Базі Даних	
ID: 131709 Назва: Система захисту мікросервісних застосунків від “атаки на відмову” Додано в БД: 2024-06-20 Автора: Ткачук Т.С. Керівники: Стецюк М.В. Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	100499	762	2199 (2%)	16 (2%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK
by Turnitin

Ім'я користувача:
Кафедра кібербезпеки

ID перевірки:
1016377632

Дата перевірки:
20.06.2024 12:02:11 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
20.06.2024 12:04:27 EEST

ID користувача:
100008300

Назва документа: **Ткачук_дипл антиплагіат (1)**

Кількість сторінок: **61** Кількість слів: **13744** Кількість символів: **107739** Розмір файлу: **715.71 KB** ID файлу: **1016186100**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

12.6%
Схожість

Найбільша схожість: **3.06%** з Інтернет-джерелом (https://origin-production.wikiwand.com/uk/DDoS)

12.2% Джерела з Інтернету	276	Сторінка 63
1% Джерела з Бібліотеки	73	Сторінка 64

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	35
Підозріле форматування	56 сторінок

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КІБЕРБЕЗПЕКИ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Розумна система на основі мікроплати для ведення відеонагляду

Автор: Ткачук Тимур Сергійович

Спеціальність: 125 – Кібербезпека

Освітня програма: кібербезпека

Науковий керівник: Стецюк Микола Васильович, др. філософії

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Оригінальність тексту роботи за результатами перевірки системою Unichek складає 86.4%, оригінальність тексту роботи за результатами перевірки системою Anti-Plagiarism v-15.257 складає 99%.

Згідно з Положенням про систему забезпечення академічної доброчесності у ХНУ (https://khmnu.edu.ua/wp-content/uploads/normatyvni-dokumenty/polozhennya/pro-systemu-zabezpechennya-akademichnoyi-dobrochesnosti.pdf, Додаток В) кваліфікаційна робота, виконана за , освітньо-професійною програмою, кількісні показники рівня унікальності тексту у відсотках до загального обсягу матеріалу в якій складає 75-100 %, визнається роботою з високою унікальністю тексту: «Текст вважається унікальним і не потребує додаткових дій щодо запобігання неправомірним запозиченням».

Керівник роботи _____ Микола СТЕЦЮК

Завідувач кафедри кібербезпеки _____ Юрій КЛЬОЦ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітнього ступеня «бакалавр»

Студент Ткачук Тимур Сергійович

Тема Система захисту мікросервісних застосунків від атаки на відмову

Спеціальність 125 – Кібербезпека

Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «бакалавр»:

кількість листів креслень 3; кількість сторінок записки 72.

1. Короткий зміст роботи та прийнятих рішень Кваліфікаційна робота на тему "Система захисту мікросервісних застосунків від атаки на відмову" спрямована на розробку ефективної системи захисту, здатної проактивно виявляти та нейтралізувати потенційні кіберзагрози. У роботі досліджено особливості мікросервісної архітектури, типи DDoS-атак та існуючі методи захисту. Розроблено модель загроз, спроектовано та реалізовано систему захисту, проведено її апробацію та оцінку ефективності

2. Висновок про відповідність кваліфікаційної роботи завданню У кваліфікаційній роботі було виконано поставлене завдання як у теоретичній, так і в практичній частині.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі роботи наведена загальна характеристика задачі, визначені об'єкт, предмет та методи дослідження, а також сформульована мета. Зазначені задачі, що потрібно виконати для досягнення поставленої мети, проведений аналіз досліджуваної проблеми та обгрунтований підхід до її вирішення. У першому розділі розглядаються об'єкти захисту інформації та системи контролю доступу. Проведено апробацію розробленої системи та оцінено її ефективність у реальних умовах. У роботі використані сучасні методи шифрування даних, веб-брандмауери (WAF), методи моніторингу мережевого трафіку та ліцензування

4. Позитивні сторони роботи Кваліфікаційна робота має практичну цінність. Вона полягає у розробці системи захисту мікросервісних застосунків від атаки на відмову, що забезпечує захист інформації та спрощує користування обладнанням. Завдяки цьому підприємство є захищеним від атаки на відмову.

5. Негативні сторони роботи В системі не передбачено резервне живлення на випадок зникнення електроенергії, що є надзвичайно актуальним в сучасних умовах, тому за відсутності електроенергії не буде працювати захист від атаки на відмову, стане потреба механічного режиму, що знижує ефективність захисту від вторгнень.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення кваліфікаційної роботи відповідає темі роботи та виконане з дотриманням стандартів. В цілому, графічне оформлення є якісним, а пояснювальна записка відповідає нормам оформлення.

7. Відгук про роботу в цілому Кваліфікаційна робота заслуговує позитивної оцінки, оскільки весь матеріал роботи є структурованим, чітким та послідовним. Усі розділи роботи мають логічну послідовність, що сприяє зрозумінню викладеного матеріалу в рамках теми роботи. Графічний матеріал допомагає наочно продемонструвати доцільність та ефективність прийнятих рішень для досягнення мети.

8. Інші зауваження В переліку використаних джерел наявні посилання на популярні ресурси, такі, як Вікіпендія, які не рекомендовано використовувати при написанні кваліфікаціних робіт.

9. Оцінка кваліфікаційної роботи Враховуючи всі позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінки «добре».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Підченко Сергій Костянтинович,
завідувач кафедри ТМІТ, доктор технічних наук, професор

« 19 » травня 2024. _____ (підпис)