МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних систем та мереж

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

комп'ютерних систем і мереж

_____ Жуков І.А_____

(підпис) (ПІБ)

"_____"_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬО-КВАЛІФІКАЦІЙНОГО РІВНЯ

"БАКАЛАВР"

напрям підготовки - 6.050102 "Комп'ютерна інженерія"

Тема: Комп'ютерна мережа на основі маршрутизації SDN

Виконавець: _____ Нікітін Я.В.

(підпис) **(ПІБ)**

Керівник: _____ Телешко І.В.

(підпис) **(ПІБ)**

Нормоконтролер: _____ Журавель С.В

(підпис) **(ПІБ)**

Київ 2023

Факультет_____
Кафедра комп'ютерних систем та мереж
Спеціальність 123 "Комп'ютерна інженерія"

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних систем та мереж
_____ Жуков І.А
(підпис) (ПІБ)
"_____" _____2023 р.

ЗАВДАННЯ
на виконання дипломного проекту

Нікітіну Ярославу Володимировичу
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема дипломного проекту: Комп'ютерна мережа на основі маршрутизації SDN
затверджена наказом ректора від "26"квітня 2023 р., № 591
2. Термін виконання проекту: з "22" травня 2023 р. по "25" червня 2023 р.
3. Вихідні дані до проекту Комп'ютерна мережа на основі маршрутизації SDN у середовищі Mininet

4. Зміст пояснювальної записки:Аналіз напрямків розвитку комп'ютерних мереж на базі SDN; Реалізація робочого середовище для моделювання SDN мереж; Методи створення та застосування мережевих топологій SDN з використанням Mininet**.**

5. Перелік обов'язкового графічного (ілюстративного) матеріалу:
Презентація Power Point

6.Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1.	Ознайомитись з завданням на виконання дипломного проекту.	22.05.23-25.05.23	
2.	Вивчити спеціальну літературу по темі дипломного проекту.	26.05.23-28.05.23	
3.	Визначити план змісту проекту.	29.05.23-31.05.23	
4.	Зробити опис основних засобів і методів побудови локальної комп'ютерної мереж	01.06.23-05.06.23	
5.	Проаналізувати робоче середовище для моделювання SDN мереж	06.06.23-08.06.23	
6.	Зробити опис створеної мережі	09.06.23-12.06.23	
7.	Налаштувати створеної мережевої топології у середовищі Mininet	13.06.23-17.06.23	
8.	Оформити пояснювальну записку	18.06.23-20.06.23	
9.	Підготувати презентацію	21.06.23-22.06.23	
10.	Захистити дипломний проект	23.06.23-25.06.23	

7. Дата видачі завдання: "22" травня 2023 р.

Керівник дипломного проекту _____ Телешко І.В.

(підпис керівника) (ПІБ.)

Завдання прийняв до виконання _____ Нікітін Я.В.

(підпис випускника) (ПІБ.)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту “ Комп’ютерна мережа на основі маршрутизації SDN ”: 45 с., 28 рис., 14 літературних джерел, 2 додатока.

Об’єкт дослідження – Комп’ютерна мережа на основі маршрутизації SDN

Предмет дослідження – Вплив маршрутизації SDN на ефективність та надійність комп'ютерної мережі

Мета дипломної роботи – Створення комп’ютерної мережі на основі маршрутизації SDN

Методи дослідження – моделювання мереж, вимірювання продуктивності мережі, методи обробки та аналізу комп’ютерних мереж.

Технічні та програмні засоби – Середовище моделювання Mininet є віртуальним інструментом, який дозволяє створювати та емулювати комп'ютерні мережі. Воно дозволяє досліджувати, тестувати та випробовувати різні мережні сценарії без необхідності фізичного обладнання.

Прогнозні припущення щодо розвитку об’єктів розроблення – Розробка та моделювання комп'ютерних мереж на основі SDN

Актуальність теми комп’ютерна мережа на основі маршрутизації SDN

У сучасному світі, комп’ютерні мережі є невід'ємною частиною інформаційного суспільства та займають центральне місце в розвитку сучасних технологій та послуг. Зростаюча складність та потреба в гнучкості та ефективності мережевих інфраструктур вимагають нових підходів до маршрутизації та управління мережами. Одним з таких підходів є технологія маршрутизації SDN .

Технологія SDN надає можливість програмно керувати мережею, відокремлюючи планування маршрутизації від фізичного обладнання. Це дає змогу розробникам та адміністраторам мереж розглядати мережу як програмований об'єкт і використовувати алгоритми маршрутизації, які найкращим чином відповідають потребам мережі.

Такі мережі є одними з ключових інновацій у мережевій індустрії. Вони дозволяють використовувати відкриті стандарти та програмні інтерфейси для розробки нових рішень у галузі мереж. SDN надає можливість розробникам створювати власні алгоритми маршрутизації, впроваджувати нові сервіси та забезпечувати безпеку мереж.

ЗМІСТ

Кафедра КСМ							
Виконав	Нікітін Я.В.			Комп'ютерна мережа на основі маршрутизації SDN	Літера	Аркуш	Аркушів
Керівник	Телешко І.В.					5	
Консульт.					КС-431		
Н. контр.	Журавель С.В						
Зав. каф.	Жуков І.А.						

ПЕРЕЛІК СКОРОЧЕНЬ,УМОВНИХ ПОЗНАЧЕНЬ

SDN – Software determined network (програмно-конфігурована мережа)

NFV – Network function virtualization (технологія віртуафлізації фізичних мережевих елементів)

ЦОД – центр обробки даних

API – Application programming interface (інтерфейс програмного програмування)

ВМ – Віртуальна машина

VM - Virtual machines (віртуальна машина)

ПЗ – програмне забезпечення

ARP - Address Resolution Protocol (протокол визначення адрес)

RIP - Routing Information Protocol (протокол маршрутної інформації)

EIGRP - Enhanced Interior Gateway Routing Protocol (пропрієтарний протокол маршрутизації)

BGP - Border Gateway Protocol (протокол граничного шлюзу)

OSI - The Open Systems Interconnection model (мережева модель)

WAN - Wide Area Network (глобальна комп’ютерна мережа)

OS - Operating system (програмне забезпечення)

ОС - Програмне забезпечення

ВСТУП

Комп’ютерні мережі є невід'ємною частиною сучасного світу, який все більше залежить від передачі даних та безперервного з'єднання. Швидкий розвиток технологій, збільшення кількості підключених пристроїв та зростання обсягу обміну даними ставлять перед комп’ютерними мережами нові виклики. Для забезпечення ефективності, масштабованості та гнучкості мереж, виникають нові підходи та технології.

Одним з таких інноваційних підходів є маршрутизація на основі програмного визначення мереж (SDN - Software-Defined Networking). SDN змінює традиційний підхід до мережевого управління, розділяючи контрольну площину від пропускної площини. Це дозволяє забезпечити більшу гнучкість, автоматизацію та керованість мережі.

Таким чином, актуальність дослідження теми "Комп’ютерна мережа на основі маршрутизації SDN" полягає в необхідності розуміння та використання цього новаторського підходу для побудови ефективних та масштабованих комп’ютерних мереж. Це дозволить забезпечити більшу гнучкість, швидкість реакції та покращену керованість мережевими ресурсами.

У даному дипломному проєкті було розглянуто різні аспекти комп’ютерних мереж, побудованих на основі маршрутизації SDN. Дослідження цієї теми допоможе розкрити потенціал SDN для покращення продуктивності та управління комп’ютерними мережами.

Метою цього дипломного проєкту є розробка та реалізацію комп’ютерної мережі на основі маршрутизації SDN з використанням середовища моделювання Mininet.

РОЗДІЛ 1

АНАЛІЗ НАПРЯМКІВ РОЗВИТКУ КОМП'ЮТЕРНИХ МЕРЕЖ НА БАЗІ SDN

1.1 Історія появи мережевої технології SDN.

Програмно-конфігурованиі мережі SDN (Software Defined Network), здається, виникала відносно недавно, але насправді ж у неї є тривала передісторія. Технологія SDN - це завершальна фаза робіт, спрямованих на те, щоб зробити мережі програмованими.

На сьогоднішній день комп'ютерні мережі вирізняються складністю управління та структури.Вони використовуєть безліч устаткування, таких як: маршрутизатори та комутатори, транслятори мережевих адрес, а також пристрої та системи розпізнавання. Бувають випадки коли мережеві адміністратори налаштовують пристрої окремо, дотримуючись правил маршрутизації та цілісності мережевої топології . Інтерфейси конфігурації використовуються для кожного пристрою і для кожного виробника. Також використовується пакетний метод конфігурації пристроїв, загалом цей підхід доволі сильно ускладнює процес модернізації та розвитоку мереж. Неспроможність впровадження інновацій також призводить до великих капітальних і операційних витрат. Візуальне зображення мережі SDN зображено на Рис. 1.1.:

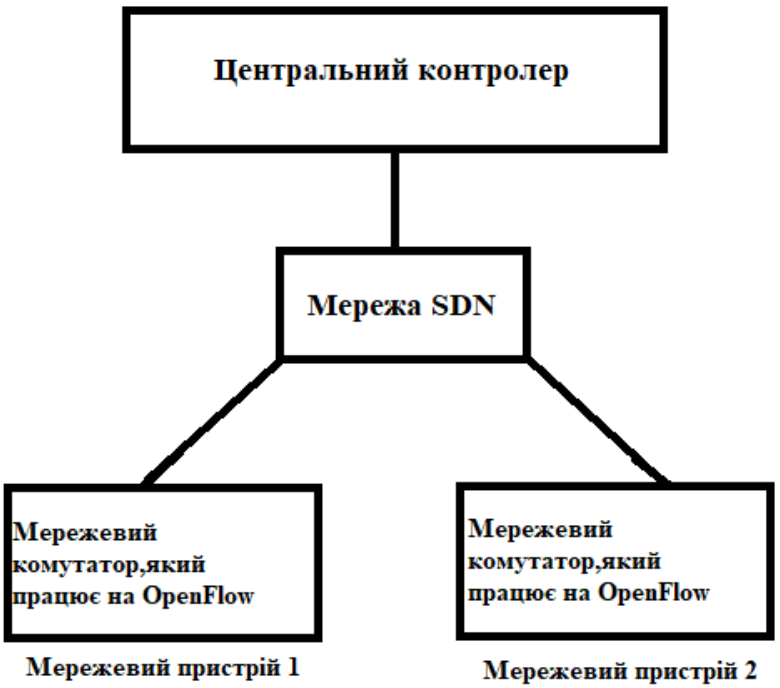

Рисунок 1.1 - Візуальне зображення мережі SDN

Програмно-конфігурована мережа (SDN) змінює проектувальний підхід та управління мережею. По-перше, SDN розділяє площину керування мережею, яка відповідає за пересилання трафіку, і площину даних, яка спрямовує трафік відповідно до правил, отриманих від площини керування. По-друге, SDN інтегрує контрольну точку з набором програм керування на сервері з багатьма пристроями в площині даних. Для цього використовуються стандартизовані інтерфейси прикладного програмування (API). Інтерфейс OpenFlow є чудовим прикладом прикладного програмування. Тому для побудови мережі SDN необхідно реалізувати підтримку OpenFlow в елементах мережі. Одночасно буде одна або кілька таблиць, і кожна таблиця повинна містити правила маршрутизації. Кожне з яких визначає, як пересилаються пакети для певного потоку трафіку. У той же час всі комутатори OpenFlow можеуть діяти як комутатор, маршрутизатор або транслятор мережевих адрес.

Дивлячись на те що концепція SDN набула популярності в останні роки, сама концепція не є новою і розробляється вже понад 20 років. Сліди також можна простежити до розвитку перших телефонних мереж з комутацією каналів, де керування мережею було відокремлено від мереж голосового трафіку з комутацією каналів. SDN використовує спрощене управління та розгортання послуг.

Концепція так званого «програмного комутатора» для комп'ютерних мереж на основі комутації пакетів також дуже схожа на SDN з точки зору функціональності та реалізації.

Ідея програмованої мережі вперше почала формуватися з баченням SDN концепції «активної мережі», але ця концепція так і не прижилася. Тоді з суто практичних міркувань були зроблені спроби розділити площини управління та передачі даних, щоб краще спрямувати трафік.

Нарешті, зусилля OpenFlow і рідної операційної системи досягли оптимального балансу між передбаченням і практичністю. Коли SDN застосовували для віртуалізації мережі, цінувався баланс між широким і чітким баченням і практичною стратегією для широкого впровадження.

SDN продовжує розвиватись, і ми спостерігаємо все більшу комерціалізацію цієї технології. SDN часто рекламують як панацею від усіх мережевих проблем, але слід пам’ятати, що SDN – це лише інструмент, який полегшує вирішення проблем керування мережею. SDN дозволяє розробляти нові програми та розробляти рішення «старих» проблем.

1.2.Дослідження архітектури технології SDN

Концепція програмно визначеної мережі (SDN). Відокремлює рівень керування мережею від рівня передачі даних, делегуючи функції керування (маршрутизатори, комутатори тощо) програмам, що працюють на окремих серверах (контролерах). Залежно від розміру вашої мережі, контролер може бути сервером або групою серверів із встановленим спеціальним програмним забезпеченням. У той же час мережеві елементи з вилученими функціями керування мережею виконують суто основне завдання пересилання пакетів.

Ця архітектура дозволяє відокремити площину управління від мережевого обладнання та програмувати (визначати програму або налаштовувати програму). У той же час базова інфраструктура передачі даних також відокремлена від мережевих служб і програм. Використовуючи цей підхід, керування мережею можна розвантажити на окремі централізовані обчислювальні ресурси (контролери SDN), які обслуговують всю інфраструктуру.

Це дозволяє представити всю інфраструктуру як єдиний «логічний» комутатор/маршрутизатор для програм, які використовують мережеві функції.

Архітектура SDN

Мережу SDN можна розділити на три рівні, кожен з яких складається з різних компонентів.

- Прикладний рівень.
- Рівень контрольної площини
- Рівень інфраструктури або даних

Візуальне зображення рівнів мережі SDN зображено на Рис. 1.2:

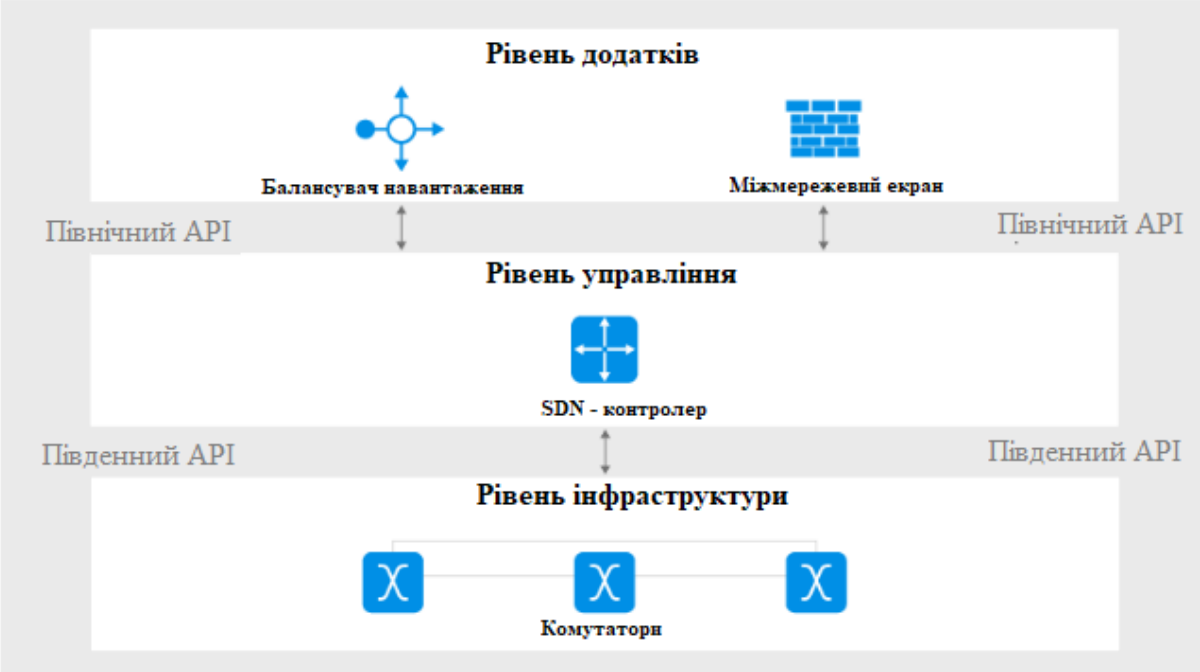

Рисунок 1.2 - Мережі рівні SDN

1. Рівень додатків.

Цей рівень реалізує гнучке та ефективне управління мережею за допомогою різних бізнес-додатків. Основне призначення цього рівня - контроль виконання завдань, необхідних додатку. Іншими словами, контролер спілкується і надає відповідні команди мережі через API.

2. Рівень управління.

Забезпечує маршрутизацію та доступ для трафіку СЦБ. Цей рівень включає мережеву операційну систему для управління мережею та мережевими пристроями за допомогою набору пов'язаних сервісів через API та відкриті інтерфейси.

Функція рівня управління полягає в централізованому конфігуруванні та управлінні, створенні пакетів і призначенні їх маршрутизаторам. Тобто це програмне забезпечення яке призначене передавати мережевим комплектуючим SDN здобуті команди та вказівки від рівня даних. Контролер витягує та передає інформацію та дані з апаратних пристроїв разом з іншими діями в мережі для передачі їх до програм SDN. Програмне забезпечення контролера SDN також знаходиться тут, що дозволяє заповнювати площину даних централізованим керуванням пристроями шляхом усунення рівня керування окремими комутаторами та маршрутизаторами. Крім того, рівень керування, що використовує IPv4, IPV6 і ARP (протокол розпізнавання адрес), керує протоколами маршрутизації, такими як: RIP (протокол інформації про маршрутизацію), OSPF (спершу відкрити найкоротший шлях), EIGRP (розширений протокол маршрутизації внутрішнього шлюзу) і BGP (протокол прикордонного шлюзу).

3. Рівень інфраструктури (даних).

Цей рівень схожий на фізичний рівень моделі OSI (Open Systems Interconnection) і включає всі елементи мережі, що забезпечують передачу даних - це фізичні та віртуальні пристрої та канали передачі даних. Функція рівня інфраструктури полягає у фізичному пересиланні пакетів від його вхідного інтерфейсу до його вихідного інтерфейсу за допомогою транспортного протоколу, використовуючи рівень керування. Ці рівні також включають основні поняття, такі як: бізнес-додатки, мережі та служби безпеки, комутатори SDN та гібридні комутатори.

Типи SDN

Усі програмно визначені мережі пропонують можливість централізованого керування потоками даних у програмному забезпеченні комутаторів і маршрутизаторів, але існують різні моделі SDN.

Відкрита мережа SDN. Мережеві адміністратори використовують такі протоколи, як OpenFlow, для керування роботи віртуальних і фізичних комутаторів на рівні даних.

SDN з API-інтерфейсами. Управління переміщенням даних по мережі на кожному пристрої здійснюється за допомогою використання інтерфейсів API.

Модель накладання SDN. У цьому типі SDN віртуальні мережі працюють на існуючій апаратній інфраструктурі, створюючи динамічні тунелі до різних локальних і віддалених центрів обробки даних. Віртуальна мережа розподіляє пропускну здатність між різними каналами та призначає пристрої кожному каналу без потреби у фізичній мережі.

Гібридна мережа SDN. Ця модель поєднує програмно визначену мережу та традиційні мережеві протоколи в єдине середовище, яке підтримує широкий спектр мережевих функцій. Частина трафіку все ще маршрутизується за стандартними мережевими протоколами, тоді як SDN обробляє решту трафіку. Це дозволить мережевим адміністраторам поступово інтегрувати SDN у свої застарілі середовища.

Відмінності SDN від традиційних мереж.

Відмінності між розподіленими мережами в традиційних мережах і централізованими мережами з контролерами SDN в основі мережі полягає :

Традиційна мережа — це повністю розподілена мережева структура, у якій кожен маршрутизатор самостійно обчислює записи маршрутизації. Кожен маршрутизатор у мережі збирає інформацію про стан зв'язку як вхідні дані для алгоритмів маршрутизації, які обчислюють найкоротший маршрут до заданого пункту призначення. Коли відбувається зміна топології мережі, маршрутизатори надсилають великі обсяги інформації про стан нових каналів і самостійно обчислюють нові маршрути.

Тому в разі збою традиційні мережі мають автоматичне перемикання. Надійність також має високу ефективність.

У мережі SDN контролер SDN є основним компонентом і централізовано керує мережевими пристроями, тому конвергенція мережі залежить від контролера SDN. Це може створити єдину точку відмови. Візуальне зображення відмінності SDN від традиційних мереж зображено на Рис. 1.3:

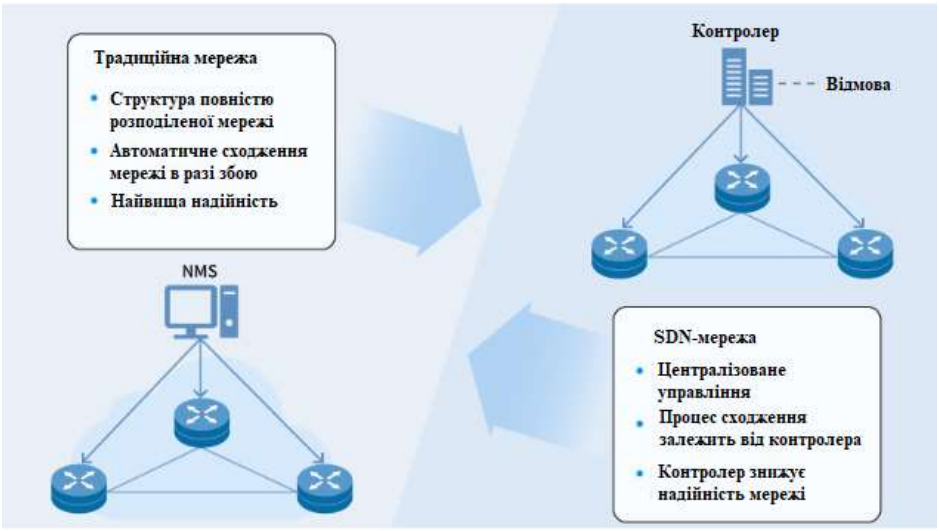

Рисунок 1.3 - Відмінності SDN від традиційних мереж

Переваги SDN

SDN має багато переваг перед традиційними мережами::

1.Більші можливості керування з підвищеною швидкістю та гнучкістю..

Замість того, щоб вручну програмувати декілька апаратних пристроїв від певного постачальника, розробники можуть програмувати готові контролери на базі програмного забезпечення з відкритим кодом для керування трафіком у мережі. Крім того, мережеві адміністратори можуть вибирати з більш широкого діапазону мережевих пристроїв, тому що вони можуть обирати протоколи з відкритим кодом для доступу до будь-якою кількості апаратних пристроїв за допомогою центрального контролера.

2.Мережева інфраструктура, що налаштовується.

Завдяки SDN адміністратори конфігурують мережеві служби та розподіляють віртуальні ресурси для централізованої зміни мережевої інфраструктури в реальному часі. Це дозволяє мережевим адміністраторам покращити потік мережевих даних, визначаючи пріоритети програм, які потребують вищого рівня доступності. рівень доступності.

3.Високий рівень безпеки.

SDN забезпечує видимість та більш повне розуміння виявлених загроз безпеці у всій мережі. З розповсюдженням розумних пристроїв, підключених до Інтернету, SDN пропонує незаперечні переваги перед традиційними мережами. Розробники можуть проектувати окремі зони для пристроїв, які вимагають різного рівня безпеки, і миттєво ізолювати вразливі пристрої від хакерських атак у мережі.

1.3. Варіанти використання SDN мереж.

SDN-мережі застосовують у різних галузях: додатки для спільної роботи, мережевий обмін, послуги мобільної мережі, мережі центрів обробки даних.

Основні варіанти використання SDN мереж.

Домени доступу та агрегація комунікаційних мереж загального користування та центрових мереж обробки інформації.

Домен мережі доступу включає mobile backhaul (ретрансляційна мережа передачі даних, що з'єднує базові станції).

Завдяки цьому принцип віртуалізації SDN відіграє важливу роль у ізоляції різних груп послуг і створенні спільної інфраструктури, яка може використовуватися кількома операторами. З іншого боку, використання SDN у центрах обробки даних може покращити балансування навантаження та динамічний розподіл ресурсів (підвищити використання мережі та серверів), а також підтримувати міграцію серверів за допомогою конфігурації комутатора.

Розподілені принципи SDN можуть підтримувати розробку постанов для центрів обробки даних на основі стандартних комутаторів, а не типового дорогого апаратного забезпечення високого класу.

Центри обробки даних мають одну з найбільших і найважливіших галузей інтересу для постачальників послуг. Центри обробки даних знаходять використання у нових великих корпоративних мережах. У цьому контексті постачальники глобальних мереж (WAN) центрів обробки даних стають критично важливою фігурою ефективної та рентабельної роботи. Малюнок 3 ілюструє це. Тепер оператори використовують статичні механізми статичної ініціалізації для пошуку ресурсів у глобальних мережах, що не тільки погіршує ефективність центрів обробки даних, але й збільшує їх експлуатаційні витрати. Тому існує потреба в тому, щоб глобальна мережа між центрами обробки даних була мережевим ресурсом, який можна розподіляти та надавати з такою ж мобільністю та гнучкістю, що й центри обробки даних. SDN є чудовим кандидатом для того, щоб забезпечити такі гнучкі операції WAN, які очікують оператори центрів обробки даних.

Huawei Agile Controller є основним компонентом програмно-визначеної мережі (SDN). Його можна застосувати до чотирьох бізнес-сценаріїв: центрів обробки даних(мережевих), кампус, глобальна мережа та мережі Інтернету. Місія AC полягає в тому, щоб забезпечити автоматичний переклад даних із додатків у фізичну мережу, повний контроль над мережевими ресурсами та видимість функцій операційної та технічної підтримки. AC базується на відкритих технологіях і пропонує максимально широкі можливості для сумісності зі сторонніми продуктами та створення комплексних рішень на їх основі.

Візуальне зображення переходу від звичайної мережі ЦОД до SDN зображено на Рис. 1.4:

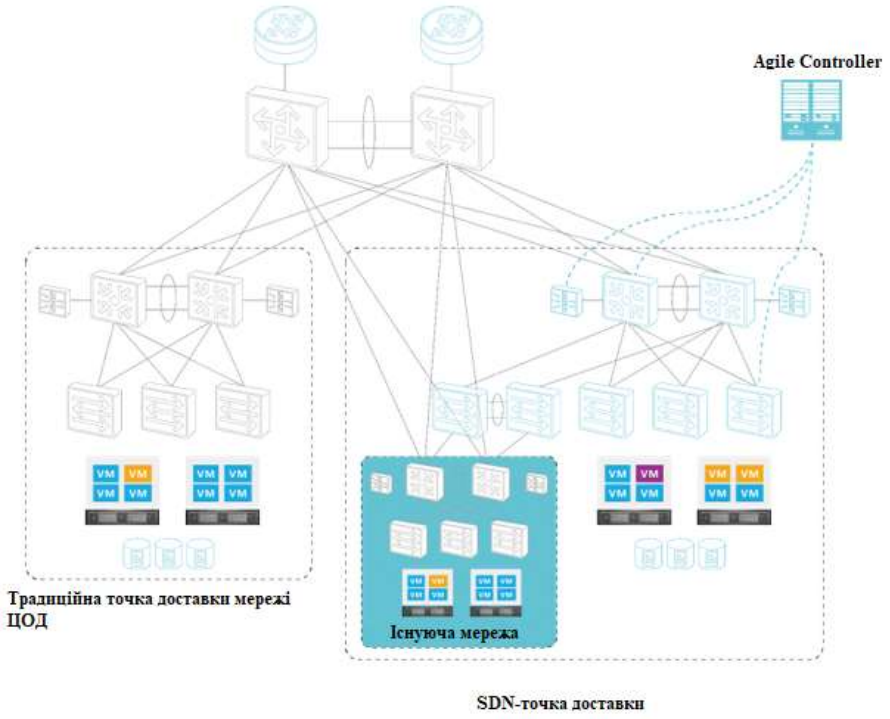

Рисунок 1.4 - Перехід від звичайної мережі ЦОД до SDN

Не так давно віртуалізація SDN і мережеві функції (NFV) посіли центральне місце в мережах операторів мобільного зв'язку та інших постачальників послуг Проблеми, які вирішуються SDN і NFV, включають відокремлення логіки програми та реалізації від відповідних даних про абонента. Технологія вирішує проблему двома способами – за допомогою динамічної оркестровки ресурсів і інтелектуальної оркестровки послуг. Для динамічної оркестровки ресурсів із використанням NFV потрібен єдиний стек віртуалізації для всіх програм, як правило, у приватному хмарному середовищі оператора.

Висновки по розділу

У першому розділі була розглянута історія появи мережевої технології SDN, а саме те що незважаючи на досить недавнє виникнення, програмно-конфігуровані мережі SDN (Software Defined Network), мають досить тривалу передісторію. Програмно-конфігурована мережа (SDN) змінила підхід до проектування та управління мережею дозволивши використовувати спрощене управління та введення нових послуг.

Під час розгляду архітектури SDN було визначено, три рівні архітектури , кожен з яких складається з різних компонентів. Прикладний рівень ,рівень контрольної площини та рівень інфраструктури або даних.

Проаналізувавши типи мережі SDN ,було розглянуто наступні моделі: модель накладання SDN , відкрита мережа SDN та SDN з API-інтерфейсами.Було досліджено відмінності та переваги мережі SDN від традиційної.

Дослідження варіантів використання SDN мереж показало що: SDN-мережі мають застосування у найрізноманітніших галузях такі як: передавання даних ,мобільні послуги та мережеві станції обробки даних.

РОЗДІЛ 2

РОБОЧЕ СЕРЕДОВИЩЕ ДЛЯ МОДЕЛЮВАННЯ SDN МЕРЕЖ

2.1 Огляд віртуальної машини для реалізації роботи мереж

Щоб емулювати мережу SDN на комп'ютері з обмеженими програмними ресурсами, необхідно використовувати віртуальні машини та емулятори з відкритим кодом. Щоб забезпечити дослідження та моделювання SDN-хостів, контролерів і комутаторів за допомогою OpenFlow, нам потрібне таке програмне забезпечення, як: Oracle VM VirtualBox під керуванням Mininet, яке включає додаткові модулі MiniEdit, Xterm і Wireshark.

Oracle VM VirtualBox — це програма, що дозволяє запустити на вашому ПК деякі операційні системи віртуально. За допомогою цієї програми з'являються можливості використовувати інші операційні системи віртуально. VirtualBox дозволяє працювати з FreeBSD , Linux, ReactOS, Solaris або OpenSolaris, Mac OS та DOS.На рисунку 1.5 зображено початкове вікно Oracle VM VirtualBox.

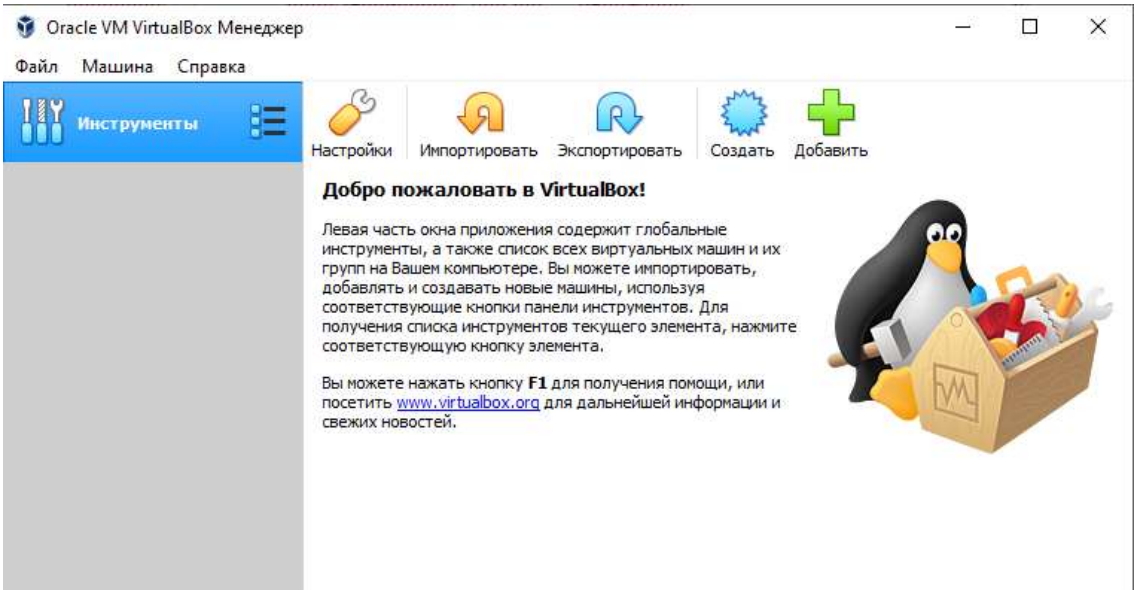

Рисунок 2.1 - Початкове вікно Oracle VM VirtualBox

.

Переваги програми:

- Перспектива навчатись роботі з різними ОС;
- Вільний та безкоштовний доступ;
- Практична для використання;
- Мережевий стек на просунотому рівні;
- Наявна сумісність з DirectX, OpenGL;
- Легке використання командного рядка для автоматичного розгортання;
- Підтримка віртуального USB-контролера, який приєднується до віртуальної машини через USB 1.1, USB 2.0;
- Наявна можливість приєднання віддалено — через вбудований RDP-сервер;
- Має підтримку різновидів мереж ,таких як: NAT, Internal, Host Networking via Bridged;
- Можливість легко змінити мову інтерфейсу
- Shared Folders дозволяє передавати файли між гостьовою до хостовою системами та навпаки;
- Можна запуску двох або більше віртуальних машин одночасно
- Наявна портативна версія.

Програма має багато переваг, зокрема простоту установки, користування та багатофункціональність. Проте, недоліками вважають дещо заплутане управління дисками та відсутність підтримки скріншотів.

Віртуалізація виявляється корисною для вивчення різних операційних систем, оскільки вона забезпечує більш зручний підхід порівняно з безпосереднім встановленням на фізичний комп'ютер. Раніше користувачі зазвичай вдавалися до подвійного завантаження, коли хотіли запустити кілька операційних систем на своєму комп'ютері, наприклад, Linux і Windows на одному комп'ютері. Однак, подвійне завантаження часто вимагало розбиття жорсткого диска на розділи і перезавантаження комп'ютера кожного разу для перемикання між операційними системами. Щоб зрозуміти роботу програми Oracle VM VirtualBox та її конфігурацію, дуже важливо розуміти її функціональність.

VirtualBox також корисний для тих, хто хоче отримати більше досвіду в конфігуруванні мережі. Він дозволяє встановлювати кілька операційних систем одночасно, навіть різних, і забезпечує гнучкість у налаштуванні та роботі з ними, не впливаючи на операційну систему хоста. Це дозволяє користувачам вивчати комп'ютерні технології, практикуватись і набувати нових навичок.

2.2 Дослідження емулятора Mininet .

Mininet - це. Емулятор комп'ютерної мережі під якою маються на увазі прості системні прилажи - хости, комутатори, а також OpenFlow-контролери. За допомогою найпростішого синтаксису в примітивному інтерпретаторі команд можна розгортати мережі з довільної кількості хостів, комутаторів у різних топологіях, і все це в рамках однієї віртуальної машини (ВМ). На всіх хостах можна змінювати мережеву конфігурацію, користуватися стандартними утилітамиі навіть отримувати доступ до терміналу. На комутатори можна додавати різні правила і маршрутизувати трафік. Загалом Mininet дає змогу познайомитися з будовою і функціонуванням комп'ютерних мереж без необхідності використання будь-якого мережевого обладнання.

Mininet підтримує дослідження, розробки, навчання, створення прототипів,тестування, налагодження та будь-які інші завдання, які можуть виникнути

Дає можливість користуватися перевагами повної лабораторної мережі на своєму ноутбуці чи іншому ПК.

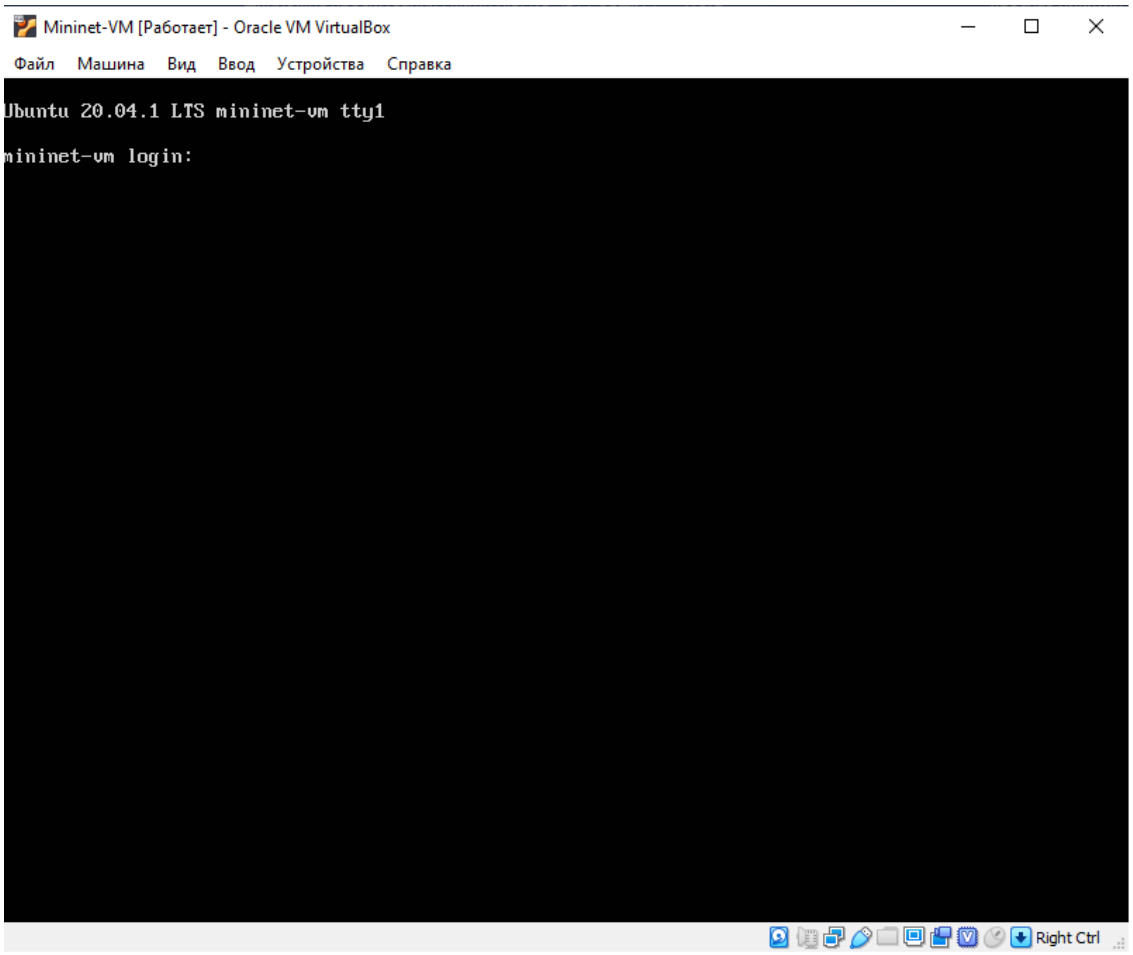

Рисунок 2.2 - Початкове вікно Mininet

Mininet має наступні можливості:

- Надає просте та недороге середовище розробки для програми OpenFlow
- Дозволяє кільком розробникам одночасно працювати над однією або декількох топологіях
- Підтримка відтворюваного регресійного тестування на рівні системи ,які можуть повторюватися та легко упаковуватись.
- Пропонує можливість тестувати складних топологій без наявного фізичного підключення до мережі
- Включає інтерфейс командного рядка з підтримкою OpenFlow для налагодження або реалізації тестів мережі.
- Включає в себе базовий набір локальних системних топологій , а також має змогу взаємодіяти з доволі спеціальними топологіями.
- У своєму розпорядженні має API Python для створення мереж і експериментів з ними.

Mininet забезпечує простий спосіб отримання правильну підтримку функціональності і продуктивності системи ,а також дає можливість експериментувати з топологіями.

ПЗ Mininet відкриває код, до якого входять стандартні програми мережування Unix/Linux або Windows, а також реальне ядро і мережевий стек

Це дозволяє вам перенести код, який ви розробляєте та тестуєте в Mininet для контролерів OpenFlow, модифікованих комутаторів або хостів, в реальні системи з мінімальними змінами для тестування. Варто підкреслити, що такий підхід дозволяє проектам, що працюють на Mininet, плавно перейти на апаратні комутатори, забезпечуючи швидку передачу пакетів на рівні фізичного з'єднання.

Майже всі операційні системи використовують віртуалізацію обчислювальних ресурсів через абстракцію процесів.

Mininet включає в себе різноманітні емулятори, апаратні тестові панелі та тренажери з великим набором функцій, що перевершують підходи, що працюють на повну віртуалізацію системи. Деякі приклади цих інструментів включають:

- Завантаження відбувається дуже швидко, витрачаючи всього кілька секунд;
- Можливість масштабування до сотень хостів і комутаторів;
- Забезпечує високу пропускну здатність;
- Легка установка - достатньо мати встановлену віртуальну машину, таку як VMware або VirtualBox для Mac/Win/Linux, з вже наявними інструментами OpenFlow v1.0.

У порівнянні з апаратними тестовими панелями, Mininet має такі переваги:

- Суттєво нижча вартість і легка доступність;
- Швидше встановлення налаштувань і перезавантаження.

Mininet виділяється серед інших емуляторів завдяки своїм особливостям:

- Він використовує справжній, незмінений код, що включає програмний код, код ядра ОС, код середовища конфігурації та управління, а також код контролера OpenFlow.
- Він пропонує прості засоби підключення до реальних мереж.

2.3 Огляд та налаштування графічного інтерфейсу Miniedit

MiniEdit

У Mininet, мережевому емуляторі, доступний графічний інтерфейс, який дозволяє користувачам візуально створювати різні топології мереж SDN. За допомогою Miniedit користувачі можуть спочатку розробити графічне представлення топології мережі, демонструючи структуру і розширені налаштування активних компонентів і мережевих з'єднань. Після того, як мережа SDN побудована і необхідні елементи налаштовані, проект може бути збережений у вигляді файлу .mn. Крім того, Miniedit дозволяє користувачам безпосередньо запускати мережу з графічного інтерфейсу. Після вивчення продуктивності та параметрів мережі, користувачі можуть повернутися до етапу конфігурації та редагування віртуального макету. Однією з помітних переваг Miniedit є можливість імпортувати макет мережі у файл Python з відповідною конфігурацією мережевих елементів.

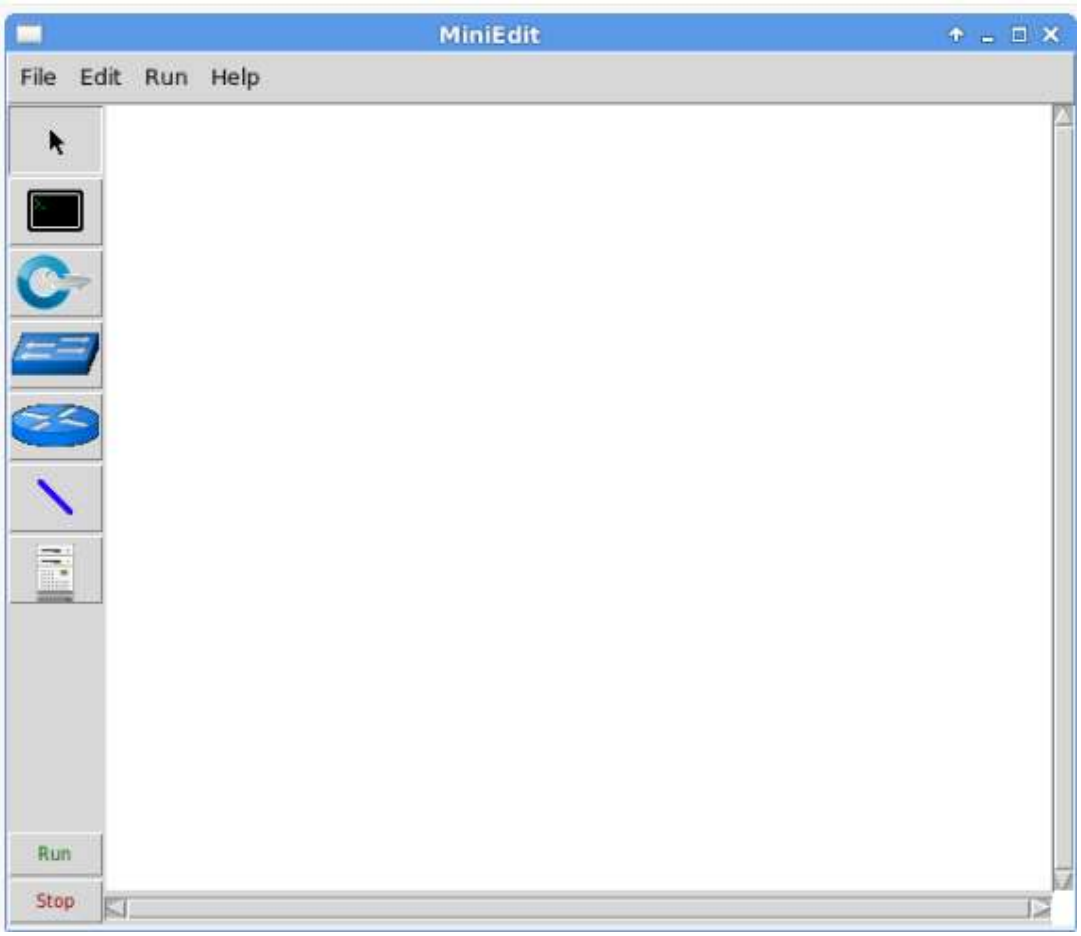

Рисунок 2.3 - Початкове вікно MiniEdit

За замовчуванням MiniEdit генерує еталонний контролер mininet OpenFlow, який емулює функціональність і можливості навчання комутатора. Отже, технологія Mininet пропонує зручне рішення для створення прототипів та моделювання. Вона слугує економічно вигідною альтернативою тестуванню реальних мереж. Створення віртуальної мережі в Mininet дозволяє швидко тестувати різні конфігурації системи. MiniEdit покращує розуміння мережевих процесів і пропонує зручний функціонал для проведення мережевих досліджень.

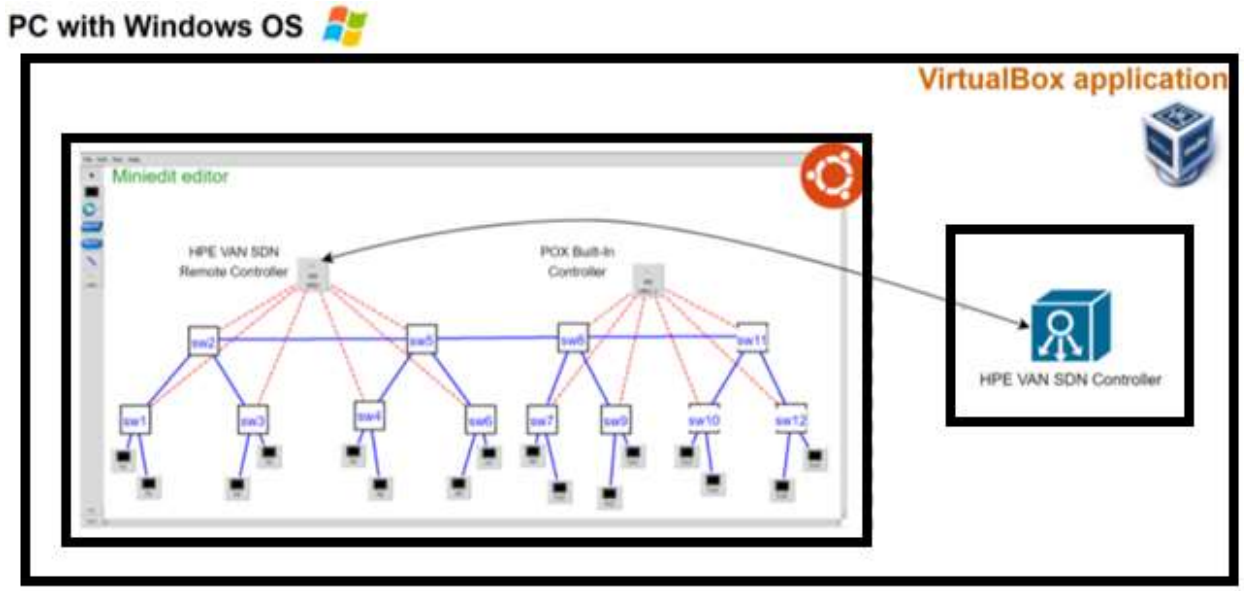

Рисунок 2.4 - Структурна схема мережі SDN на базі Mininet з використанням

Для запуску та налаштування MiniEdit потрібно ввести відповідну команду з правами root у середовищі Mininet: .

```
mininet@mininet-vm:~$ sudo ~/mininet/examples/miniedit.py
```

Рисунок 2.5 – Код для запуску MiniEdit у середовищі Mininet

Є можливість виникнення помилок ,тому можна відкрити безпосередньо через управління файлами :

```
mininet@mininet-vm:~$ cd mininet
mininet@mininet-vm:~/mininet$ cd mininet
mininet@mininet-vm:~/mininet/mininet$ cd examples
mininet@mininet-vm:~/mininet/mininet/examples$ ls_
```

Рисунок 2.6 - Команди аналогічного запуску MiniEdit

На рисунку 2.7 можна побачити наявні додатки які розташовані у каталозі examples.

```
mininet@mininet-vm:~$ cd mininet
mininet@mininet-vm:~/mininet$ cd mininet
mininet@mininet-vm:~/mininet/mininet$ cd examples
mininet@mininet-vm:~/mininet/mininet/examples$ ls
baresshd.py       controllers2.py  limit.py           multitest.py      scratchnet.py
bind.py           controllers.py   linearbandwidth.py natnet.py         scratchnetuser.py
clustercli.py     controlnet.py    linuxrouter.py     nat.py            simpleperf.py
clusterdemo.py    cpu.py           miniedit.py        numberedports.py  sshd.py
clusterperf.py    emptynet.py      mobility.py        popenpoll.py      test
cluster.py        hwintf.py        multilink.py       popen.py          tree1024.py
clusterSanity.py  __init__.py      multiping.py       __pycache__       treeping64.py
consoles.py       intfoptions.py   multipoll.py       README.md         vlanhost.py
```

Рисунок 2.7 – Каталог examples

Після огляду додатків потрібно ввести наступну команду для запуску програми, яку зображено на рисунку 2.8.

```
mininet@mininet-vm:~/mininet/mininet/examples$ sudo ./miniedit.py
```

Рисунок 2.8 – Команда запуску MiniEdit.py

Після запуску MiniEdit потрібно перейти у Налаштування ,після чого відкриється діалогове вікно

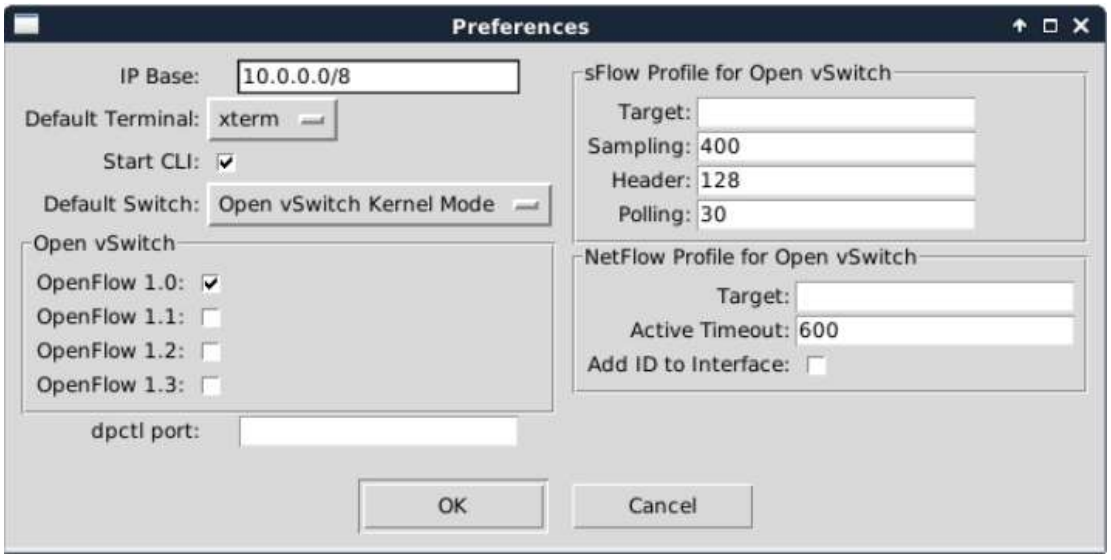

Рисунок 2.9 - Діалогове вікно налаштувань

Необхідно поставити прапорець навпроти «Start CLI» це надасть доступ до інтерфейсу командного рядка. Також потрібно обрати версію OpenFlow.

Всі проведені налаштування забезпечують створення програмних сценаріїв створюваної мережі, які дозволять хостам взаємодіяти один з одним.

Параметри зберігаються окремо, тому є можливість вказати різні значення для кожної топології.

2.4.Аналіз додатків необхідних для емуляції мережі

Xterm

Xterm - широко розповсюджений емулятор терміналу, розроблений для середовища Unix у віконній системі X Window System. Він дозволяє користувачам одночасно працювати з декількома терміналами xterm на одному дисплеї. Кожен віртуальний термінал надає незалежні можливості вводу і виводу для процесів, запущених у ньому, як правило, процесів оболонки Unix.

Існують різні варіанти xterm, причому багато емуляторів терміналів для X спочатку створюються як похідні від xterm. За замовчуванням xterm не містить рядка меню. Щоб отримати доступ до одного з трьох меню xterm, користувачеві потрібно утримувати клавішу Control і клацнути лівою, середньою або правою кнопкою миші. Втім, підтримку рядка меню можна увімкнути під час компіляції, у результаті чого згадані вище меню буде доступно з рядка меню.

PuTTY

PuTTY - це безкоштовне програмне забезпечення, яке полегшує підключення до серверів за допомогою мережевих протоколів Ці протоколи дозволяють ініціювати віддалені сеанси на комп'ютері, а PuTTY слугує програмою на стороні клієнта для таких сеансів.

З технічної точки зору, PuTTY працює як емулятор терміналу, імітуючи поведінку терміналу, який слугує пристроєм вводу/виводу для користувачів, що підключаються до віддалених серверів. Простими словами, коли ви запускаєте PuTTY на своєму комп'ютері і встановлюєте з'єднання з сервером, ви отримуєте можливість вводити команди, які виконуються на сервері. Відповіді та вивід з сервера відображаються у вікні, що дозволяє віддалено адмініструвати сервер.

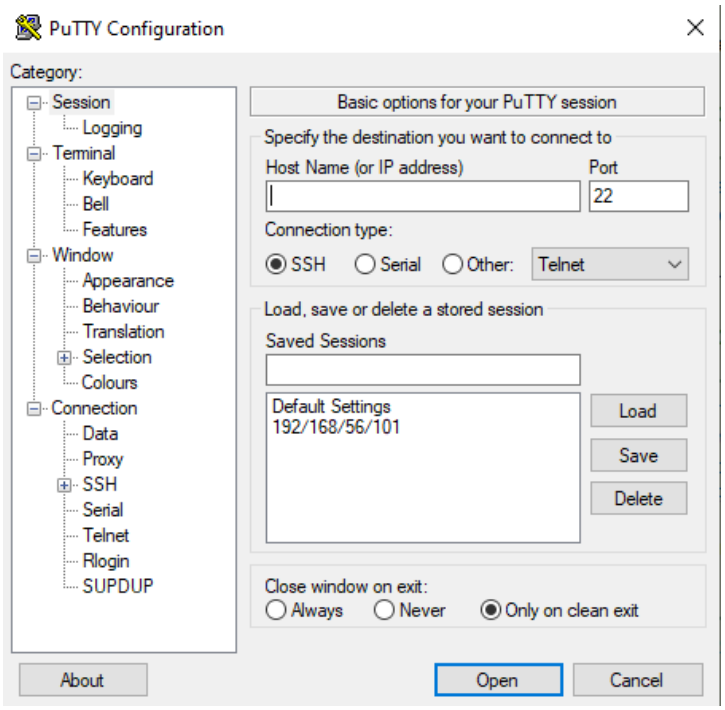

Рисунок 2.10 - Вікно налаштування PuTTY

PuTTY - це програмне забезпечення, призначене для комп'ютерів з Windows, яке також включає в себе емулятор терміналу xterm. Вона виявляється корисною, коли вам потрібно підключитися до Unix-сервера. Програма пропонує кілька функцій, серед яких

- Налаштування шрифтів, кольорів і роздільної здатності консолі.
- Функція логування для ведення записів про сеанси.
- Можливість збереження ключів авторизації для спрощеної аутентифікації.
- Підтримка роботи через проксі-сервер.
- Можливості передачі файлів.

Wireshark.

Wireshark - це вільно розповсюджуване програмне забезпечення з відкритим вихідним кодом, яке дозволяє аналізувати мережеві пакети в Ethernet та інших мережах. Вона надає для цього зручний графічний інтерфейс.

Wireshark пропонує функціональність, подібну до tcpdump, але відрізняється графічним інтерфейсом користувача і розширеними можливостями сортування і фільтрації. Увімкнувши режим проміскуїтету на мережевій карті, програма дозволяє користувачам спостерігати за трафіком, що проходить через мережу в

реальному часі, надаючи повний огляд всіх даних.

Wireshark - це програмне забезпечення, яке здатне ідентифікувати та інтерпретувати структуру різноманітних мережевих протоколів. Таким чином, вона дозволяє користувачам аналізувати мережеві пакети, відображаючи значення окремих полів протоколу на різних рівнях. Для перехоплення пакетів програма використовує pcap, що обмежує її можливості мережами, які підтримуються цією бібліотекою. Тим не менш, Wireshark сумісний з різними форматами вихідних даних, що дозволяє користувачам відкривати файли даних, перехоплених іншими програмами, тим самим розширюючи його можливості перехоплення.

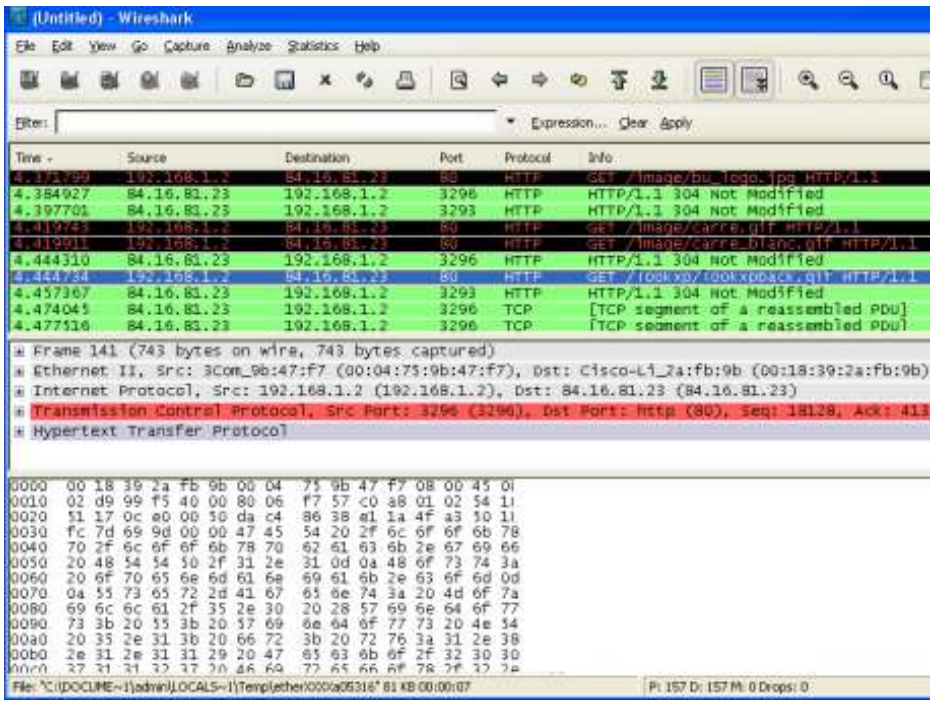

Рисунок 2.11 - Головне вікно програми Wireshark

Програма Wireshark поширюється під ліцензією GNU GPL і використовує крос-платформну бібліотеку GTK+ для створення графічного інтерфейсу користувача. Вона доступна у версіях, сумісних з різними операційними системами на базі UNIX, такими як GNU/Linux, Solaris, FreeBSD, NetBSD, OpenBSD, а також Mac OS X і Microsoft Windows.

Висновоки до розділу:

В данному розділі було проведено аналіз віртуальної машини для реалізації роботи мережі. Виявлено, що використання віртуальних машин є ефективним і зручним способом для моделювання та тестування мережних сценаріїв. Віртуальна машина надає можливість створювати віртуальні мережі з різними хостами, комутаторами та з'єднаннями між ними.

Досліджено емулятор Mininet, який є потужним інструментом для створення віртуальних мереж у середовищі SDN. Mininet дозволяє моделювати різні мережні топології та виконувати експерименти з протоколами та алгоритмами маршрутизації. Використання Mininet дозволяє реалістично емулювати мережні умови та досліджувати їх ефективність.

Також проведено огляд та налаштування графічного інтерфейсу Miniedit, який надає зручну інтерактивну засіб для візуалізації та налаштування мережевих топологій. Графічний інтерфейс спрощує процес створення та налаштування мереж, дозволяючи легко додавати та змінювати вузли, з'єднання та параметри мережі.

Також було проаналізовано необхідні додатки для емуляції мережі, такі як Xterm, PuTTY та Wireshark. Використання відповідних додатків є ключовим фактором для успішної емуляції та дослідження мережних сценаріїв.

В цілому, проведений аналіз показав, що використання віртуальних машин, емулятора Mininet та додатків для емуляції мережі є потужними та ефективними інструментами для дослідження комп'ютерних мереж на основі маршрутизації SDN. Вони надають зручний спосіб моделювання та тестування мережних сценаріїв, дозволяючи здійснювати дослідження та вдосконалення протоколів та алгоритмів маршрутизації.

РОЗДІЛ 3.

МЕТОДИ СТВОРЕННЯ ТА ЗАСТОСУВАННЯ МЕРЕЖЕВИХ ТОПОЛОГІЙ SDN З ВИКОРИСТАННЯМ MININET.

3.1 Дослідження топологій мережі з використанням Mininet.

Топології у середовищі Mininet дозволяють моделювати різноманітні мережні сценарії та проводити випробування протоколів та алгоритмів. Кожна з цих топологій має свої унікальні особливості і може бути використана відповідно до конкретних потреб дослідження або експерименту. Вибір певної топології залежатеме від того, які аспекти мережі потрібно дослідити Знання про ці топології допоможе ефективно використовувати середовище Mininet для будь яких досліджень.

Однорівнева топологія:

Це найпростіший тип топології, де один комутатор підключений до декількох хостів. Хости безпосередньо підключені до комутатора, створюючи прямі з'єднання між собою. Цей тип топології підходить для базових експериментів, навчання та досліджень

```
mininet@mininet-vm:~$ cd mininet
mininet@mininet-vm:~/mininet$ cd examples
mininet@mininet-vm:~/mininet/examples$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

Рисунок 3.1 – Команда створення однорівневої топології

Після запуску Mininet з командою `sudo mn` для тестування створеної топології можна використовувати такі команди:

- “pingall”: Виконує ping між всіма хостами в топології для перевірки зв'язку.

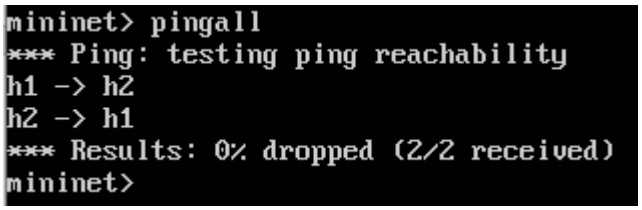

Рисунок 3.2 – Команда “pingall”

- “iperf”: Виконує тестування пропускної здатності мережі за допомогою інструменту iperf.

Рисунок 3.3 – Команда “iperf”

- “net.pingAll()”: Виконує ping між всіма хостами в топології.

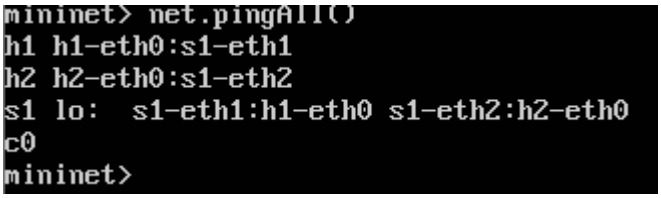

Рисунок 3.4 – Командв “net.pingAll()”

- “net.iperf()”: Виконує тестування пропускної здатності мережі за допомогою інструменту iperf.

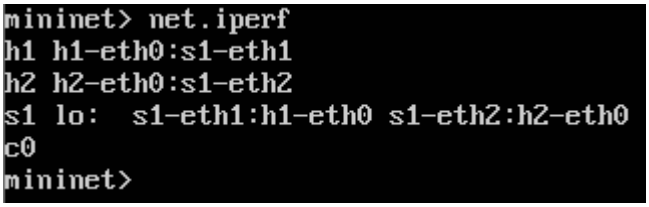

Рисунок 3.5 – Команда “net.iperf()”

Ці команди дозволяють перевірити зв'язок та виконати тестування пропускної здатності мережі в створеній топології Mininet. Залежно від потреб, можливо також використовувати додаткові команди та інструменти для тестування та аналізу топології.

Деревоподібна топологія:

У цій топології комутатори організовані в деревоподібну структуру, де кореневий комутатор з'єднаний з підкомутаторами, а хости з'єднані з підкомутаторами.

Цей тип топології дозволяє створювати багаторівневі мережі з розширеними комунікаційними можливостями.

Хости підключаються до субкомутаторів, забезпечуючи гнучкість у налаштуванні рівнів зв'язку між ними.

Команда для створення деревоподібної топології в середовищі Mininet - `sudo mn --topo tree`. Ця команда створює деревоподібну мережу з за замовчуванням чотирма рівнями комутаторів і по одному хосту, підключеному до кожного комутатора.

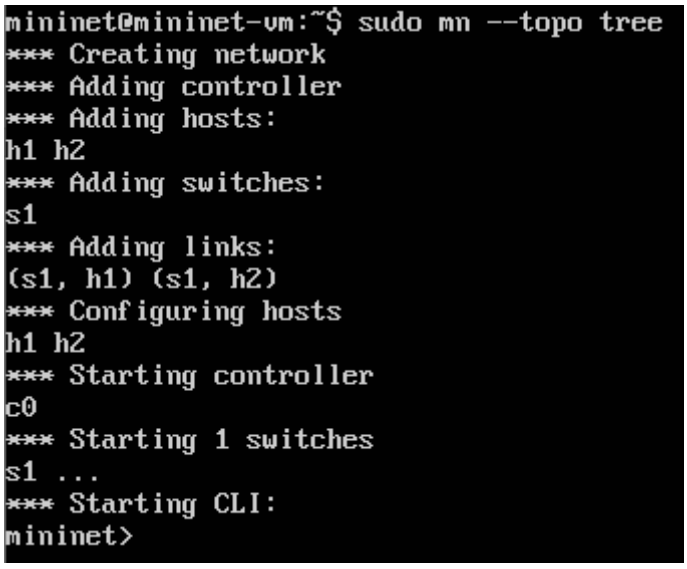

Рисунок 3.6 - Команда для створення деревоподібної топології

Можливо також вказати власні параметри для кількості рівнів та хостів, використовуючи параметри командного рядка. Наприклад, `sudo mn --topo tree,depth=3,fanout=2` створить деревоподібну топологію з трьома рівнями та коефіцієнтом розгалуження 2.

Топологія “linear”

Топологія “linear” представляє собою просту лінійну мережу, де хости та комутатори з'єднані послідовно у ланцюжок. Кожен хост підключений до наступного хоста або комутатора у цьому ланцюжку. Така топологія може бути корисною для дослідження основних концепцій мережевого з'єднання та пересилання даних.

Ось як виглядає команда для створення лінійної топології з 2 хостами та 2 комутаторами:

```
mininet@mininet-vm:~$ sudo mn --topo=linear,2,2
*** Creating network
*** Adding controller
*** Adding hosts:
h1s1 h1s2 h2s1 h2s2
*** Adding switches:
s1 s2
*** Adding links:
(h1s1, s1) (h1s2, s2) (h2s1, s1) (h2s2, s2) (s2, s1)
*** Configuring hosts
h1s1 h1s2 h2s1 h2s2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Starting CLI:
```

Рисунок 3.7 - Команда для створення лінійної топології

У цій команді --topo linear вказує, на створення лінійної топології. Параметри 2 та 2 вказують відповідно кількість хостів та комутаторів у мережі.

У лінійній топології всі хости та комутатори розташовані в одному ряду, і кожен з них має пряме з'єднання лише зі своїм попереднім та наступним вузлом у ланцюжку. Це означає, що дані, які надсилаються від одного вузла до іншого, проходять через всі проміжні вузли.

В лінійній топології немає замкненого кола, що дозволяє просте розуміння маршрутизації даних. Кожен пакет даних буде передаватись послідовно від одного вузла до наступного, поки не досягне призначеного вузла. Така простота дозволяє легко вивчати і тестувати механізми мережевого з'єднання та пересилання даних, а також досліджувати проблеми затримок та пропускної здатності.

Таким чином, лінійна топологія в Mininet є простим, послідовним з'єднанням хостів та комутаторів у ланцюжку, що дозволяє вивчати та експериментувати з основними концепціями мережевого з'єднання та пересилання даних.

Топологія "reversed"

Топологія "reversed" в Mininet є варіацією лінійної топології, де порядок з'єднання комутаторів та хостів є оберненим. Замість того, щоб з'єднувати перший комутатор з другим, другий з третім і так далі, у зворотній топології з'єднання відбуваються в зворотному порядку.

Така топологія може бути корисною для тестування мережних протоколів, роутингу, алгоритмів балансування навантаження та інших сценаріїв, де порядок з'єднання є важливим.

Команда для створення топології “reversed” у Mininet виглядає наступним чином:

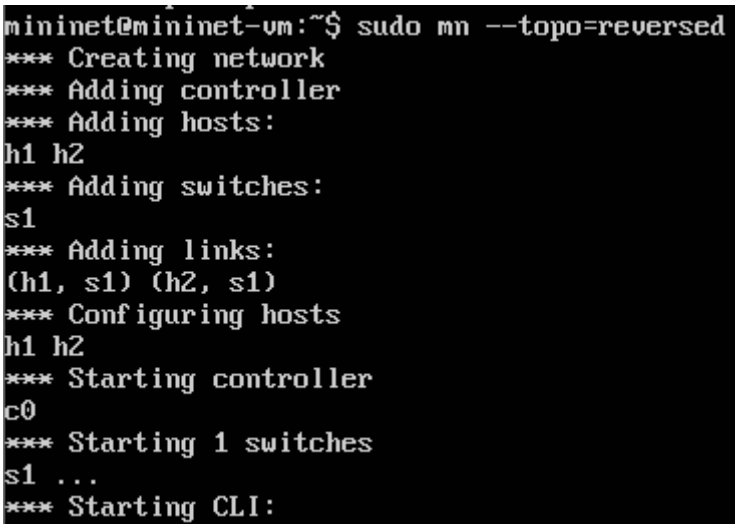

Рисунок 3.8 - Команда для створення топології “reversed”

Така топологія може бути корисною для тестування мережних протоколів, роутингу, алгоритмів балансування навантаження та інших сценаріїв, де порядок з'єднання є важливим

Це лише короткі пояснення кожного типу топології, які можна створити в середовищі Mininet. Використовуючи ці типи топології, ви можете експериментувати з різними мережевими сценаріями і вивчати їх вплив на взаємодію і продуктивність мережі.

У Mininet для показу існуючих топологій використовуються наступні команди:

- net або net.get() - ця команда повертає об'єкт, який представляє поточну мережу Mininet.
- net.topo - ця команда повертає об'єкт, який представляє топологію мережі, яку ви створили.
- net.topo.nodes() - ця команда повертає список вузлів (хостів та комутаторів) у поточній топології.
- net.topo.links() - ця команда повертає список з'єднань (лінків) у поточній топології.
- net.topo.get_hosts() - ця команда повертає список хостів у поточній топології.
- net.topo.get_switches() - ця команда повертає список комутаторів у поточній топології.
- net.topo.get_links() - ця команда повертає список з'єднань (лінків) у поточній топології.

Використовуючи ці команди, можна отримати доступ до інформації про існуючі топології в середовищі Mininet.

3.2 Створення мережевої топології у середовищі Mininet

Для створення простої мережі з одним комутатором, підключеним до кількох хостів, використовуючи бібліотеку Mininet, потрібно створити власний клас топології та використати його для створення мережі.

Цей код призначений для створення простої мережі з одним комутатором, підключеним до кількох хостів, використовуючи бібліотеку Mininet. Він виконує наступні дії:

- Визначає клас SingleSwitchTopo, який успадковує від класу Topo і будує топологію з одним комутатором, що підключений до n хостів.

- Визначає функцію simpleTest, яка створює топологію SingleSwitchTopo, запускає мережу, виводить інформацію про стан підключення хостів та виконує тест зв'язку між ними.
- Виконує функцію simpleTest, якщо файл виконується безпосередньо, а не імпортується як модуль.
- Встановлює рівень журналювання на рівень info.
- Викликає функцію simpleTest.

Отже, загальне призначення цього коду - створення та тестування простої мережі з одним комутатором та кількома хостами за допомогою Mininet. На рисунку 3.9 зображено код простої мережі:

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class SingleSwitchTopo(Topo):
    "Один комутатор, підключений до n хостів."
    def build(self, n=2):
        switch = self.addSwitch('s1')
        # Діапазон від 0 до N-1
        for h in range(n):
            host = self.addHost('h%s' % (h + 1))
            self.addLink(host, switch)

def simpleTest():
    "Створення та тестування простої мережі"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print("Вивід інформації про стан підключення хостів")
    dumpNodeConnections(net.hosts)
    print("Тест зв'язку у мережі")
    net.pingAll()
    net.stop()

if __name__ == '__main__':
    # Вивід інформації
    setLogLevel('info')
    simpleTest()
```

Рисунок 3.9 - Код для створення простої мережі

Щоб запустити створений код у файлі з розширенням .py у середовищі Mininet, знадобиться виконати наступні кроки:

Відкрити термінал або командний рядок на комп'ютері.

Перейти до теки, де знаходиться файл з кодом Python.

Запустити файл з кодом, використовуючи команду python simplenet.py.

Зачекати, поки виконання коду завершиться. На екрані з’явиться вивід, який відображатиме інформацію про стан підключень хостів та результати тесту зв'язку у мережі.

```
$ sudo python simplenet.py
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 1 switches
s1 ...
Dumping host connections
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth3
h4 h4-eth0:s1-eth4
Testing network connectivity
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
*** Stopping 1 controllers
c0
*** Stopping 4 links
....
*** Stopping 1 switches
s1
*** Stopping 4 hosts
h1 h2 h3 h4
*** Done
```

Рисунок 3.10 – Результат виконання коду

Створимо інший приклад з власною конфігурацією мережі. Ця модель буде мати 4 хости і 2 комутатори, які будуть з'єднані між собою.

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class CustomTopology(Topo):
    def build(self):
        switch1 = self.addSwitch('s1')
        switch2 = self.addSwitch('s2')

        host1 = self.addHost('h1')
        host2 = self.addHost('h2')
        host3 = self.addHost('h3')
        host4 = self.addHost('h4')

        self.addLink(host1, switch1)
        self.addLink(host2, switch1)
        self.addLink(host3, switch2)
        self.addLink(host4, switch2)
        self.addLink(switch1, switch2)

def run_custom_topology():
    topo = CustomTopology()
    net = Mininet(topo)
    net.start()

    print("Вивід інформації про стан підключення хостів")
    dumpNodeConnections(net.hosts)

    net.pingAll()

    net.stop()

if __name__ == '__main__':
    setLogLevel('info')
    run_custom_topology()
```

Рисунок 3.11 – Код для створення мережі(4 хости і 2 комутатори)

У цьому прикладі створюється клас CustomTopology, який успадковує клас Topo і визначає метод build() для побудови топології. У цьому методі використовуються 2 комутатори (switch1 і switch2) і 4 хости (host1, host2, host3, host4). З'єднання між комутаторами створюється за допомогою останнього виклику addLink().

У функції run_custom_topology() потрібно створити екземпляр топології, створюємо мережу Mininet з цією топологією і запускаємо мережу. Також потрібно вивести інформацію про стан підключення хостів,після виконання пінгування між

всіма хостами (net.pingAll()) зупиняємо мережу (net.stop()).

Потрібно зберегти цей код у файл з розширенням .py і запустити його в середовищі Mininet, використовуючи команду sudo python simple.py

```
$ sudo python simple.py
***Creating network
***Adding controller
***Adding hosts:
h1 h2 h3 h4
***Adding switches:
s1 s2
***Adding links:
(h1, s1) (h2, s1) (h3, s2) (h4, s2) (s1, s2)
***Configuring hosts
h1 h2 h3 h4
***Starting controller
c0
***Starting 2 switches
s1 s2
***Starting CLI:
mininet>
```

Рисунок 3.12 - Результат виконання коду

Після запуску мережі всі хости будуть мати доступ до своїх відповідних комутаторів, а комутатори будуть об’єднані між собою.

Таким чином, було створено мережу з 4 хостами (h1, h2, h3, h4) і 2 комутаторами (s1, s2). Комутатори об'єднані між собою, а хости підключені до відповідних комутаторів

На рисунку 3.13 зображено загальний вигляд створеної топології.

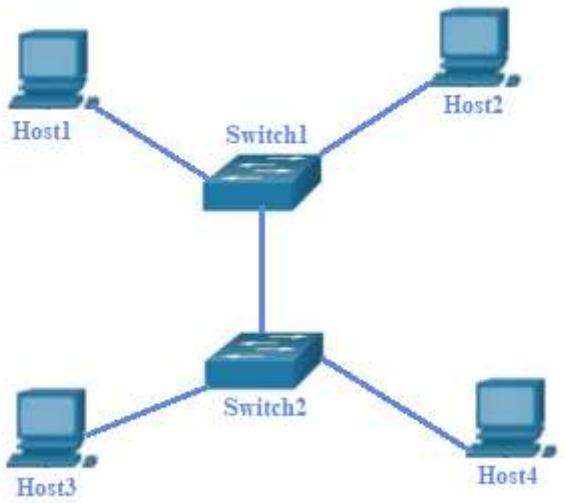

Рисунок 3.13 - Загальний вигляд створеної топології

Висновки до розділу:

Дослідження топологій мережі з використанням Mininet є важливим аспектом розробки, тестування та валідації мережевих рішень. За допомогою Mininet можна легко створювати віртуальні мережі, що дозволяє експериментувати з різними конфігураціями та перевіряти роботу мережевих протоколів та алгоритмів.

У розділі "Створення мережевої топології у середовищі Mininet" було наведено приклад створення мережі з 4 хостами і 2 комутаторами. Використовуючи функції Mininet, можливо легко створювати вузли мережі, підключати їх один до одного та встановлювати з'єднання з різними характеристиками.

Проведення досліджень топологій мережі з використанням Mininet дозволяє проаналізувати ефективність розподілу трафіку, перевіряти протоколи маршрутизації та комутації, оцінювати навантаження на різних вузлах мережі та багато іншого. Mininet також надає зручні інструменти для візуалізації топології мережі та моніторингу мережевих показників, що сприяє глибокому розумінню роботи мережевих систем.

Загалом, дослідження топологій мережі з використанням Mininet є потужним інструментом для розробників, мережевих інженерів та дослідників, що дозволяє вивчати, тестувати та оптимізувати мережеві рішення перед їх впровадженням у реальній мережі.

ВИСНОВКИ

Дипломний проєкт присвячений аналізу напрямків розвитку комп'ютерних мереж на базі SDN (Software-Defined Networking) та розробці робочого середовища для моделювання SDN мереж з використанням Mininet.

У першому розділі було проведено аналіз історії появи мережевої технології SDN, розглянуто архітектуру цієї технології та вивчено різні варіанти її використання в мережевих системах.

У другому розділі було проведено аналіз віртуальної машини, що використовується для реалізації роботи мережі, а також досліджено емулятор Mininet, який є потужним інструментом для моделювання мереж. Було також оглянуто та налаштовано графічний інтерфейс Miniedit та проаналізовано необхідні додатки для емуляції системи.

Третій розділ присвячено методам створення та застосуванню мережевих топологій SDN з використанням Mininet. Було проведено дослідження різних топологій мережі та описано процес створення мережевої топології у середовищі Mininet.

У загальному, цей дипломний проєкт надає глибокий аналіз технології SDN та її застосування у комп'ютерних мережах. Розроблене робоче середовище на базі Mininet дозволяє ефективно моделювати та тестувати мережеві рішення, що дозволяє виконувати належну перевірку та оптимізацію перед їх реалізацією у реальних мережах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Приклади топології Mininet [Електронний ресурс] – Режим доступу до ресурсу:https://medium.com/@abdulkaderhajjouz/mininet-topology-examples 62ae4c9d8168

2. Покрокове керівництво Mininet [Електронний ресурс] – Режим доступу до ресурсу: http://mininet.org/walkthrough/#ssh-daemon-per-host

3. Признаяення PuTTY [Електронний ресурс] – Режим доступу до ресурсу:https://tuthost.ua/uk/blog/chto-takoe-putty/

4. Використання MiniEdit , як графічним інтерфейсом користувача Mininet [Електронний ресурс] – Режим доступу до ресурсу: https://www.brianlinkletter.com/2015/04/how-to-use-miniedit-mininets-graphical-user-interface/

5. Підручник з Python [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3.11/tutorial/index.html

6. Miniedit: графічний інтерфейс miniedit для створення і проектування прототипів мереж SDN /Бурлака Г.Ю. [Електронний ресурс] – Режим доступу до ресурсу: http://conferenc.its.kpi.ua/2022/paper/view/25673/14102

7. Огляд Mininet [Електронний ресурс] – Режим доступу до ресурсу: http://mininet.org/overview/

8. Огляд Oracle VM VirtualBox ,ресурсомісткість ,виконувані функції особливості установки та настройки [Електронний ресурс] – Режим доступу до ресурсу:https://hi-news.pp.ua/kompyuteri/13386-oracle-vm-virtualbox-scho-ce-za-programa-hto-rozrobniki-resursomstkst-vikonuvan-funkcyi-osoblivost-ustanovki-nastroyki.html

9. Основи SDN і мережевої архітектури OpenFlow_[Електронний ресурс] – Режим доступу до ресурсу:https://noviflow.com/the-basics-of-sdn-and-the-openflow-network-architecture/

10. Встановлення пакетів Python [Електронний ресурс] – Режим доступу до ресурсу:https://packaging.python.org/en/latest/tutorials/installing-packages/

11. Застосування SDN - рішень для оптимізації транспортних мереж мобільних операторів [Електронний ресурс] – Режим доступу до ресурсу: https://science.donntu.edu.ua/tks/volynskyi/diss/indexu.htm

12. Мережеві технології SDN – Software Defined Networking [Електронний ресурс] – Режим доступу до ресурсу:

https://habr.com/ru/companies/muk/articles/251959/

13. Програмно-конфігурована мережа [Електронний ресурс] – Режим доступу до ресурсу:https://community.fs.com/ru/blog/what-is-software-defined-networking-sdn.html

14. Перехід від звичайної мережі ЦОД до SDN [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/companies/huawei/articles/337918/

ДОДАТОК А

Код для створення простої мережі

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class SingleSwitchTopo(Topo):
    "Один комутатор, підключений до n хостів."
    def build(self, n=2):
        switch = self.addSwitch('s1')
        # Діапазон від 0 до N-1
        for h in range(n):
            host = self.addHost('h%s' % (h + 1))
            self.addLink(host, switch)

def simpleTest():
    "Створення та тестування простої мережі"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print("Вивід інформації про стан підключення хостів")
    dumpNodeConnections(net.hosts)
    print("Тест зв'язку у мережі")
    net.pingAll()
    net.stop()

if __name__ == '__main__':
    # Вивід інформації
    setLogLevel('info')
    simpleTest()
```

ДОДАТОК Б

Код для створення мережі(4 хости і 2 комутатори)

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class CustomTopology(Topo):
    def build(self):
        switch1 = self.addSwitch('s1')
        switch2 = self.addSwitch('s2')

        host1 = self.addHost('h1')
        host2 = self.addHost('h2')
        host3 = self.addHost('h3')
        host4 = self.addHost('h4')

        self.addLink(host1, switch1)
        self.addLink(host2, switch1)
        self.addLink(host3, switch2)
        self.addLink(host4, switch2)
        self.addLink(switch1, switch2)

def run_custom_topology():
    topo = CustomTopology()
    net = Mininet(topo)
    net.start()

    print("Вивід інформації про стан підключення хостів")
    dumpNodeConnections(net.hosts)

    net.pingAll()

    net.stop()

if __name__ == '__main__':
    setLogLevel('info')
    run_custom_topology()
```