Вінницький національний технічний університет
Факультет інформаційних технологій та копм'ютерної інженерії
Кафедра обчислювальної техніки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської дипломної роботи

на тему: Технологія виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів

Виконав: студент 4 курсу, групи 2КІ-18Б
Спеціальності 123 Комп'ютерна інженерія

Нич В. О.
(прізвище та ініціали)

Керівник: доц. кафедри ОТ
Савицька Л. А.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент: к.т.н., доц. Кафедри МБІС
Шиян А.А.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

м. Вінниця – 2022 рік

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітній рівень бакалавр

Спеціальність 123 — «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач кафедри

обчислювальної техніки

проф. Азарову О.Д.

«08» 02 2022 р.

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Ничу Вадиму Олеговичу

1 Тема роботи: «Технологія виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів»

Керівник роботи к.т.н., доц. каф.ОТ Савицька Людмила Анатоліївна

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "24" 03 2022 року №66

2 Строк подання студентом проекту (роботи) 21.06.2022р

3 Вихідні дані до проекту (роботи) — ключ для AWS – CLI, акаунт для роботи із AWS, Teraform, biling акаунт AWS.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): вступ, аналіз предметної області, розробка методу.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): алгоритм по визначенню початку антаки і виділенню шкідливого трафіку

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
	Савицька Л.А.		

7. Дата видачі завдання 11.02.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання комплексної бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	14.03.22	вик.
2	Аналіз предметної області	15.03-18.03.22	вик.
3	Вплив DoS атак на хмарні середовища та мережі передачі даних	21.03-24.03.22	вик.
4	Визначення методів та інструментаріїв захисту від DDoS атак	25.03-30.03.22	вик.
5	Принцип роботи механізму захисту від DoS атак	31.03-04.04.22	вик.
6	Вибір технологій для створення клієнтської частини застосунку	05.04-.11.04.22	вик.
7	Підготовка матеріалів та опис розробки інформаційної системи	23.05-26.05.22	вик.
8	Аналіз виконання роботи, висновки, додатки	27.05-31.05.22	вик.
9	Перевірка якості виконання бакалаврського проекту та усунення недоліків	01.06 -.11.06.22	вик.

Студент _____ Шиз В.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Савицька Л.А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Нич В.О. Технологія виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів. Бакалаврська робота зі спеціальності 123 — Комп'ютерна Інженерія, Вінниця: ВНТУ, 2022. Пояснювальна записка містить __ сторінки, __ рисунків та __ посилань.

Бакалаврська робота присвячена технології виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів. Проаналізовано існуючі типи та види мереж, їх класифікація та топології. Виконано аналіз вразливостей сучасних мереж до атак, визначено інструментарій для боротьби із атаками, а саме їх виявлення попередження та запобігання

Ключові слова: Teraform, AWS, DDoS, хмарні сервіси, захист від атак.

ABSTRACT

Nych V. O. Technology for detecting and preventing DDoS attacks implemented using cloud services. Bachelor's thesis in specialty 123 — Computer Engineering, Vinnytsia: VNTU, 2022. Explanatory note contains __ pages, __ figures and __ links.

The bachelor's thesis is devoted to the technology of detecting and preventing DDoS attacks implemented using cloud services. Existing types and kinds of networks, their classification and topologies are analyzed. The analysis of vulnerabilities of modern networks to attacks is carried out, the tools for struggle against attacks are defined, and their detection and prevention and prevention is defined.

Keywords: Teraform, AWS, DDoS, cloud services, attack protection.

ЗМІСТ

ВСТУП

DDoS атаки являються чи не одними з найпоширеніших типів атак, спрямованих на порушення функцій обслуговування мереж передачі даних. DDoS-атаки генеруються за рахунок надмірно великої кількості небажаного мережевого трафіку (пакетів данних) або шляхом примушування обчислювальних ресурсів до опрацювання невірних процесів та зберігання даних.

Об'єктом дослідження є процеси які відбуваються в комп'ютерних системах під час атаки на комп'ютерні мережі.

Предметом дослідження є методи присічення DDoS атак, і відсіювання зловмисного трафіку.

Метою даної роботи є визначення найліпшого механізму виявлення та запобігання DDoS атакам, шляхом аналізу існуючих механізмів та інструментаріїв.

Аби досягнути цілі необхідно розв'язати такі **задачі**:

— проаналізувати які є види комп'ютерних мереж;

— визначити класифікації та топології комп'ютерних мереж;

— виконати аналіз вразливостей сучасних комп'ютерних мереж до DDoS атак;

— визначити інструментарій для боротьби із DDoS атаками, а саме їх виявлення та запобгання.

Практичне значення бакалаврської дипломної роботи складається у дослідженні методів і алгоритмів захисту комп'ютерних мереж від DDoS-атак, що дають можливість проводити активну протидію на боці атакованого ресурсу. Це підтверджується дослідженням та подальшою реалізацією хмарних сервісів для присічення DDoS-атак, та подальше блокування шкідливих звернень на різних рівнях інформаційної системи.

1 ОГЛЯД НА КОМП’ЮТЕРНІ МЕРЕЖІ

1.1 Загальні поняття

Комп'ю́терна мере́жа — система зв'язку між комп'ютерами. У ширшому розумінні комп'ютерна мережа — це система зв'язку через кабель або через бездротове середовище, самі комп'ютери різного функціоналу і мережеве обладнання. Для передачі інформації використовуються різні фізичні явища, найчастіше — різні види електричних сигналів чи електромагнітного випромінювання. Комп’ютерні мережі можуть мати телефонні лінії та спеціальні мережеві кабелі: коаксіальні кабелі, виту пару, волоконно-оптичні кабелі, радіохвилі, оптичні сигнали.

Принципи комунікації у комп’ютерних мережах поділяють на наступні на багатоадресна передача — це особлива форма широкомовлення, при якій копії пакетів надсилаються певній підмножині одержувачів.

Обміну - процес з'єднання користувачів мережі зв'язку через транзитний вузол. Комунікаційні мережі повинні гарантувати, що їхні користувачі підключені один до одного. Зазвичай у мережі загального доступу неможливо надати кожній парі користувачів власну фізичну лінію зв'язку, яку вони можуть монополізувати «власною» і використовувати в будь-який момент. Тому в мережі завжди використовується певний метод перемикання користувачів, який забезпечує розділення існуючих фізичних каналів між кількома сеансами зв’язку та між користувачами мережі.

Маршрутизації — це процес визначення шляху доставки інформації в комунікаційній мережі. Маршрути можна вказати в режимі керування (статична маршрутизація) або використовувати алгоритми маршрутизації для обчислення маршрутів на основі топології мережі та інформації про стан, отриманої за допомогою протоколів маршрутизації (динамічна маршрутизація). Маршрутизація в комп'ютерних мережах виконується спеціальним програмно-апаратним забезпеченням - маршрутизаторами, у простих конфігураціях може виконуватися окремо від комп'ютерів загального призначення.

Моніторингу мережі — робота системи, яка постійно виконує спостереження за комп'ютерною мережею у пошуках сповільнених або несправних систем і яка при

виявленні помилок повідомляє про них мережевого адміністратора через пошту, пейджер або інші засоби сповіщення.

Існує кілька основних технологій які використовується у сучасних комп'ютерних мережах.

Технологія ADSL (Асиметрична цифрова абонентська лінія) відноситься до високошвидкісної технології передачі даних, яка називається технологією цифрової абонентської лінії (DSL), широко відома як xDSL.

Технологія VDSL (Very high bitrate Digital Subscriber Line) — це найвища швидкісна технологія xDSL, від 1,5 до 33 Мбіт/с, яка використовує пару телефонних ліній витої пари. Технологію VDSL можна вважати економічно ефективною альтернативою прокладці волоконно-оптичних кабелів для кінцевих користувачів. Однак найбільша відстань передачі даних цієї технології становить від 300 до 1700 метрів.

Існує також симетрична версія VDSL (до 33 Мбіт/с в кожному напрямку). Технологія VDSL може використовуватися для тих же цілей, що й ADSL (використання паралельних телефонних ліній для високошвидкісної передачі даних і традиційного телефонного зв'язку).

Технологія IDSL (ISDN Digital Subscriber Line) забезпечує повнодуплексну передачу даних зі швидкістю до 144 Кбіт/с. На відміну від ADSL, функціональність IDSL обмежена передачею даних. Технологія HomePNA 1.0 (1 Мбіт/с) використовує метод доступу IEEE 802.3 CSMA/CD (Ethernet). Пропускна здатність сигналу знаходиться в діапазоні від 5,5 МГц до 9,5 МГц, що дозволяє не впливати на роботу ADSL і VDSL (пристрою і телефону). HomePNA використовує багаторазове кодування одного бітового імпульсу. У кожному мережевому інтерфейсі схема приймача вміщує різні рівні перешкод, які можуть бути присутніми на лінії. Крім того, вузол-відправник може змінювати рівень сигналу. Термінали-передавач і прийом постійно контролюють умови сигналу і відповідно до цих умов коригують свої параметри. Саме ця адаптивність значно знижує вимоги до середовища передачі. По суті, технологія

HPNA - це Gigabit Ethernet, що працює через телефонні лінії. Це дозволяє використовувати широкий спектр Ethernet-сумісних програм, драйверів, програм та обладнання.

Технологія PON (Пасивна оптична мережа) Пасивна оптична мережа або пасивна розподілена оптична мережа — це волоконно-оптична кабельна система з топологією дерева з використанням пасивних оптичних розгалужувачів 1:n. Між центральним вузлом і віддаленими вузлами користувача створюється повністю пасивна оптична мережа з топологією дерева. Середній вузол дерева — це пасивний оптичний розгалужувач, який розподіляє загальний сигнал джерела багатьом приймачам користувачів. Вихідна передача від абонентів відбувається по зворотному каналу з використанням протоколу множинного доступу (TDMA) з тимчасовим розділенням.

Технологія MPLS (Multi-Protocol Label Switching) - багатопротокольне перемикання на основі міток, розроблене комітетом IETF і є результатом симбіозу різних власних механізмів, таких як IP-коммутація (Ipsilon Networks), перемикання міток (Cisco Systems), Aris (IBM) і стільниковий комутаційний маршрутизатор (Toshiba). Він заснований на принципі відображення мережевих адрес на спеціальні мітки, які можна використовувати для маршрутизації пакетів. Архітектура MPLS об'єднує найкращі елементи всіх згаданих механізмів брендингу і стала стандартом Інтернету завдяки зусиллям компаній, зацікавлених у швидкому просуванні технології на ринку.

1.2 Топології комп’ютерних мереж

Комунікаційні мережі можуть специфікуватись також за топологією з'єднання пристроїв. Базовими технологіями є:

— Шина;
— Кільце;
— Зірка;
— Змішана.

При використанні шинної топології (рисунок 1.1) пристрої в мережі з’єднуються таким чином, що дані, що передаються між будь-якими двома пристроями, доступні всім іншим пристроям мережі. Визначте пристрій і місце, де було здійснено передачу, проаналізувавши дані, передані всіма пристроями в мережі. Надзвичайно проста, але й найменш ефективна топологія. Прикладом мережі шинної топології є добре відомий стандарт Ethernet.

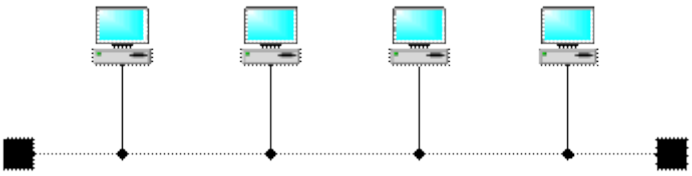

Рисунок 1.1 — Топологія «Шина»

У мережі з кільцевою топологією (рисунок 1.2) дані переміщуються в одному напрямку. Деякі реалізації кільцевих мереж є дещо ефективнішими, ніж шинні мережі (наприклад, ранні реалізації стандарту Token Ring були приблизно в 1,5 рази ефективнішими, ніж Ethernet), але, як правило, це топології, близькі до класу. В обох типах дані можуть передаватися лише між парою пристроїв одночасно.

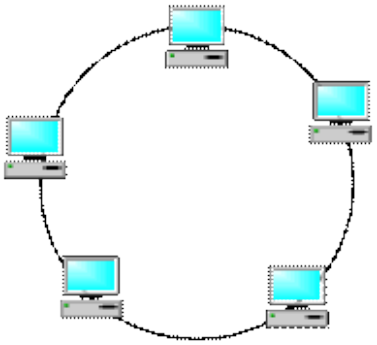

Рисунок 1.2 — Топологія «Кільце»

Зіркоподібні мережі (рис.1.3) мають у складі обов'язковий службовий елемент — комутатор. Комутатор не бере участі у комунікаціях безпосередньо, але забезпечує

зв'язок між будь-якими двома пристроями, які залежні від комунікації. Те на скільки складним і ефективним є комутатор — значною мірою визначає ефективність всієї мережі.

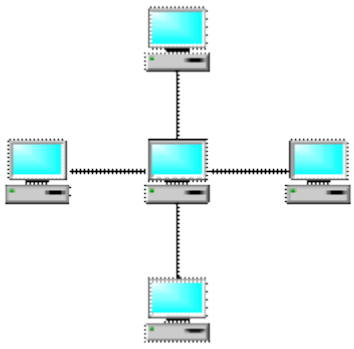

Рисунок 1.3 — Топологія «Зірка»

Особливістю цих типів є те, що вони зазвичай можуть використовуватися в обмежених областях, оскільки збільшення довжини каналу зв’язку призводить до того, що вони стають менш ефективними та дорожчими. Наприклад, кілька робочих груп локальної мережі, створених топологією шини, також можуть бути з’єднані топологією «зірка», тоді як найпоширенішим з’єднанням з Інтернетом є стільникова топологія.

Топологія мережі визначає не тільки фізичне розташування комп’ютерів, але, що найбільш важливіше, характер зв’язків між комп’ютерами, особливості поширення сигналів по всій мережі. Характер зв’язків визначає ступінь відмовостійкості мережі, необхідну складність мережної апаратури, найбільш підходящий метод керування обміном, можливі типи середовищ передачі (каналів зв’язку), припустимий розмір мережі (довжина ліній зв’язку й кількість абонентів), необхідність електричного узгодження й багато чого іншого.

Коли в літературі згадується про топологію мережі, то можуть мати на увазі чотири зовсім різних поняття, що ставляться до різних рівнів мережної архітектури:

— Фізична топологія (а саме схема розміщення комп'ютерів і прокладки кабелю). У цьому змісті, наприклад, пасивна зірка нічим не буде відрізнятися від активної зірки, тому її часто називають просто «зіркою».

— Логічна топологія (тобто структура зв'язків, характер розповсюдження сигналів по мережі). Це, напевно, найточніше визначення цієї топології.

— Топологія керування обміном (тобто принцип і порядок передачі права на захвачення мережі між окремими комп'ютерами).

— Інформаційна топологія (тобто куда саме йде інформація, передана по мережі.).

Наприклад, мережа з фізичною й логічною топологією «шина» може одночасно передавати всю інформацію через один виділений комп'ютер (бути в цьому змісті зіркою) і метод керування - використовувати естафетну передачу права захвата мережі (тобто бути в цьому змісті кільцем)

1.3 Хмарні середовища обробки данних

Хмарні обчислення - одна з передових технологій, яка дозволяє отримати величезний обсяг зберігання даних та надання послуг, в той же час вони пропонують високу продуктивність і низьку вартість. Хмарні обчислення, відомі як обчислювальна модель, яка керує діапазоном обчислювальних ресурсів з можливістю гнучкого налаштування. На основі моделей розгортання хмарні обчислювальні середовища можна класифікувати як середовища з публічним доступом, середовища з обмеженим доступом та гібридні середовища.

Хмарні обчислювальні середовища стали легкими мішенями для зловмисників для заволодіння інформацією користувачів. DoS-атаки є найбільшою загрозою для цієї галузі обчислень і найбільшою перешкодою для широкого впровадження хмарних обчислювальних середовищ. DOSатаки вважаються одними з найбільш значущих проблем, що стоять перед зростанням популярності хмарних середовищ. Кілька причин роблять DoSатаки серйозною загрозою: наприклад, така атака є руйнівною в

цьому середовищі, і її легко здійснити. Крім того, зловмисники використовують підроблені IP-адреси, таким чином, відстежувати джерело нападу стає важко.

Зазвичай мережа зловмисника, з якої відбувається DDoS атака, має три рівні (рис 1.4). Назва такої мережі «кластер DDoS». Верхній рівень кластера складається с декількох комп’ютерів, з яких відбувається початок атаки та надалі іде координація дій інших учасників. Другий рівень – складається з десятків комп'ютери, які вже надають сигнали про атаку на наступний рівень кластера. На третьому рівні — зазвичай DDoS-зловмисники покладаються на ботнети — колекції мережі інфікованих зловмисними програмами, які централізовано контролюються. Ці заражені кінцеві точки, як правило, є комп'ютерами та серверами, але все частіше є IoT та мобільними пристроями. Зловмисники збирають ці системи шляхом виявлення вразливих систем, якими вони можуть потенційно заразитися за допомогою фішинг-атак, зловмисних атак та інших методів масового зараження. Все частіше зловмисники також орендують цих ботнетів у тих, хто їх побудував.

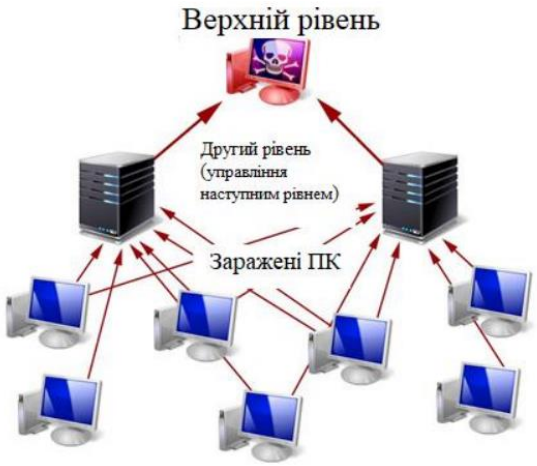

Рис. 1.4 — Структура ботнет мережі

Підсумовуючи викладену інформацію у розділі розглянуто загальні принципи роботи комп’ютерних мереж, їх топології, види, та схематичне відображення задля ліпшого розуміння. Що допомогло в подальшому аналізі слабких вузлів та вразливостей. Як підсумок розділу – виконано класифікацію хмарних обчислювальних середовищ та вразлиивих точок (вузлів) перед DoS атаками.

2 АНАЛІЗ ВПЛИВУ ТА МЕТОДІВ ЗАХИСТУ ВІД DDOS АТАК КОМП'ЮТЕРНИХ МЕРЕЖАХ

2.1 Вплив DoS атак на хмарні середовища та мережі передачі даних

Зловмисники можуть запускати різні DoS атаки, які можуть впливати на приватні мережі, на загальнодоступні мережі, або на всі типи мереж. Ці атаки орієнтовані на такі ресурси, як пропускна здатність мережі, пам'ять та процесор, а також такі, що орієнтовані на додатки, такі як служба баз даних та веб-додатки. DoS-атаки генеруються за рахунок надмірно великої кількості небажаного мережевого трафіку або шляхом примушування обчислювальних ресурсів до опрацювання фейкових процесів та зберігання даних. Можуть трапитися три види впливу:

Direct denial of service - операційна система хмарного обчислювального середовища розглядає додаткове навантаження на конкретну службу як спосіб працювати проти зловмисника.

Indirect denial of service - атака направлена на обчислювальну потужність. Негативний вплив прямої атаки набувається шляхом перенасичення нерелеватною інформацією однієї служби, що впливає на інші служби, що працюють на тих же апаратних засобах, вони можуть страждати від навантаження, спричиненого атакою на іншу службу. Тому у випадку, коли служба працює на одному апаратному засобі з іншими, це вплине на їх загальну доступність.

Основним впливом атаки за допомогою перенасичення нерелеватною інформацією (flooding) на хмарні середовища є стягнення плати з клієнтів за використання ресурсів. Це означає, що немає обмежень у використанні обчислювальної потужності. Хмарні обчислення мають деякі властивості, які впливають на захист від DoS атак. Методи захисту узагальнені у трьох поняттях: виявлення, ідентифікація та фільтрація. По-перше, хмарні обчислювальні середовища забезпечені фізичними серверами, які керують мережевими та обчислювальними ресурсами замість користувачів. По-друге, з точки зору захисника, виділення ресурсів та міграція віртуальної машини є новими джерелами змін в топології мережі, і процеси швидко розвиваються. Отже, захист від DoS атак повинен мати можливість налаштовуватися на динамічну мережу,

що має часті зміни в топології, зберігаючи високий показник швидкості виявлення та здатність швидко реагувати. По-третє, хмарні обчислення дозволяють усім користувачам ділитися однаковою мережевою інфраструктурою, що викликає потребу надійного розділення мережі. При традиційному захисті від DoS атак ця вимога не враховувалася. Операції виявлення та захисту від DoS атак не повинні впливати на роботу мережі і користувачів, але і користувачі не повинні впливати на ці операції.

Операційна модель хмарних середовищ представила інші завдання для DDoS атак, такі як динамічна топологія мережі та розширений периметр оборони. Для ефективного вирішення цих проблем адміністратор хмарного середовища повинен мати можливість легко доручити управління мережі користувачам хмарного середовища та швидко налаштувати управління на основі змін топології мережі. SDN забезпечує розширену логіку виявлення та легко реалізує результативні операції, а також забезпечує ефективність обробки пакетів. Тим часом, затримки в мережі та потоці трафіку, що виникають у зв'язку з керуванням мережею та змінами в топології, наприклад, зв'язок між схемами захисту від DDoS та комутаторами, може генерувати нову хвилю атаки та призвести до відмови в одній точці мережі. Щоб уникнути нової вразливості в безпеці SDN, слід врахувати втрати на обчислення та комунікаційні витрати при розробці рішення захисту від DDoS-атак. Підсумувавши можна сказати, що SDN може мати перевагу в захисті від DDoS-атаки на хмарні середовища та мережі передачі даних, коли всі комунікаційні та обчислювальні процеси будуть вивірені та оптимізовані.

2.2 Класифікація та структура DDoS атак

DDoS атака здатна знищити великі веб-сервіси, для яких зазвичай потрібні тисячі скомпрометованих машин. Структура такої розподіленої атаки є складною, та вказана на рисунку 2.1. Зловмисники завжди намагаються зробити одну чи обидві наступні речі:

Перша мета атаки - відключити з'єднання для законного користувача через споживання пропускної здатності мережевих ресурсів або ємності маршрутизатора (атаки мережевого і транспортного рівнів методом перенасичення)

Друга мета атаки - відключити послуги для законного користувача через вичерпання ресурсів сервера (атаки перенасичення на рівні додатків).

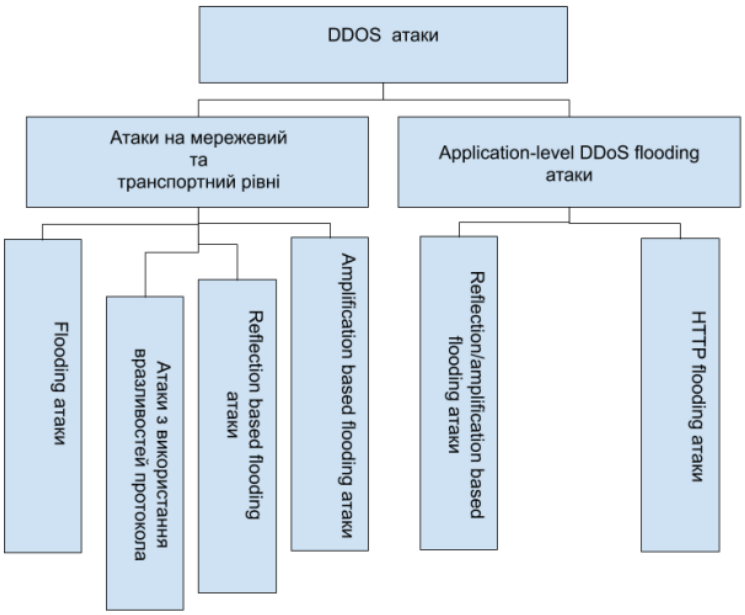

Рисунок 2.1 – Структура DDoS атаки

DDoS-атаки на мережевий та транспортний рівні запускаються через UDP, TCP та DNS. У цій категорії атаки класифікуються на чотири підкатегорії

Reflection-based flooding атаки: щоб вичерпати ресурси жертви, зловмисники надсилають сфальсифіковані запити, а не прямі запити.

Атаки з використанням вразливостей протокола: з метою надмірного споживання ресурсів атакованої мережі, зловмисники впроваджують помилки в мережеві протоколи атакованої мережі.

Amplification-based flooding атаки: з метою посилення трафіку до атакованої мережі, зловмисники генерують кожне повідомлення, яке вони отримують.

Flooding атаки: для розриву зв'язку з законним користувачем зловмисники вичерпують пропускну здатність мережі для атакованої мережі.

Оскільки основною функцією цих атак є зрив роботи служб, вони споживають меншу пропускну здатність. Як правило, DDoS-атаки на рівні додатків мають однаковий вплив на сервіси, оскільки вони орієнтовані на конкретні характеристики в таких додатках, як DNS та HTTP.

HTTP flooding атаки: використовуються щоб відключити веб-сервер, що є ціллю атаки, зловмисники надсилають величезну кількість HTTP-запитів.

Reflection/amplification based flooding атаки: Зловмисники використовують ті самі методи на мережевому і транспортному рівнях з метою збоїв у системі трафіку.

2.3 Визначення методів та інструментаріїв захисту від DDoS атак

У даному розділі роботи визначаються різні методи пом'якшення DDoS атак та актуальні технічні проблеми. "Denial-of-Service атаки, можуть завдати серйозної шкоди інфраструктурі жертви нападу".

Оскільки жоден законний користувач не може користуватися послугою, це призводить до величезних фінансових втрат та втрат продуктивності. Таким чином, необхідно створити механізми для пом’якшення впливу цих атак на сервіси мережі. У літературі представлено кілька механізмів пом'якшення DDoS-атак.

Виділимо чотири категорії механізмів захисту від DDoS-атак на мережевому та транспортному рівні на основі класифікацій:

Source-based механізми - розташовуються близько до джерел атаки, щоб зменшити шкоду для користувачів мережі від DDoS атак (фільтрація вхідного та вихідного трафіку).

Network-based механізми - механізми розподіляються всередині мереж на маршрутизаторах автономних систем (Autonomous Systems - ASs) (наприклад, фільтрування пакетів на основі маршруту).

Destination-based механізми - механізми виявлення та реагування на напад застосовуються в пункті призначення (тобто в атакованому вузлі).

Hybrid (Distributed) механізми - розгортаються на кількох сайтах, включаючи джерело та місце призначення, наприклад, активна фільтрація інтернет-трафіку AITF.

На рівні додатків виділено дві категорії механізмів захисту від DDoS атаки на основі класифікацій:

Destination-based (server-side) механізми: використовують статистичні методи в DDoS-Shield як основний механізм захисту для виявлення характеристик HTTP сессій.

Public Turing test to tell Computers and Humans Apart (CAPTCHA). В механізми захисту класифікуються на три основні типи:

Proactive defense механізми - спочатку запропонована методика не починає боротьбу з DDoS атакою, а підготовлює базу правил потоку, які застосовуються безпосередньо під час атаки на мережу. Це означає, що атакований потерпілий має доступ до ресурсів, які можуть протистояти такій атаці та функціонувати. У наші дні ця інфраструктура може бути хмарною, коли ресурси виділяються лише коли це потрібно, і ці ресурси можуть бути основою стратегії виживання, коли немає механізму захисту мережі.

Reactive defense механізми - полегшують наслідки або перешкоджають DDoS атаці. Вони розпізнають наступну атаку за певними зразками поведінки мережі. Наприклад, система виявлення вторгнень (IDS) діє як рухомий екран (movement screen) та аналізатор. Це означає, що даний механізм захисту від DDoS-атаки хороший лише як розподілений IDS.

Post attack аналіз: основна мета аналізу після атаки - дослідити атаку та використовувати отримані дані для виявлення нападника.

SDN вбачає нову архітектуру мережі для хмарних обчислювальних середовищ та мереж передачі даних, яка має багато переваг для виявлення DDoS-атак:

Площини управління та даних відокремлені одна від одної. У SDN мережеві пристрої не мають функціоналу управління, і вони стали простими вузлами переадресації.

Рішення щодо переадресації пакетів базуються на основі потоку трафіку. Набір значень пакетів та набір дій визначаються як потік. Потік у SDN - це послідовність пакетів між джерелом та пунктом призначення. Програмування потоку забезпечує безпрецедентну гнучкість. Швидке реагування на DDoS-атаки відбувається з динамічним оновленням правил переадресації. Нова або оновлена політика безпеки, заснована на аналізі трафіку, може бути розгорнута по всій мережі у вигляді правил потоку, щоб блокувати трафік атаки без зволікань.

Передача логіки контролера зовнішньому об’єкту. Призначення контролера схоже на традиційну операційну систему, яка забезпечує основні ресурси та абстракції відповідно до логіки контролера для допомоги програмуванню пристроїв переадресації на основі абстрактного виду мережі та логічної централізації.

Основна особливість програмованої SDN мережі в тому, що вона налаштовується програмами, що працюють на контролері та взаємодіють з іншими мережевими пристроями.

SDN контролер будує послідовну політику безпеки завдяки даним системи моніторингу та аналізу моделей трафіку на предмет можливих загроз безпеці мережі. Інформація, отримана під час аутентифікації (наприклад, через RADIUS сервер) та реєстрації користувачів, дає можливість організувати централізоване керування SDN для забезпечення динамічного блокування скомпрометованих хостів та аутентифікації легітимних хостів.

Аналіз трафіку на основі програмного забезпечення дозволяє знаходити нові методи боротьби, оскільки для покращення можливостей комутатора можливо використовувати всі типи інтелектуальних алгоритмів, баз даних та будь-яких інших програмних засобів. Підозрілий трафік може бути спрямований на IPS для проведення Deep Packet Inspection, механізми захисту наведені у таблиці 1.

Таблиця 1 рішення для захисту

Тип атаки	Методи атаки	Рішення
Атака на рівень додатків	Application	FortNOX
	Northbound API attack	
Атака на контрольований рівень	Атака на контролер	FortNOX AVANT-GUARD TSL
	Northbound API attack	
Атака на рівень інфраструктури	Атака на комутатор	FLOOD-Guard AVANT-GUARD TSL
	Southbound API attack	

Таблиця 2 Механізми захисту мереж згідно SDN

Механізм	Опис	Відмінності	Застосування
Sourcebased механізми	Дозволяє SDN контролерам ідентифікувати трафік атаки, фільтрувати скомпрометовані пакети або виявляти IP-адресу джерела атаки	Аналіз трафіку	Базова станція під контролем Openflow використовує аналіз в режимі реального часу для виявлення зловмисних програм
		Програмованість	Програмованість домашнього роутера з використання OpenFlow сумісних пристроїв
Networkbased механізми	Виявляють DoS атаки в потоці трафіку	Аналіз трафіку	Ідентифікують зловмисний трафік за допомогою статистичних даних
		Програмованість	Фреймворк безпеки FRESCO

Продовження таблиці 2

Destinationbased механізми	Використовує зворотнє IP трасування для дослідження джерела та шляху розповсюдження атак	Динамічне оновлення правил безпеки	Метод покрокового розподілу згідно вибіркових потоків для зворотнього IP трасування
			NetSight -платформа яка використовує додатки для відновлення історії пакетів

Оскільки можливі DDoS-атаки на рівень додатків можуть бути реалізовані через атакуючу програму або northbound API головним завданням є виявлення та узгодження конфліктних правил потоку, які накладаються через динамічні програми OpenFlow. FortNox пропонує аутентифікацію на основі ролі та застосовує обмеження безпеки для захищеної авторизації, забезпечує цілісність процесу для NOX OpenFlow контролера для кожної Open-Flow програми. OpenFlow пропонує підтримку шифрованого захисту безпеки транспортного рівня (TLS) та сертифікат обміну сертифікатами між контролером та комутаторами, а також можливість використовувати декількох моделей з декількома органами сертифікації. Крім того, забезпечення взаємодії з шифруванням за допомогою реплік контролера може бути корисним для отримання корректного повідомлення від контролера. Крім того, для забезпечення безпеки між площиною даних та пристроями площини управління можуть використовуватись динамічні та автоматизовані механізми пристроїв.

Алгоритм захисту IP-фільтрації є практичним рішенням для захисту від розподіленої атаки від відмови в обслуговуванні (DDoS) на основі фільтрації IP джерела. При такому підході граничні маршрутизатори зберігають історію всіх дійсних IP-адрес, які раніше з’являлися в мережі. Ця історія використовується, щоб вирішити, чи приймати пакети з певної IP-адреси, коли граничний маршрутизатор пере-

вантажений. На відміну від інших алгоритмів захисту від DDoS-атак, цей метод підходить для розширених розподілених DDoS-атак, тобто з великої кількості джерел. Обгрунтування алгоритму полягає в тому, що атаки трафіку DDoS намагаються використовувати підроблені випадкові адреси джерел, щоб замаскувати їх справжні адреси, що дає змогу захиститися від атак DDoS. Тому все ще необхідно розрізняти законні IP-адреси та випадкові кримінальні IP-адреси. Алгоритм використовує механізм, який називається фільтрацією IP на основі історії (HIF), де граничні маршрутизатори отримують пакети на основі попередньо створеної бази даних IP-адрес. База даних IP-адрес має бути заснована на попередній історії маршрутизатора. Існує кілька евристик, щоб зробити базу даних IP-адрес точною та надійною. Програма складається з трьох частин. Він дуже надійний проти розподілених атак, таких як відмова в обслуговуванні. Цей механізм не вимагає глобальної співпраці з іншими мережевими ресурсами. Технологію можна застосувати до багатьох типів трафіку і не вимагає спеціальної ручної настройки.

В другому розділі були розглянуті основні принципи, за якими зловмисники за допомогою DDoS атак можуть впливати на побудовані на базі SDN мережі передачі даних та хмарні обчислювальні середовища. Були розглянуті DDoS атаки на мережевий та транспортний рівні, а також на рівень додатків, що допомогло знайти вразливі місця SDN мереж. На основі цього були знайдені переваги SDN мереж в захисті від DDoS атак а також розглянуті існуючі методи захисту. Були розглянуті базові принципи роботи існуючих рішеннь захисту (такі як TLS, AVANT-GUARD та інші), що допомогло у подальшому аналізі вразливостей SDN мереж та аналізі принципів роботи та переваг перед DDoS атаками Прозорої Системи Виявлення Вторгнень (TIDS).

AVANT-GUARD - це нова структура, яка пропонує вдосконалений захист та гнучкість в мережах OpenFlow, як розширення до площини даних OpenFlow, яка вирішує дві проблеми безпеки. По-перше, використовує техніку міграції з'єднання на площині даних, щоб захистити інтерфейс між площиною управління та площиною даних та забезпечити взаємодію, що виникає під час DDoS-атак. По-друге,

створення спрацьовуючих тригерів на службах збору статистики в площині даних, які дозволяють площині управління прискорити реагування та виявлення для зміни правил потоку, таким чином, додатки можуть реагувати на загрози через мережеву статистику.

На ранніх стадіях DDoS-атаки трафік якогось типу обрізається, наприклад, атака Smurf з використанням пакетів ICMP. Це дозволяє дуже легко блокувати атаки трафіку на основі типу протоколу. На жаль, нещодавні дослідження показують, що інструменти DDoS-атаки постійно розвиваються і здатні створювати пакети з підробленими випадковими адресами джерела, вихідними портами і навіть генерувати корисне навантаження пакетів. Це значно ускладнює процес фільтрації та вимагає додаткових методів моделювання потоку. Цей підхід забезпечує практичне вирішення цієї проблеми шляхом розрізнення хороших і поганих пакетів і порівняння їх з попередньою історією. На відміну від алгоритмів фільтрації джерела IP, які намагаються описати трафік атак, замість цього ви можете описати звичайний трафік. Цей метод відноситься до класу виявлення аномалій. Дослідження показали, що більшість звичайних IP-адрес трафіку надсилали запити раніше. Статистичний аналіз інтернет-трафіку, зібраного мережею середнього розміру, показує, що невелика частина пулу IP-адрес, що входить до Інтернет-трафіку, продовжує з'являтися за щоденними спостереженнями. З цього можна зробити висновок, що надійною функцією визначення нормального трафіку є те, що він часто з’являється на веб-сайті. Фундаментальна проблема безпеки IP-мереж полягає в тому, що не існує схеми аутентифікації джерела IP-пакетів. Якщо маршрутизатор або веб-сервер можуть визначити, які адреси є легітимними, тоді проблему аутентифікації можна вирішити. Щоб розрізнити хороші та погані пакети, маршрутизатор або веб-сервер повинні вивчати історію попередніх мережевих з’єднань. Цей підхід полягає у записі IP-адрес усіх попередніх успішних мережевих з’єднань та складання бази даних IP-адрес. Коли наша мережа або веб-сайт відчуває серйозні перевантаження, ми можемо відмовитися від пакетів, вихідні адреси яких не відображаються в нашій базі даних IP-

адрес. У цьому випадку ми матимемо високу точність фільтрації пакетів у разі DDoS-атаки.

В другому розділі були розглянуті основні принципи, за якими зловмисники за допомогою DDoS атак можуть впливати на побудовані на базі SDN мережі передачі даних та хмарні обчислювальні середовища. Були розглянуті DDoS атаки на мережевий та транспортний рівні, а також на рівень додатків, що допомогло знайти вразливі місця SDN мереж. На основі цього були знайдені переваги SDN мереж в захисті від DDoS атак а також розглянуті існуючі методи захисту. Були розглянуті базові принципи роботи існуючих рішеннь захисту (такі як TLS, AVANT-GUARD та інші), що допомогло у подальшому аналізі вразливостей SDN мереж та аналізі принципів роботи та переваг перед DDoS атаками Прозорої Системи Виявлення Вторгнень (TIDS).

3 ТЕХНОЛОГІЯ ВИЯВЛЕННЯ І ЗАПОБІГАННЯ DDoS АТАК НА БАЗІ ХМАРНИХ СЕРВІСІВ

3.1 Принцип роботи механізму захисту від DoS атак

Системи виявлення вторгнень, також відомі як (IDS) функціонують як пристрої пасивного контролю, які попереджають адміністратора мережі або іншого мережевого пристрою у разі підозрілої мережевої активності. З іншого боку, системи запобігання вторгнень (IPS) активно беруть участь в обробці трафіку, оскільки процес запобігання вторгнень вимагає від них контролю за передачею даних.

Усі вищеперераховані типи пристроїв безпеки можна класифікувати за їх підходом до аналізу трафіку або методологією виявлення загроз. Одним з часто використовуваних методів аналізу трафіку є повна перевірка пакетів (DPI), яка забезпечує найширший спектр можливостей для виявлення, оскільки процес виявлення може базуватися на будь-якій інформації, що міститься в заголовку або в полі даних пакета. Останнім часом спостерігається значний розвиток та ріст популярності в іншому підході до аналізу трафіку, який називається «flow-based detection», незважаючи на певні властиві проблеми ефективності, що стосуються механізмів експорту та відбору зразків потоку у високошвидкісних мережах. Однак було запропоновано кілька розширень, які повинні підвищити продуктивність виявлення атак на основі потоку трафіку, особливо в області DDoS-атак.

У залежності від методики виявлення загрози виявлення вторгнень може бути на основі підпису або аномалії. В алгоритмі виявлення на основі підписів припускають, що кожна атака, з різною точністю, може бути описана набором правил та моделей пакетів, які порівнюються сканером з кожним вхідним пакетом в режимі реального часу. Отже, підпис атаки повинен бути відомий до того, як трапиться атака, щоб правильно конфігурувати відповідний шаблон. Насправді це часто не так, і таку стратегію зловмисники можуть легко використовувати на свою користь, встановивши, які шаблони розпізнаються пристроєм виявлення, і відповідно змінюючи налаштування атаки.

Друга група спеціальних алгоритмів виявлення, заснована на аномалії, покладається на те, що під час атаки одне або більше значень мережевих параметрів будуть значно відрізнятися від вимірюваних значень параметрів мережі. Ці алгоритми вимагають від сканера пройти навчальний період для встановлення базових значень. Виявлення на основі аномалії потребує меншого втручання та налаштувань з боку адміністратора і, ймовірно, більш надійне з точки зору виявлення атак, які описуються у раніше невідомих моделях атак.

Абсолютно незалежно від підходу до виявлення атак, IPS повинен виконувати свою функцію, не вводячи значних затримок, оскільки перевірка безпосередньо збільшує навантаження на мережеві пристрої та їх пропускну здатність. Апаратні засоби IDS зазвичай незначно збільшують навантаження на мережеві пристрої та вводять не таку велику затримку, оскільки вони зазвичай отримують копію трафіку, який передається через комутатор.

Найбільш поширена та відома на данний момент точка розгортання IPS або IDS у захищеній мережі (тобто мережі, яка є ціллю атаки) максимально наближена до ймовірного джерела атаки. Таке розгортання збільшує шанс на позитивне виявлення, оскільки пристрій IPS або IDS має доступ до агрегованого трафіку, що дозволяє йому отримати цілісний вигляд атаки в разі нападу. З іншого боку, розміщення пристрою призводить до створення вразливого місця і являється єдиною точкою відмови для високошвидкісних мереж. Тому логічним рішенням цього питання є впровадження масштабованого пристрою, який розподілятиме навантаження процесора, необхідне для обробки пакетів, на декілька процесорів, зберігаючи при цьому високу ефективність. Сучасні пристрої безпеки (побудовані для виявлення або запобігання атакам) повинні працювати майже в реальному часі, щоб мати можливість ефективно реагувати на загрози, пов’язані з великими обсягами трафіку, які часто зустрічаються в сучасних мережах.

При умові, що процес виявлення ґрунтується на перевірці пакетів, то робота IPS та IDS споживає значну продуктивності мережевих пристроїв та додає затримку якщо IPS/IDS вставлена на шляху трафіка. Тому забезпечення ефективної обробки пакетів

є важливою умовою таких систем IDS та IPS. Цього можна запобігти, розподіливши навантаження на декілька мережевих процесорів, а також за допомогою ефективної обробки пакетів у процесорах. Більшість алгоритмів розподілення навантаження розроблені для роботи з мережевими середовищами, які мають певні конфігурації хостів (наприклад, центри обробки даних з розподіленими веб-серверами тощо). Наприклад, деякі алгоритми базуються на розподілі запитів клієнтів порівну на кількість серверів дата-центра. Такі алгоритми за допомогою розподільного пристрою змінюють адреси призначення пакетів, тим самим перенаправляючи трафік до відповідного пункту призначення.

3.2 Комплексний метод виявлення DDoS атак

Для джерела аналізу отриманих даних, у якості прикладу виступає звичайний access log одного з найпопулярніших вебсерверів – Nginx.

Виконання програмного застосунку відбувається на прикладі аналізу інформації що отриманується з log-файлів цього сервера. Дані запитів до серверу Nginx зберігаються у log-файлі access. log, що містить

% H% l% u% t \ "% r \»% > s% b \ "% $\{Referer\}$ i \" \ "% $\{User-Agent\}$ i \», в котрому:

- % H - хост / IP-адреса, з якого зробленр зверненнядо сервера;

% T - час запиту до сервера і часовий пояс сервера;

% R - тип запиту, його версія і вміст;

% S - код стану HTTP запиту; % B - кількість переданих сервером байт;

% $\{Referer\}$ - URL-джерело звернення;

% $\{User-Agent\}$ - HTTP-заголовок, що має в собі інформацію про запит (мову, клієнтську програму і т. д.).

Комплексний метод виявлення розподілених атак на інформаційну систему включає наступні кроки:

— програма завжди проводить аналіз трафіку на факт початку атак, дані для аналізу отримуються з файлу access.log, далі вони проходять обробку та завантажуються у базу даних;

— у разі початку атаки фіксується точка початку атаки та в базі даних створюються дві додаткові таблиці, відповідні благонадійності трафіку та, відповідно трафіку, який пройшов після початку нападу;

— за допомогою алгоритму k-середніх трафік, пройшовший після початку атаки, ділиться на дві групи;

— трафік який позначений як шкідливий, присікається;

— на підставі швідливого трафіку створюються всі залежні правила заборони для файєрволу;

Для створення гнучкості кросплатформеності ПЗ розділене на три блоки:

— блок обробки даних і їх завантаження в базу даних;

— блок виявлення початку атаки;

— блок фільтрації трафіка.

Всі присутні в log-файлі дані можуть бути використані для аналізу. На підставі блоку GeoIP, можна виділити звернення з певного регіону, країни і т.д.

Структура розробленого програмного комплексу представлена на рис. 3.1:

— в результаті опрацювання звернень, що надходять до вебсерверу вони додаються в журнал;

— блок завантаження даних направляє їх в базу даних;

— відбувається аналіз звернень за допомогою модулю виявлення початку атаки. В разі його спрацювання в базі даних формуються дві порожні таблиці, одну для дозволених звернень, другу для нелегітимних;

— блок виявлення загрозливого трафіку відстежує їх стан і проводить кластеризацію і первинне заповнення таблиць, якщо в таблицях вже є дані, блок аналізує звернення, шо надійшли на предмет приналежності до груп надійних або нелегітимних звернень, і добавляю інформацію про зверненення у відповідну таблицю;

— модуль блокування запиту одержує паролі IP-адрес з таблиці, що містить шкідливі звернення і вносить їх в «чорні списки» файєрволу.

Сезонні методи аналізу показали більш високу точність у виявленні DDoS-атак і менший час від початку атаки до діагностики. Якщо під час аналізу файлів журналів виявлено доступність IP-адреси комп'ютера-ботнету, можна визначити точний час атаки. У середньому, виходячи з усіх перевірок і тестів, сезонний підхід зайняв у 4 рази менше часу на виявлення DDoS-атак і зменшив кількість помилкових спрацьовувань. Досить проблемною складністю, яка виникає при використанні цього методу, є правильний підбір цих періодів. Дані з цих серверів були відібрані для тестування з чітко визначеним періодом часу, який не викликав жодних сумнівів. Однак важко визначити різні періоди роботи, наприклад, великі магістральні маршрутизатори. Його діяльність може не обмежуватися щоденним або тижневим циклом, але може також мати свій власний цикл, який може бути складним циклом через додавання різних груп користувачів, наприклад користувачів із різних часових поясів. Крім того, існуючі сезонні цикли можуть змінюватися і можуть бути додані нові цикли, що вимагатиме постійного моніторингу переміщення упаковки та визначення нових робочих сезонних циклів.

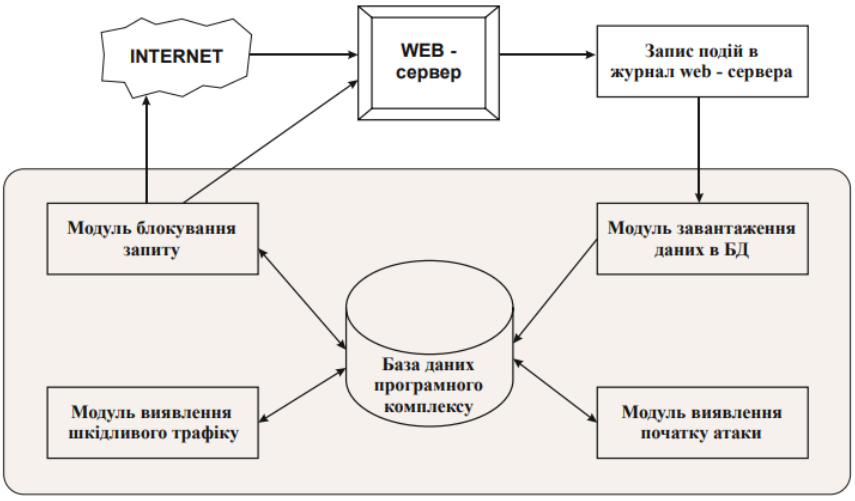

Рисунок 3.1 — Структура ПЗ

Усі отримані нами засоби відповідали цілям, поставленим у дослідженні. Робота програми, як і теоретична частина алгоритму, була перевірена в лабораторії з

використанням даних, що відповідають реальним DDoS-атакам. Універсальність пакетів для виявлення DDoS-атак можна використовувати для виявлення не тільки http-flood, але і DDoS - різних типів атак. За допомогою лише кількох змін програмне забезпечення можна використовувати для аналізу даних, що містяться у файлах журналів різних мережевих служб.

3.3 Реалізація захисту і атаки із використанням вебсерверу

Застереження, дане програмне забезпечення використано лише у демонстраційних та навчальних цілях.

За для виконання поставлених задач вирішено використання програмного забезпечення Terraform, а також мову HCL.

Terraform – це модуль, котриий надає можливість керувати зовнішніми ресурсами, до прикладу як хмарна інфраструктура, інфраструктура приватної хмари, мережеві апаратні засоби та інші. При виконанні таких задач використовуються спеціалізовані провайдери, див. програмний код нижче у лістингу 3.1.

Літиснг 3.1 — Модуль Terraform

```
terraform {
 required_providers {
   aws = {
     source  = "hashicorp/aws"
     version = "~> 3.0"
   }
 }
}
```

Саме цей фрагмент кодду надає можливість використання вбудованого провайдеру і що не мало важливо – проводити маніпуляції із AWS безпосередньо через

програмний код через код. Виконання та створення варіативності конфігурацій, використовуємо наступний програмниий модуль наведений у лістингу 3.2.

Лістинг 3.2 — Програмний модуль

```
resource "aws_launch_template" "course" {
  name = var.lc_name
  image_id = var.ami_id
  instance_initiated_shutdown_behavior = "terminate"
  key_name = var.key_name // !!
  instance_type = "t2.micro"
  user_data = filebase64("user_data.sh")
```

Усі вказані параметри описуємо як окремі змінні, такий підхід надає можливість використовувати шаблон необмежену кількість разів, а також змінювати його параметри, при умові що у цьому є вагома потреба. Для цього створюємо наступний файл із змінними, такий файл включає у себе певний набір параметрів, перерахований у коді нижче у лістингу 3.3.

Лістинг 3.3 — Файл із зімінними

```
variable "asg_name" {
  type        = string
  description = "Auto Scaling Group name"
}

variable "lc_name" {
  type        = string
  description = "Launch configuration name"
}
```

По завеершенню створення файлу із параметрами, переходимо до створення файлу із окремою конфігурацією код якої приведено у лістингу 3.4.

Лістинг 3.4 — Файл конфігурації

```
asg_name = "Laurel-scale"

desired_capacity = 0

vpc_zone = ["subnet-b9c398e3", "subnet-8d657ceb", "subnet-deb85095"]

min_size = 0

max_size = 32
```

У цьому фрагменті вкажемо необхіндні значення для розгортання системи. Така система розгортується за допомогою команди terraform apply. Вигляд розгортання демонструється у консолі розробника та зображений на рисунку 3.2.

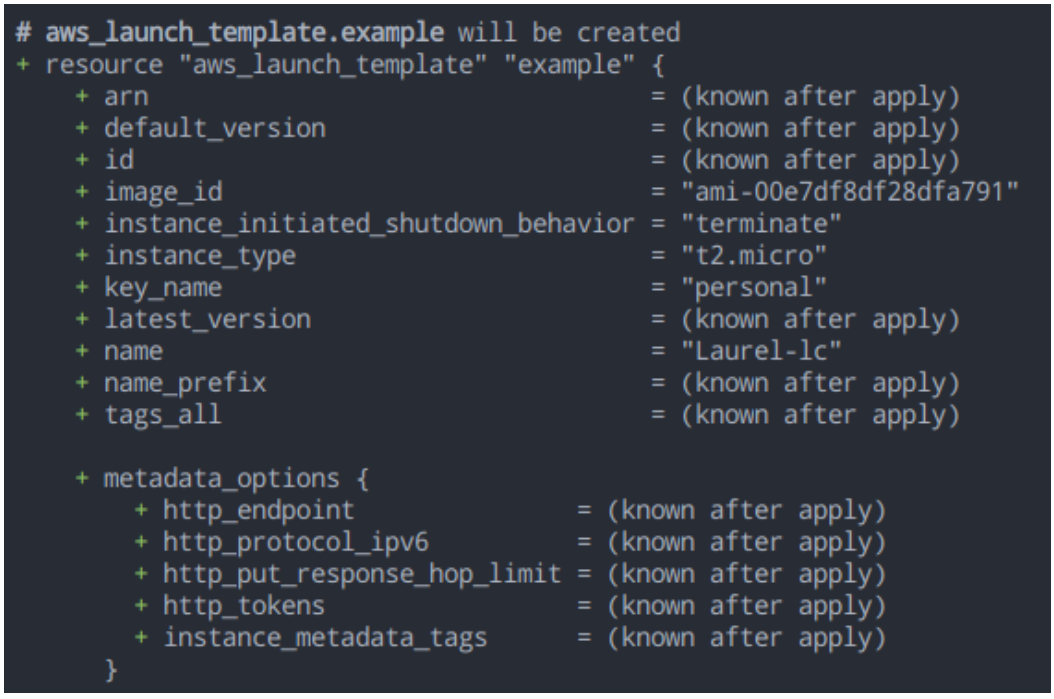

Рисунок 3.2 — Демонстрація розгортання інфраструктури

Аналогічним чином створюємо необхідну групу автоматичного масштабування серверів, котрі будуть додаватись або ж знищуватись при збільшенні або зменшуванні власне маштабування, код якого приведено у лістингу 3.5.

Лістинг 3.5 — Файл масштабування

```
resource "aws_autoscaling_group" "example" {
  name = var.asg_name
  capacity_rebalance  = true
  desired_capacity    = var.desired_capacity
  max_size            = var.max_size
  min_size            = var.min_size
  vpc_zone_identifier = var.vpc_zone
  health_check_grace_period = 180
  launch_template {
    id      = aws_launch_template.example.id
    version = aws_launch_template.example.latest_version
  }
}
```

Таким чином після після процесу розгортання цієї частини, автоматично створюється автономне масштабування серверів та шаблон запуску системи. Такі дії надають змогу гнучко та чітко визначати кількість серверів а також оперувати такими параметрами як кількість секунд до включення серверу в роботу.

Власне представлення автоматичного масштабування у інтерфейсі Amazon AWS зображено на рисунку 3.3.

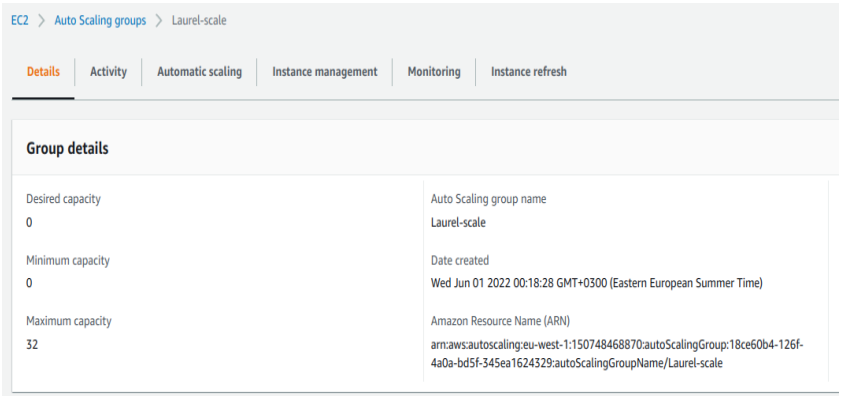

Рисунок 3.3 – Вигляд автоматичного масштабування у інтерфейсі

Після опису автоматичного масштабування та шаблону запуску, необхідно створити модуль балансування навантаження. Такий модуль - це апаратний засіб або ж технологія, яка діє як зворотній проксі-сервер та розподіляє мережевий або програмний трафік між кількома серверами. Вони використовуються для збільшення кількості користувачів у один і той самий момент часу та для надійності а також відмовостійкості програмних застосунків, код наведено у лістингу 3.6.

Лістинг 3.6 — Модуль балансування навантаження

```
resource "aws_lb" "example" {
  name               = "laurel-lb1"
  internal           = false
  load_balancer_type = "application"
  security_groups    = ["sg-08ec1e5c5fec5b8de"]
  subnets            = ["subnet-b9c398e3", "subnet-8d657ceb", "subnet-deb85095"]
  enable_deletion_protection = true
```

Після створення модуля балансування, для його стабільної роботи потрібно в свою чергу створити цільову групу, за для того, щоб вона автоматично керувала серверами, та мала змогу їх забирати або додавати до основного пулу. Оскільки було зазначено, що використовуватиметься сервер nginx і на ньому буде запущений сайт — протокол для групи вказується HTTP, а також 80 порт. Інакше кажучи буде виконано переадресація з 80 порту кінцевої точки модуля балансування, на 80 порт одного із серверів. Слід додати що є можливість перенадресації з 80 порту серверу на

443 порт модуля балансування, і використовувати протокол HTTPS для безпечного з’єднання. Фрагмент опису показаний нижче у лістингу 3.7.

Лістингу 3.8 — Фрагмент опису безпеченого з’єднання

```
resource "aws_lb_target_group" "example" {
  name     = "laurel-tg"
  port     = 80
  protocol = "HTTP"
  vpc_id   = "vpc-cb46eab2"
}
```

На рисунку 3.4 продемостровано загальний вигляд інтерфейсу після налаштування. Одним з ключових параметрів є статус серверу. Якщо він доступний — матиме статус healthy, якщо сервер не активний або демонструє нестабіільну роботу — статус unreachable.

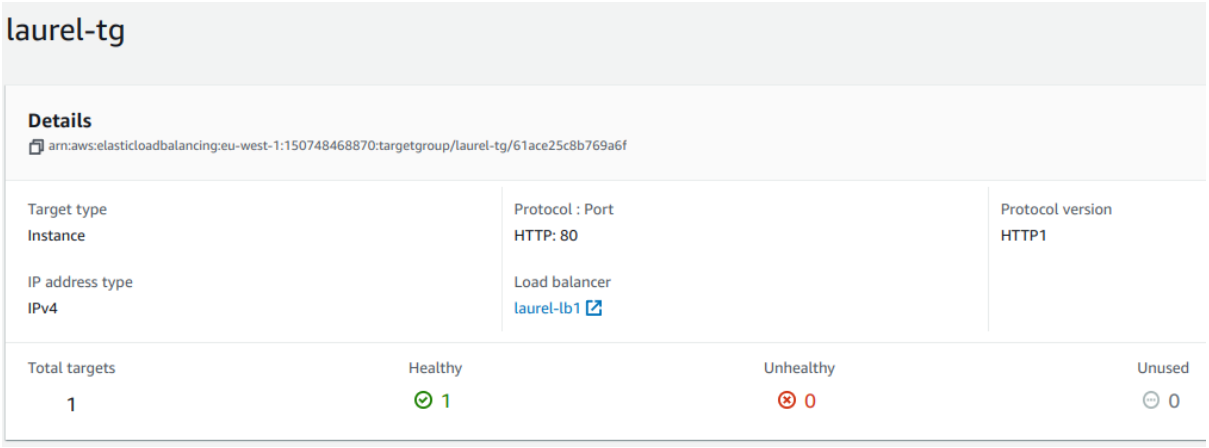

Рисунок 3.4 — Загальний вигляд інтерфейсу цільової групи

Структура Terraform проекту відображена на рисунку 3.5, де:

— main.tf зберігає у собі всю початкову конфігурацію;

— variables.tf містить змінні, які можна використати повторно;

— prod.tfvars відповідає за оголошення тих самих змінних;

— terraform.tfstate є файлом який зберігає всю інфраструктуру;

— user_data.sh файл який виконується одразу після запуску серверу.

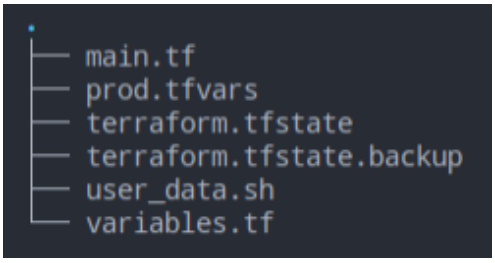

Рисунок 3.5 — Вигляд структури Terraform проекту

Наступним кроком виконуємо автоматизацію налаштування та встановлення всього необхідного програмного забезпечення на сервери. Для виконання такої задачі обрано програмні засоби Packer та Ansible. Вони схожі за структурою, але виконують різні дії. Packer використовується для створення первинного образу системи, код наведено в лістингу 3.9.

Лістинг 3.9 — Конфігурація Ansible

```
source "amazon-ebs" "generic" {
  profile         = "${var.profile}"
  ami_description = "${var.ami_name}-packer"
  ami_name        = "${var.ami_name}-packer-v${var.version}"
  instance_type   = "${var.instance_type}"
  region          = "${var.region}"
  source_ami      = "${var.source_ami}"
  ssh_username    = "${var.ssh_user}"
  temporary_security_group_source_cidrs =  []
```

Наведений вище фрагмент коду відповідає за базові параметри для створення образу (AMI) серверу. Так як усі створені параметри описані як змінні, існує можливість не редагувати основний файл, а лише виконати створення файлу із потрібними параметрами у конфігурації код якої наведено в лістингу 3.10

Лістинг 3.10 — Конфігурація для створення AMI

```
profile = "personal"
instance_type = "t2.micro"
region = "eu-west-1"
role = "nginx"
source_ami = "ami-00e7df8df28dfa791"
ssh_user = "ubuntu"
playbook = "playbook_nginx"
```

Деякі параметри необхідно вводити вручну перед запуском автоматизації і шукати безпосередньо у AWS, такі як:

— instance_type. Тип серверу, який буде використовуватись;
— source_ami. Початковий образ серверу, в курсовій роботі використовується Ubuntu 20.04;
— playbook. Параметр який дає змогу запускати Ansible під час створення образу серверу.

Враховуючи мету бакалаврської дипломної роботи, створення автоматизованої підсистеми, у якості програмного засобу для управління конфігурацією було використано Ansible. Користувач Ansible виконує створення «плейбуки» у форматі YAML з описом необхідних штатних керованих систем. «Плейбук» — це є опис стану ресурсної системи, у котрому вона повинна перебувати у конкретний момент часу, включаючи встановлені системні пакунки, виконувані на даний момент часу служби, створені файли та багато іншого. Ansible перевіряє, що кожен із ресурсів системи знаходиться в очікуваному стані і запитує та виправляє стан ресурсу, якщо він не відповідає очікуваному.

Виконуючи вказані завдання використовуємо система модулів. Кожне завдання являє із себе назву завдання, модуль що використовується і список параметрів, які описують завдання. Ansible підтримує змінні, фільтри (за допомогою бібліотеки Jinja2), умовне виконання завдань, паралелізацію, шаблони файлів. Адреси та

налаштування цільових систем містяться в статичних файлах «інвентарю» (inventory), або ж визначаються динамічно через «плаґіни інвентарю». Підтримує групування. Для реалізації набору подібних завдань існує система ролей, а для поширення уніфікованих наборів контенту, як-то плейбуків, різних типів плаґінів і ролей, є Ansible Collections — формат пакунків, які зберігаються у публічному реєстрі ansible-galaxy.

Натомість маємо створений «плейбук» котрий виконує основні завдання, це встановлення і запуск серверу nginx та оновлення скелету Linux дистрибутиву. Код якого наведено в лістингу 3.12

Лістинг 3.12 — конфігурація playbook

```
- name: "upgrade and install dependencies"
  hosts: default
  become: true
  roles:
    - InstallDependencies
- name: "install nginx"
  hosts: default
  become: true
  roles:
      - nginx
```

Наступним кроком було створено ролі, котрі містять у собі більше інформації про конкретні налаштування. Наприклад, зупинка автоматичного оновлення. Код для якої описано в лістингу 3.13

Лістинг 3.13 — конфігурація створення ролі

```
- name: Stop and disable service apt-daily.timer
  ansible.builtin.service:
    name: apt-daily.timer
    state: stopped
    daemon_reload: yes
    enabled: no
```

Структура цієї частини підсистеми показана на рисунку 3.6.

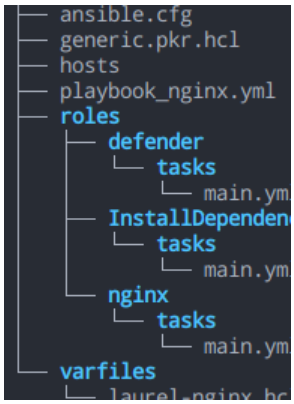

Рисунок 3.6 — Вигляд файлової структури проекту налаштуванню сервера

Перш за все, потрібно вказати, що програмне середовище та використані засоби, в якому була здійснена реалізація автоматизованої підсистеми налаштування хмарного середовища, повинна забезпечити повну надійність, можливість швидкого відклику та роботу на будь-якій операційній системі з огляду на наявність необхідних програмних застосунків.

Саме тому буде проводитись детальне тестування системи на фінальних етапах, тестування буде проведено задля перевірки правильності роботи всього запланованого функціоналу та забезпечення безпеки програмного продукту.

Виконаємо тестування створення початкового образу сервера. Команда для запуску “packer build -var 'created_by=vadim' -var-file="./varfiles/laurel-nginx.hcl" ./generic.pkr.hcl”

Образ системи створено, перевіримо чи він існує у AWS. Результат показаний на рисунку 3.7.

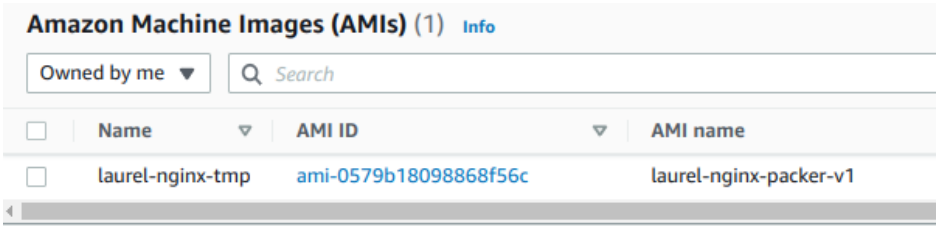

Рисунок 3.7 — Вигляд інтерфейсу AMI

Виконуємо перевірку роботи розгортання інфраструктури. Для цього вводимо наступні команди:

— terraform init
— terraform plan
— terraform apply

Виконуємо контроль наявності група автоматичного масштабування, сервери та балансувальник навантаження Результат зображений на рисунку 3.8.

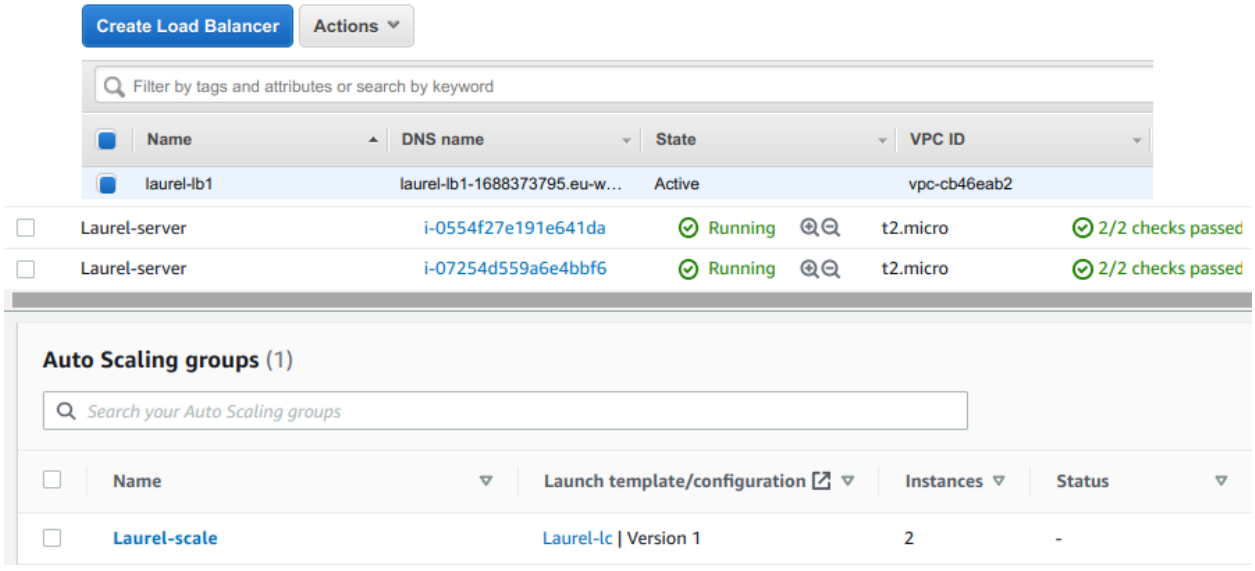

Рисунок 3.8 — Перевірка розгортання інфраструктури

Виконані перевірки проведені успішно. Наразі виконаємо спробу пошуку по IP адресі балансувальника навантаження, і перевірити чи є доступ до тестової сторінки,

і чи навантаження балансується між 2-ма серверами. Результат показаний на рисунку 3.9.

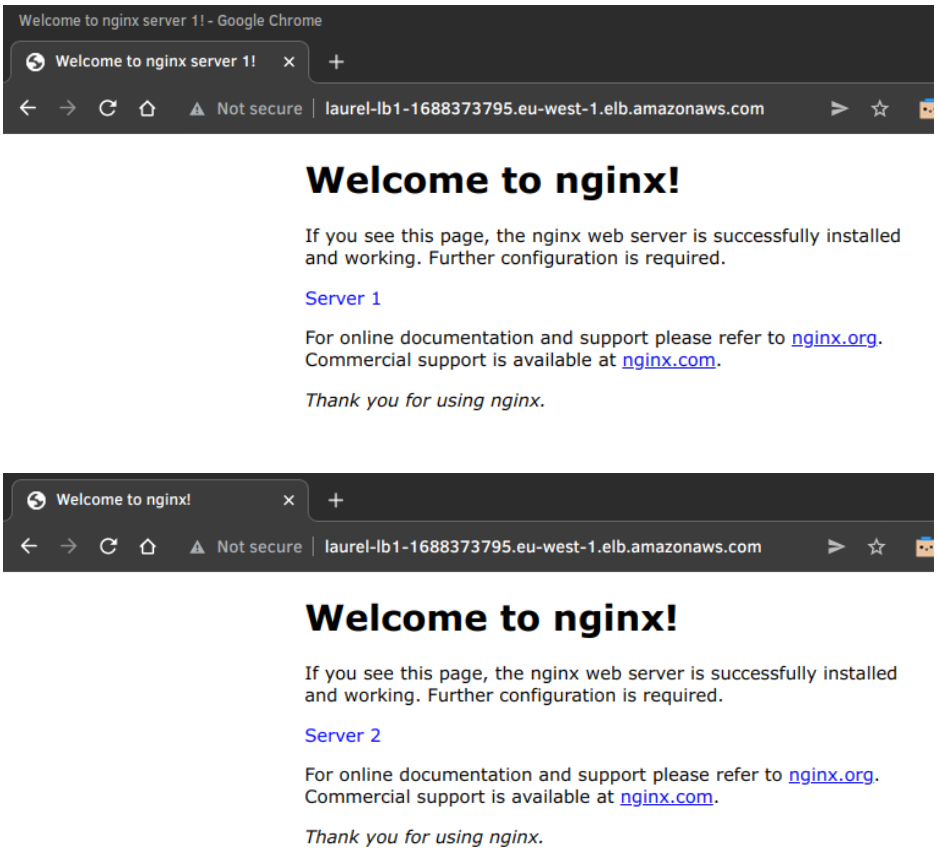

Рисунок 3.9 — Перевірка балансувальника навантаження

Перейшовши по DNS адресі балансувальника навантаження, і оновивши декілька раз сторінку — можна бачити що кожного разу сторінка змінює надпис, показуючи який сервер зараз прийняв запит. Отже, балансувальник навантаження працює. Наступне що необхідно протестувати, це роботу автоматичного масштабування серверів, оскільки це є основною і невід'ємною частиною відмовостійкої інфраструктури. Для цього, будемо імітувати велике навантаження на CPU та на мережу за допомогою DoS. Результат показано на рисунку 3.10.

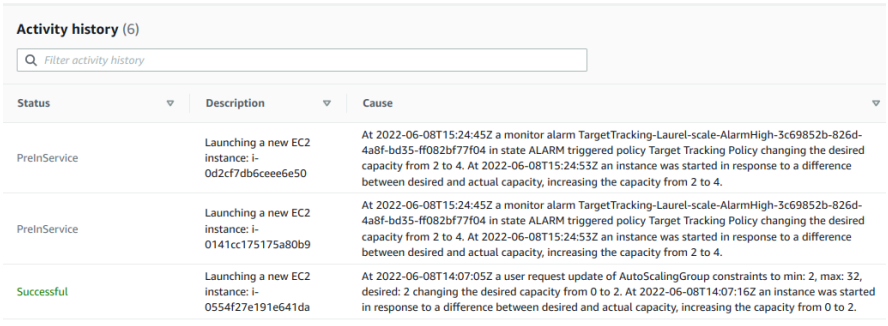

Рисунок 3.10 — Тестування автоматичного масштабування серверів

Виконавши стресс-тестування — автоматично було запущено +2 сервери щоб балансувати навантаження. Тестування виконано успішно.

Таким чином, підбиваючи підсуммки проведеної роботи у підрозділі, було здійснено опис основних програмних засобів та реалізовано базову автоматизовану підсистему налаштування хмарного середовища.

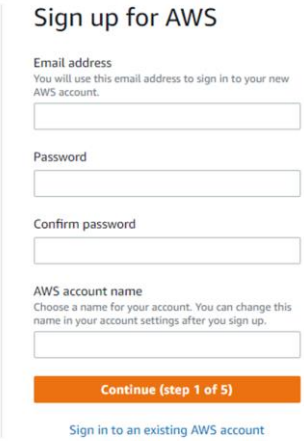

Рисунок 3.11 — Реєстрація аккаунту

Реєструємось та прив’язуємо картку. Не переживайте, за це, можете використати віртуальну або картку іншого банку. Амазон дає безкоштовно 720 годин в місяць на сервери.

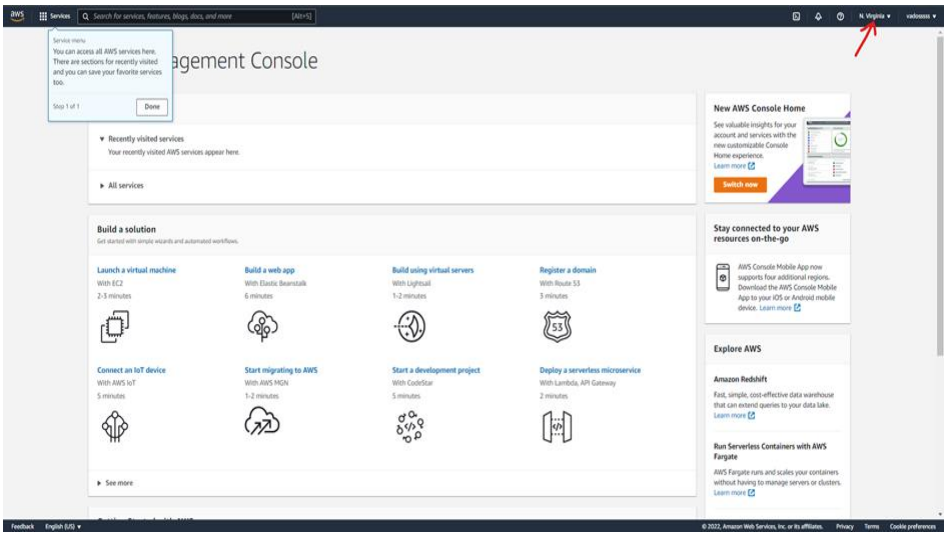

Рисунок 3.12 — Вибір регіону

Йдемо за стрілочкою та вибираємо регіон Ireland. Далі переходимо по посиланню https://eu-west-1.console.aws.amazon.com/vpc/home?region=eu-west-1#subnets: і копіюємо всі subnet-id’s.

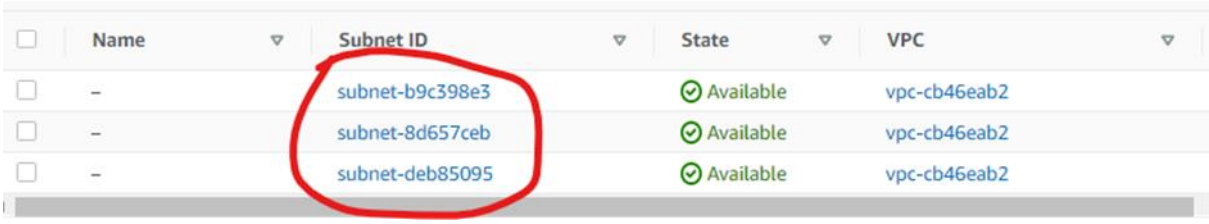

Рисунок 3.14 — Вибір підмережі

Відкриваємо файл Ireland.tfvars і вставляємо у поле vpc_zone

```
// Set-up main info
region = "eu-west-1"
ami_id = "ami-0bf84c42e04519c85"

// Set-up Auto Scaling Group
asg_name = "DDoS-scale"
desired_capacity = 0
vpc_zone = ["subnet-b9c398e3", "subnet-8d657ceb", "subnet-deb85095" ]
min_size = 0
max_size = 4
```

Рисунок 3.15 — Налаштування конфігурації

У файлі user_data.sh, де є IP:PORT/TYPE - вказуємо ip адресу, порт та тип пакетів.

```
#! /bin/bash
set -x
sudo yum install docker -y
sudo service docker start
sudo docker run -d --rm alpine/bombardier -c 10000 -d 600s -l IP:PORT/TYPE >> /
home/ec2-user/log.txt
sudo shutdown now
```

Рисунок 3.16 — Вибір порта

Далі йдемо у https://eu-west-1.console.aws.amazon.com/ec2/v2/home?region=eu-west-1#KeyPairs: і генеруємо ключ.

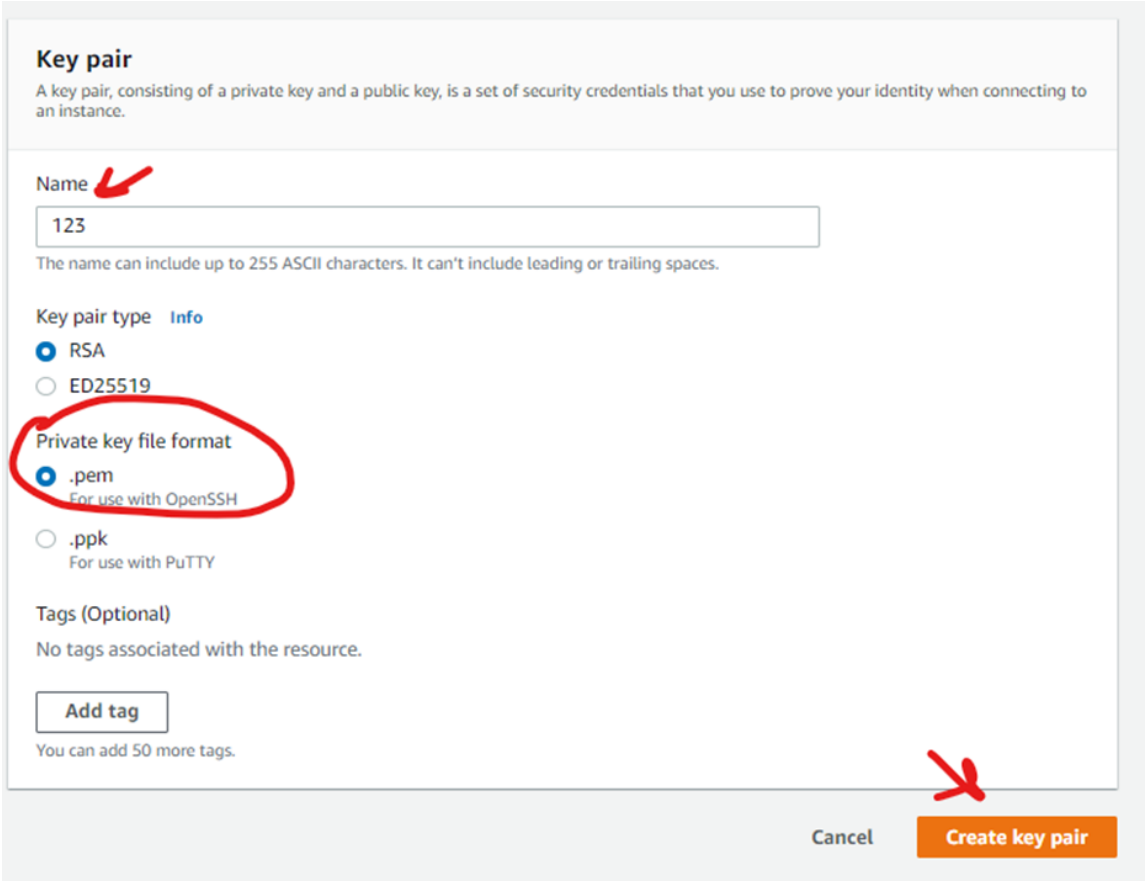

Рисунок 3.17 — Генерація ключа

Переходимо сюди та генеруємо ключі для AWS CLI

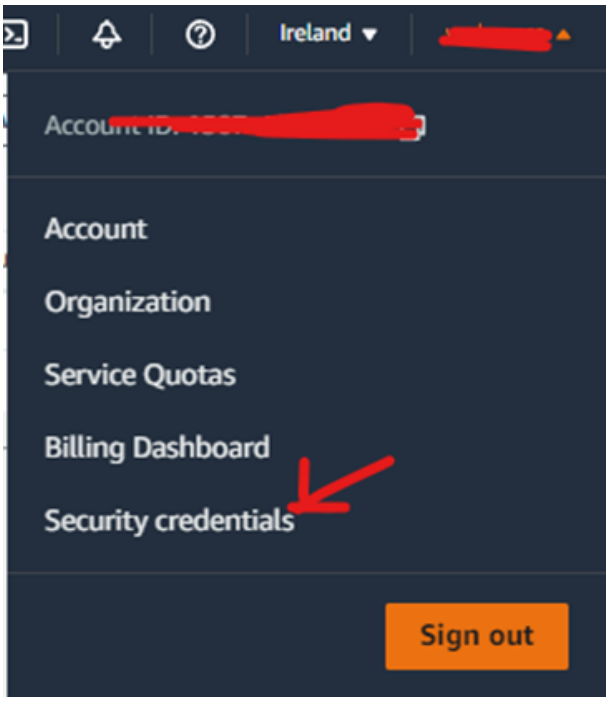

Рисунок 3.18 — Параметри безпеки

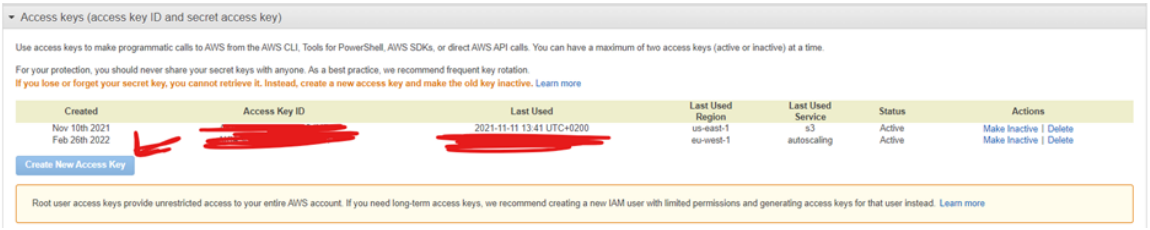

Рисунок 3.19 — Створення нового ключа

Завантажуємо файл і копіюємо з нього, або копіюємо одразу ключі

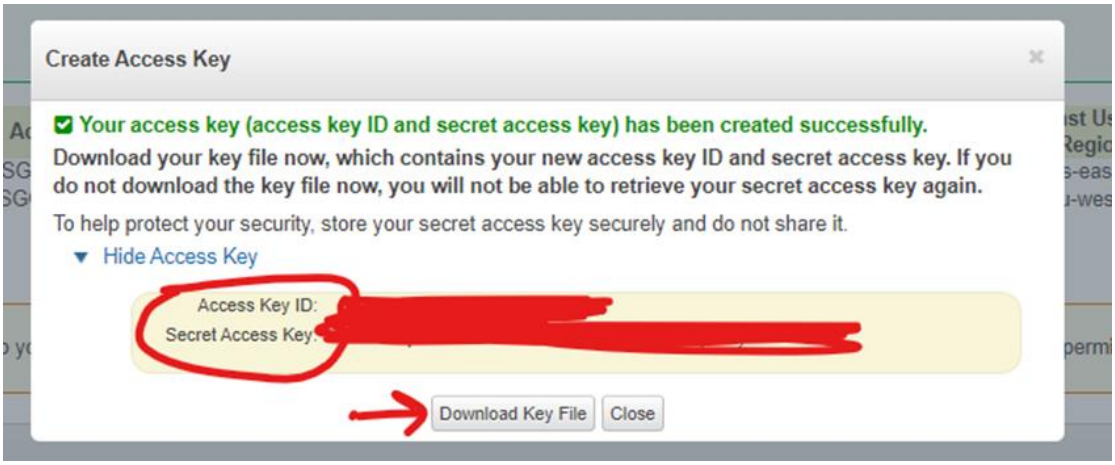

Рисунок 3.20 — Завантаження ключа

Йдемо сюди та вибираємо операційну систему, встановлюємо AWS CLI

https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

Пишемо у консолі aws configure та вводимо спочатку Access Key, а потім Secret Access Key.

Встановлюємо Terraform за посиланням https://learn.hashicorp.com /tutorials/ terraform/install-cli Якщо все виконано правильно вводимо наступні команди:

terraform init

terraform plan -var-file=ireland.tfvars

terraform apply -var-file=ireland.tfvars

Переходимо сюди https://eu-west-1. console. aws. amazon. com/ec2/v2/home? region=eu- west-1#Instances:v=31 і перевіряємо чи з’являються сервери, можливо треба зачекати хвилин 10.

Якщо сервери швидко “потухають” і підіймаються знову - перевіряємо всю конфігурацію, додаємо у Security Group 22 порт, який доступний всюди, і заходимо на сервер.

Зміна регіону виконується наступним чином

```
// Set-up main info
region = "eu-west-1"
ami_id = "ami-0bf84c42e04519c85"
```

Рисунок 3.21 — Заміна регіону

Для зміни аккаунту AWS - міняємо у ireland.tfvars параметр profile

```
// Set-up main info
region = "eu-west-1"
ami_id = "ami-0bf84c42e04519c85"
profile = "default"
```

Рисунок 3.22 — Вибір акаунту AWS

3.4 Концепція підходу ефективної протидії атакам на відмову

Враховуючи існуючі загрози у відповідь, звісно, розвивались системи захисту від атаках на відмову в обслуговуванні. На початку були прості індикатори, які фіксували, наприклад, кількість байт в секунду або кількість відкритих з'єднань. Механізми захисту паралельно ускладнювались з появою більш складних типів атак. При цьому відбувалось залучення арифметичних моделей з області статистики, нейронних мереж, імітаційного моделювання та інших .

Сучасні системи виявлення атак – це системи прийняття рішення в умовах невизначеності інформації, динамічних змін середовища та можливих загроз. Для визначення аномальних явищ у таких системах використовуються складні математичні алгоритми та спеціально побудовані бази знань. Однак існуючі системи захисту через вищезазначені причини не можуть забезпечити достатній рівень виявлення і протидії атакам на відмову. Для ефективної протидії атакам на відмову в сучасних мережах система захисту має задовольняти таким вимогам.

Вимоги безпеки в організації можуть змінюватися або змінюватися з часом. Тому при зміні параметрів і налаштувань системи повинна змінитися і її функція.

Відстежувана мережа може змінюватися з часом. Це може бути пов'язано з додатковими можливостями або ресурсами. Тому система захисту повинна мати можливість змінювати свою функцію без перезавантаження - онлайн. Система проксі може забезпечити необхідну гнучкість, встановлюючи цілі для кожного проксі. Коли відбувається зміна, ціль змінюється, що не викликає перезапуску.

Основні та найважливіші характеристики, які дозволяють виявляти нові атаки. Враховуючи атаку, видно, що сценарій атаки постійно змінюється, шукаючи нові вразливості чи реалізації. Рекомендується проводити навчання двома способами. По-перше, адміністратори повинні встановлювати нові цілі для розумних агентів. Іншим способом є самонавчання агентів і методів, що використовують інтелектуальну обробку інформації (вилучення знань, статистичні моделі, нейронні мережі тощо).

Однією з властивостей мережі є взаємозв'язок її компонентів. Для успішної роботи необхідна чітка взаємодія всіх компонентів (маршрутизатор, роутер, файловий

сервер, персональний комп'ютер). Зловмисник може атакувати всю мережу або її частини, просто атакуючи одне з посилань. Наприклад, він може заповнювати мережу фіктивними запитами ICMP від імені третіх сторін. Або запустити атаку на провайдера. Тому системи виявлення атак на основі атак можуть бути неефективними. Розподілений моніторинг з різних частин мережі виглядає більш ефективним.

Для значного спрощення завдання виявлення атак необхідний розподіл обчислювальних завдань, що виконуються на різних вузлах. Це значно скоротить час відповіді, але вам також слід створити систему для обміну інформацією, яка доповнить вимірювання вузла даними з іншого місця. Іншим аспектом, пов'язаним з автономією, є функція делегування. Динамічні процеси в комп’ютерних мережах часто вимагають зміни параметрів безпеки, щоб негайно застосувати їх у відповідь на атаку. Наприклад, чим раніше ввімкнено фільтр, який виявляє адреси, тим нижча потужність атаки. Ось чому авторизація елементів системи на визначення певних функцій управління системою значно зменшить час реагування та загальну ефективність системи захисту. Основою створення такої системи є технологія інтелектуальних агентів із застосуванням методів статистичного аналізу та теорії ігор. Методи, за допомогою яких інтелектуальні системи вирішують складні завдання, особливо в області управління комп’ютерними мережами, описані та продемонстровані в багатьох роботах. Мультиагентні системи більш мобільні, крім того, вони мають інші функції, такі як розповсюдження, здатність працювати в умовах непередбачуваних змін мережі та шкідливої активності, виявлення та запис основних подій, навчання, аналіз зібраної інформації, планування дій, автономність та адаптивність.

Відповідно до наведених вище матеріалів, існуючі системи виявлення можна розділити на системи виявлення аномалій і системи виявлення ознак. Основним недоліком систем виявлення сигнатур є те, що вони призначені для виявлення конкретних типів атак (зазвичай найнебезпечніших при створенні системи). Коли з’являються нові атаки або змінюються характеристики трафіку, завдання виявлення потрібно вирішувати заново. Спеціальні системи виявлення аномалій (через складність моделювання нормального інтернет-трафіку) використовують різні припущення щодо

функціонування системи, наприклад статистичну однорідність трафіку. Однак не обговорюється група комп'ютерних систем, до яких стосуються ці припущення, або умови, за яких вони реалізуються. Тому невеликі зміни в структурі трафіку або наданих послугах можуть призвести до необхідності нової підготовки алгоритмів виявлення.

Конкретним та ефективним рішенням цієї ситуації є використання комплексного підходу до побудови систем, що захищають від атак, включаючи моніторинг системи, збереження історії транзакцій, підтримку спеціального репозиторію для шпигування за зловмисниками та їхньої поведінки Аналіз, стратегія прийняття рішень. Рекомендується будувати систему захисту на основі наступних елементів:

— агенти стеження;

— агенти попередньої обробки і зберігання;

— сховище для зберігання інформації про трансакції, що описують функціонування системи;

— сховище з аналітичними компонентами для виявлення загроз та ознак здійснення зловмисної активності;

— агенти протидії атакам.

Важливим елементом побудови такої системи є визначення відповідного математичного забезпечення для кожного етапу роботи.

Спостерігання за трафіком. Перехоплення пакетів з метою оцінки завантаження, складу трафіка, активності користувачів. Для здійснення цієї задачі необхідно розробити алгоритми визначення кількості і частоти перехоплення пакетів в залежності від завантаження каналу й інших параметрів. Якщо пакети будуть перехоплюватися занадто часто це може призвести до вповільнення трафіку. Якщо ж пакети будуть перехоплюватися через певні постійні проміжки часу це може створити «сліпі зони», про які не буде відомостей.

Попередня обробка захоплених пакетів, оцінка найбільш небезпечних загроз, збереження інформації у сховищі. Оскільки на цьому етапі необхідна швидка оцінка

з мінімальними затратами ресурсів, доцільно використовувати прості й адаптивні порогові значення або (за необхідності) послідовний CUSUM.

Аналіз даних при завантаженні у сховище, виявлення атак, оцінка загроз. Після запису інформації в сховище можна провести комплексну оцінку й обчислити можливі загрози. Для цього доцільно використовувати вищеописані багатоканальні алгоритми CUSUM та сковзне середнє.

Фоновий аналіз даних для встановлення спроб сканування, атак погіршення якості, пульсуючих атак. Здійснюється постійно або за розкладом. Оскільки ці атаки представляють меншу загрозу, то є час для більш детального їх аналізу. Використовуються методи Data Mining, системи інтелектуальних правил, нейронні мережі тощо.

Прийняття рішення про виявлення атаки. При перевищенні певних порогових значень на одному з попередніх етапів формується признак про можливу загрозу атаки. В цьому випадку має бути створена експертна система, яка б могла оцінити рівень загрози та прийняти рішення про здійснення атаки.

Оцінка загрози, вибір моделі, її верифікація, пошук стратегії. Виявлення атаки відразу ставить питання про визначення протидії. У залежності від типу і навіть особливостей конкретної атаки така протидія може значно змінюватися.

У роботі розглядалися питання захисту від одного з найбільш небезпечних видів зловмисної діяльності в мережі Інтернет – атак на відмову. Описана історія виникнення проблеми та причини, що зумовили її появу. Проведено огляд основних типів атак та їх класифікація. Окремо розглянуті системи, що підлягають захисту – кожна з них має свої характеристики, важливі при побудові системи виявлення атак. Система протидії або захисту від атак на відмову має вирішувати наступні задачі: попередження атаки, виявлення атаки, ідентифікація джерел атаки, протидія атаці. Задача виявлення атаки полягає в детектуванні атаки на відмову в разі її появи, це важливий етап, від якого залежать всі подальші дії. Тому алгоритмам виявлення надається велике значення. Вони мають задовольняти вимогам за швидкістю, надійністю, ефективністю.

ВИСНОВКИ

У бакалаврській дипломній роботі було проаналізовано сучасний стан DDoS атак, та запропоновано їх класифікацію. Проведено аналіз нинішнього стану технологій для вирішення проблем захисту інформації, проаналізовано методи, що застосовуються для рішення задач пов’язаних з доступністю ресурсів. Розроблено математичну модель атак з врахуванням опису сезонності мережного навантаження.

Також використовуючи модель атаки запропоновано метод раннього визначення та протидії DDoS-атакам.

Розроблено алгоритм визначення точок початку атаки і алгоритм поділу змішаного трафіку на надійний та загрозливий, який враховує сезонну кількість мережного навантаження.

Запропоновано критерії успішності, які дозволяють оцінювати успішність роботи алгоритму роботи, і сторонніх засобів за допомогою фільтрації трафіку;

Описано інформаційну систему та розроблено її функціональну модель. Розроблено діаграму варіантів використання інформаційної системи виявлення мережних атак .

Проведено ряд тестів, які підтверджують ефективність та результативність розробленого програмного забезпечення для протидії DDoS-атакам та їх виявлення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Басалаєва, Ю.С. Вибір інструментів Data Mining для аналізу результатів дистанційної освіти / Ю.С. Басалаєва // Сучасні матеріали, техніка та технологія. - 2015. - № 2. - С. 22 - 25.

2. Бабаш, А.В. Информационная безопасность. Лабораторный практикум : учеб. пособие / А. В. Бабаш, Е. К. Баранова, Ю. Н. Мельников. - 2-е изд., стер. - М. : КНОРУС, 2016. - 132 с.

3. Бабаш, А.В. Криптографические методы защиты информации : учебник для студ. вузов / А. В. Бабаш, Е. К. Баранова. - М. : КНОРУС, 2016. - 190 с.

4. В. Л. Бурячок, “Сучасні системи виявлення атак в інформаційнотелекомунікаційних системах і мережах. Модель вибору раціонального варіанта реагування на прояви стороннього кібернетичного впливу”, Інформаційна безпека, № 1, с. 33-40, 2013.

5. Гремякіна О. А. Вибір платформи інтелектуального аналізу даних для застосування в академічних цілях // Молодий вчений. - 2015. - №22. - С. 26-29.

8. Д. Б. Мехед, Ю. М. Ткач, В. М. Базилевич, В. І. Гур’єв, та Я. Ю. Усов, “Аналіз вразливостей корпоративних інформаційних систем”, Захист інформації, т. 20, № 1, с. 61-66, 2018. doi: 10.18372/2410-7840.20.12453.

9. Джулій В.М. Метод виявлення та протидії розподіленим атакам, спрямованим на відмову в обслуговуванні / В.М. Джулій, В.І. Чорненький, О.О. Савіцька, - Хмельницький: Вісник Хмельницького національногоуніверситету, 2019. - Вип. №1. – С.127-134

10. І. Яковів, “Інформаційно-телекомунікаційна система, концептуальна модель кіберпростору і кібербезпека”, Information Technology and Security, № 5 (9), с. 134-144, 2017.

11. Зінченко В. В. Виявлення DDoS-атак прикладного рівня / В. В. Зінченко, М. В. Зінченко // Міжнародна науково-технічна конференція «Радіотехнічні поля, сигнали, апарати та системи» (Київ, 16-22 березня 2015). - К., 2015. - С. 262-264.

12. Грайворонський М. В. Безпека інформаційно-комунікаційних систем: підручник/ М. В. Грайворонський, О. М. Новіков ; За заг. ред. М.З. Згуровського. — К. : Видавнича група BHV, 2009. — 608 с.

13. David Dittrich, Jelena Mirkovic, Peter Reiher, Sven Dietrich. Internet Denial of Service: Attack and Defense Mechanisms/ David Dittrich, Jelena Mirkovic, Peter Reiher, Sven Dietrich // — 1. — Москва: Pearson Education, 2004. — С. 400. — ISBN 0132704544, 9780132704540.

14. PHP Manual / Mehdi Achour, Friedhelm Betz, Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana - 2020-05-26 — https://www.php.net/manual/en/

15. ТЕХНОЛОГІЇ ВИЯВЛЕННЯ ТА ЗАПОБІГАННЯ DDoS АТАК РЕАЛІЗОВАНИХ З ВИКОРИСТАННЯМ ХМАРНИХ СЕРВІСІВ, Савицька Л. А., В. О. Нич, С. В. Богомолов. електронні наукові видання, LI Науково-технічна конференція факультету інформаційних технологій та комп`ютерної інженерії. – 2022. − 2 с.

ДОДАТОК А

Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ
проф., д.т.н.. Азаров О.Д.
“28” квітня 2022 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання бакалаврської дипломної роботи
“Технологія виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів”
08-23.БДР.027.00.000 ТЗ

Науковий керівник: доцент к.т.н.
_____ Савицька Л.А.

Студент групи 2КІ-18б
_____ Нич В.О.

м. Вінниця – 2022

1 Підстава для виконання бакалаврської дипломної роботи (БДР)

Підставою для розробки даної бакалаврської дипломної роботи є наказ ВНТУ № від «__» ___ 2022 року та рішення засідання кафедри обчислювальної техніки (протокол №_1_ від «_____»__ 2022 року).

2 Вихідгі дані для виконання БДР

Розглянуті принципи роботи Ddos атак. Проведення аналізу сучасних підходів до засобів. Розробле Terraform темплуту для Ddos атаки інтернет ресурсів.

3 Перелік задач

Перелік задач, що повинні бути виконані:
—розглянуто принципи роботи Ddos атак;
—проведення аналізу сучасних підходів до засобів;
—розолюити Terraform темплейту для DDoS ататаки інтернет ресурсів.

4 Перелік матеріалів, що подаються до захисту БДР

До захисту подається: пояснювальна записка БДР, графічні і ілюстративні матеріали, протокол попереднього захисту БДР на кафедрі, відгук наукового керівника, відгук рецензента, анотації до БДР українською та іноземною мовами, довідка про відповідність оформлення БДР діючим вимогам.

5 Порядок контролю виконня та захисту БДР

Робота виконується в три етапи, таблиця А.1

Таблиця А.1 — Етапи виконання роботи

№ з/п	Назва етапів виконання комплексної бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	14.03.22	

Продовження таблиці А.1

2	Аналіз предметної області	15.03-18.03.22	
3	Вплив DoS атак на хмарні середовища та мережі передачі даних	21.03-24.03.22	
4	Визначення методів та інструментаріїв захисту від DDoS атак	25.03-30.03.22	
5	Принцип роботи механізму захисту від DoS атак	31.03-04.04.22	
6	Вибір технологій для створення клієнтської частини застосунку	05.04-.11.04.22	
7	Підготовка матеріалів та опис розробки інформаційної системи	23.05-26.05.22	
8	Аналіз виконання роботи, висновки, додатки	27.05-31.05.22	
9	Перевірка якості виконання бакалаврського проекту та усунення недоліків	01.06 -.11.06.22	

6 Обов’язковий ілюстративний матеріал

До обов’язкового ілюстративного матеріалу входить алгоритми по визначенню початку атаки і виділенню шкідливого трафіку

7 Порядок контролю та прийому

Виконання етапів графічної та розрахункової документації БДР контролюється науковим керівником згідно зі встановленими термінами. Захист БДР відбувається на засіданні Екзаменаційної комісії, затвердженої наказом ректора.

8 Вимоги до оформлювання та порядок виконання БДР

При оформлюванні БДР використовуються:

— ДСТУ 3008 : 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302 : 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— ГОСТ 2.104-2006 «Єдина система конструкторської документації. Основні написи»;

— документами на які посилаються у вище вказаних.

Технічне завдання до виконання отримав _____ Нич В.О.

ДОДАТОК Б

Алгоритми по визначенню початку атаки

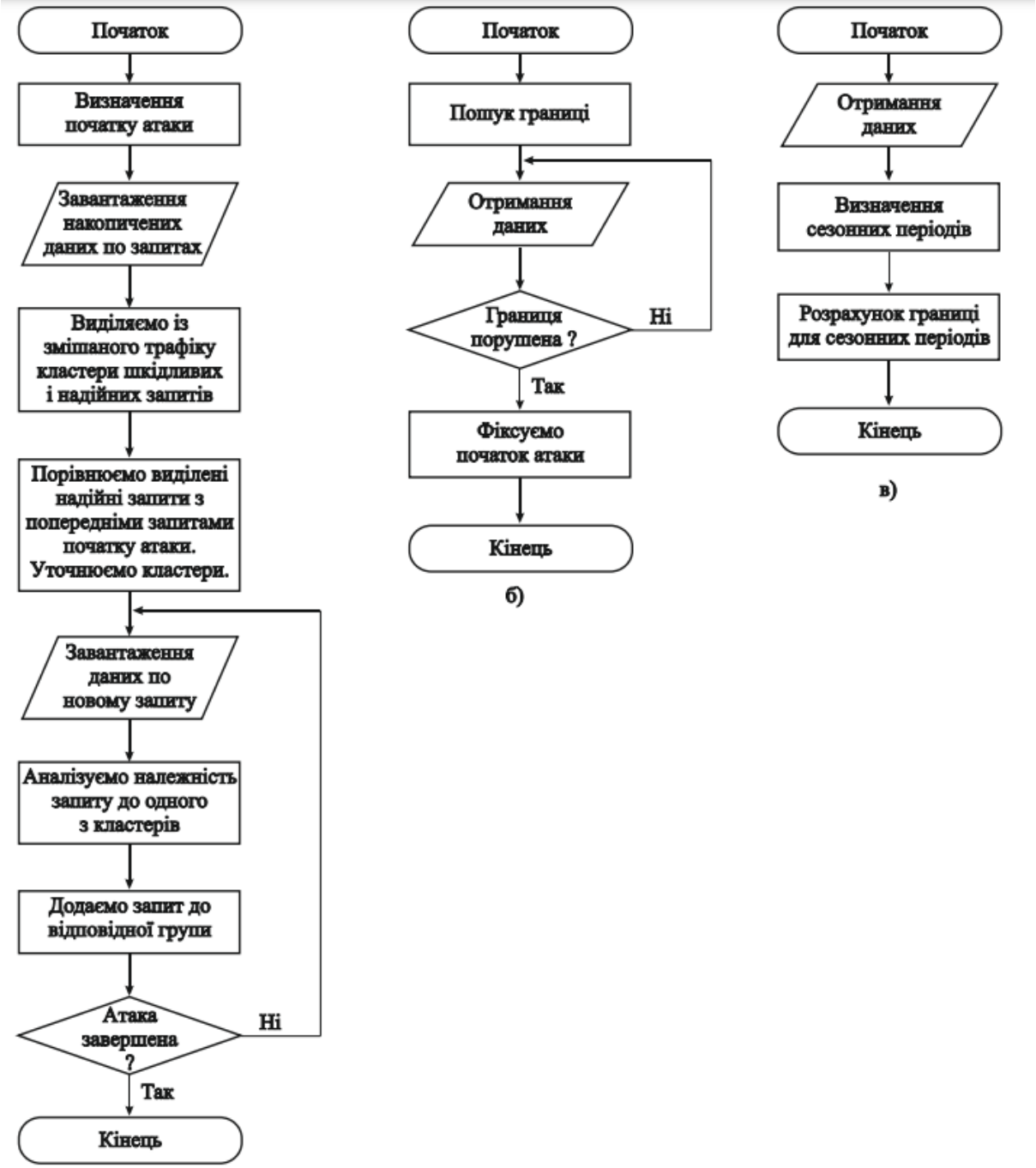

Рисунок Б — Алгоритми по визначенню початку атаки

ДОДАТОК В

Лістинг методу Terraform

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  region = var.region
  profile = var.profile
}

resource "aws_launch_template" "example" {
  name = var.lc_name
  image_id = var.ami_id
  instance_initiated_shutdown_behavior = "terminate"
  key_name = var.key_name // !!
  instance_type = "t2.micro"
```

```
  user_data = filebase64("user_data.sh")
  tag_specifications {
    resource_type = "instance"
    tags = {
      Name = "Laurel-server"
    }
  }
  tag_specifications {
    resource_type = "volume"
    tags = {
      Name = "Laurel-server"
    }
  }
  tag_specifications {
    resource_type = "network-interface"
    tags = {
      Name = "Laurel-server"
    }
  }
}
```

```
resource "aws_autoscaling_group" "example" {
  name = var.asg_name
  capacity_rebalance  = true
  desired_capacity    = var.desired_capacity
  max_size            = var.max_size
  min_size            = var.min_size
  vpc_zone_identifier = var.vpc_zone
  health_check_grace_period = 180
  launch_template {
    id      = aws_launch_template.example.id
    version = aws_launch_template.example.latest_version
  }
}
```

ДОДАТОК Г

Лістинг AWS provider vars

```
variable "region" {
  type        = string
  description = "AWS Region"
}

variable "ami_id" {
  type        = string
  description = "AWS AMI"
}

variable "profile" {
  type        = string
  description = "set profile"
}

variable "asg_name" {
  type        = string
  description = "Auto Scaling Group name"
}
```

```
variable "lc_name" {
  type        = string
  description = "Launch configuration name"
}

# variable "security_groups" {
#     type = list(string)
#     description = "Specify security group list"
# }

variable "max_size" {
  type = number
    description = "Max size of autoscale group"
}

variable "min_size" {
  type = number
    description = "Min size of autoscale group"
}

variable "vpc_zone" {
```

```
  type = list(string)
    description = "Specify subnet"
}

variable "key_name" {
  type = string
    description = "Specify SSH key for access to AWS instances"
}

// Mixed instances policy part

variable "desired_capacity" {
    type = number
    description = ""

}
```

ДОДАТОК Д

Протокол перевірки навчальної (кваліфікаційної) роботи

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Технологія виявлення і запобігання DDoS атак реалізованих з використанням хмарних сервісів

Тип роботи: бакалаврська дипломна робота
(КДР, МКР)

Підрозділ кафедра обчислювальної техніки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 84,2% Схожість 15,8%

Аналіз звіту подібності (відмітити потрібне):

a. Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.
- Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
- Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку (підпис) Захарченко С.М. (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи (підпис) Ким В.О. (прізвище, ініціали)

Керівник роботи (підпис) Савицька А.І. (прізвище, ініціали)