МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ ФАКУЛЬТЕТ АЕРОНАВІГАЦІЇ, ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ КАФЕДРА ТЕЛЕКОМУНІКАЦІЙНИХ ТА РАДІОЕЛЕКТРОННИХ СИСТЕМ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____ Віктор ГНАТЮК
“_____”_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

Тема: «Розробка та побудова високопродуктивних телекомунікаційних мереж на базі технології SDN».

Виконавець: _____ Андрій БАКАЛИН
(підпис)

Керівник: _____ Олександр ПУЗИРЕНКО
(підпис)

Консультанти з окремих розділів пояснювальної записки:

Консультант розділу «Охорона праці»: _____ Батир ХАЛМУРАДОВ

Консультант розділу «Охорона навколишнього середовища»:
_____ Андріан ЯВНЮК
(підпис)

Нормоконтролер: _____ Денис БАХТІЯРОВ
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій

Кафедра телекомунікаційних та радіоелектронних систем

Спеціальність 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Телекомунікаційні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри

Віктор Гнатюк
“_____”_____2023 р.

ЗАВДАННЯ на виконання кваліфікаційної роботи

Бакалина Андрія Васильовича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема кваліфікаційної роботи: «Розробка та побудова високопродуктивних телекомунікаційних мереж на базі технології SDN»

затверджена наказом ректора від «28» вересня 2023 р. № 1965/ст

2. Термін виконання роботи: з 02.10.2023 р. по 31.12.2023 р.

3. Вихідні дані до роботи: аналіз сучасних програмно-конфігурованих мереж, аналіз трафіку, система моніторингу, експериментальне тестування запропонованого методу побудови, оцінка ефективності і надійності розробленого методу

4. Зміст пояснювальної записки: принцип побудови мультимедійної мережі за допомогою технології OpenFlow.

5. Перелік обов'язкового графічного (ілюстративного) матеріалу: слайди презентації в програмному пакеті Microsoft PowerPoint

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Розробити деталізований зміст розділів кваліфікаційної роботи	02.10.2023-04.10.2023	Виконано
2	Вступ	05.10.2023 - 08.10.2023	Виконано
3	ПРОГРАМНО КОНФІГУРОВАНА МЕРЕЖА	09.10.2023-22.10.2023	Виконано
4	MININET	23.10.2023-05.11.2023	Виконано
5	OPENFLOW	08.06.2023-14.06.2023	Виконано
6	КОНТРОЛЕРИ	15.06.2023-20.06.2023	Виконано
7	ВПРОВАДЖЕННЯ ТА ТЕСТУВАННЯ	20.06.2023-01.12.2023	Виконано
8	Охорона праці	01.12.2023-06.12.2023	Виконано
9	Охорона навколишнього середовища	07.12.2023-17.12.2023	Виконано
10	Усунення недоліків та захист кваліфікаційної роботи	18.12.2023-31.12.2023	Виконано

7. Консультанти з окремих розділів

Розділ	Консультант (посада, П.І.Б.)	Дата, підпис	
		Завдання видав	Завдання прийняв
Охорона праці	к.м.н., професор Батир ХАЛМУРАДОВ		
Охорона навколишнього середовища	к.б.н., доц. Андріан ЯВНЮК		

8. Дата видачі завдання: “26" грудня 2023 р.

Керівник кваліфікаційної роботи _____ Олександр ПУЗИРЕНКО
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Андрій БАКАЛИН
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Кваліфікаційна робота «Розробка та побудова високопродуктивних телекомунікаційних мереж на базі технології SDN» містить _90_ сторінок, _24_ рисунків, _21_ використане джерело.

SDN, OPENFLOW, MININET, CONTROLLER, TCP, QOS

Об'єктом дослідження – спосіб передачі трафіку в програмно-конфігурованій мережі.

Предмет дослідження – є дослідження розробка методів доставки потокового відео в програмно-конфігурованій мережі з використанням технології OpenFlow.

Мета кваліфікаційної роботи –. вивчення потенціалу SDN для покращення продуктивності потокової передачі відео.

Метод дослідження –. проаналізувати сучасні методи та технології побудування мереж, та розробити високопродуктивну телекомунікаційну мережу на базі технології програмно-конфігурованої мережі (SDN).

Матеріали кваліфікаційної роботи рекомендується використовувати в якості підґрунтя для розробки нових методів або рішень в області розробки нових методів побудування мереж. Ці матеріали можуть стати основою для подальшої розробки продуктів або систем, які будуть використовуватися в індустрії.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

SDN (Software-Defined Networking) – програмно конфігурована мережа

API (Application Programming Interface) – інтерфейс програмування додатків

CLI (Command Line Interface) – (інтерфейс командного рядка)

IP (Internet Protocol) – (протокол Інтернету)

UDP (User Datagram Protocol) – (протокол користувацьких дейтаграм)

TCP (Transmission Control Protocol) – (протокол передачі даних)

HTTP (Hypertext Transfer Protocol) – (протокол передачі гіпертексту)

TLS (Transport Layer Security) – (безпека транспортного рівня)

OVSDB(Open vSwitch Database) – протокол для керування віртуальними комутаторами

BGP (Border Gateway Protocol) – (протокол міжмережевого шлюзу

LLDP (Link Layer Discovery Protocol) – протокол виявлення мережевого рівня

GUI (Graphical User Interface) – (графічний інтерфейс користувача)

ВСТУП

Актуальність теми. Швидке зростання споживання онлайн-відео викликало необхідність розробки ефективних і масштабованих рішень для надання високоякісного потокового передавання кінцевим користувачам. SDN з його централізованим керуванням і програмованістю пропонує перспективний підхід до оптимізації продуктивності потокового відео.

Зв'язок роботи з науковими програмами, планами, темами.

Мета і завдання дослідження. Дослідження, зосереджені на використанні принципів і технологій SDN для покращення потокового відео з точки зору якості, надійності та адаптивності. Дослідження включає розробку та впровадження системи потокового відео з використанням мережевих симуляторів Mininet та MiniEdit за підтримки Wireshark, VLC Media Player та протоколу OpenFlow.

Для досягнення поставленої мети вирішуються такі наукові завдання.

1. Дослідження потокового відео через інфраструктуру програмно-визначених мереж (SDN)
2. Розробка та впровадження системи потокового відео з використанням мережевих симуляторів Mininet та MiniEdit.
3. Оцінка продуктивності

Об'єктом дослідження – спосіб передачі трафіку в програмно-конфігурованій мережі.

Предметом дослідження є дослідження та розробка методів доставки потокового відео в програмно-конфігурованій мережі з використанням технології OpenFlow

Методи досліджень. збір та аналіз статистичних даних: збір інформації про нові технології побудування сучасних мереж. Вимірювання показників пропускної здатності, завантаженості мережі, а також аналіз типових характеристик якості мережі.

Наукова новизна та практичне значення отриманих результатів.

Практичне значення отриманих результатів.

Ця кваліфікаційна робота стане цінним джерелом для дослідників, інженерів мереж та студентів, які цікавляться дослідженням концепції SDN і потокової передачі відео. Вона не лише надає всебічний огляд предмету, а й пропонує практичні висновки та рекомендації для впровадження рішень з відеопотоковою передачею на основі SDN. Завдяки цьому дослідженню ми сприяємо зростанню обсягу знань про SDN і підкреслюємо потенціал цієї технології в сфері потокового відео

Апробація отриманих результатів. Основні положення роботи доповідалися та обговорювалися на таких конференціях:

- Науково-практична конференція «Проблеми експлуатації та захисту інформаційно-комунікаційних систем», м. Київ, 2023 р.

РОЗДІЛ 1
ПРОГРАМНО КОНФІГУРОВАНА МЕРЕЖА (SDN)

1.1. Загальні відомості про програмно-конфігуровані мережі

Програмно-конфігурована мережа (SDN) полегшує організаціям розгортання додатків і забезпечує гнучку доставку, надаючи можливість масштабування мережевих ресурсів у відповідності з потребами додатків і даних, що значно зменшує капітальні та оперативні витрати [1]. SDN — це інноваційний підхід до проектування, впровадження та керування мережами, який розділяє управління мережею (рівень управління) та процес пересилання (рівень даних) для кращої взаємодії з користувачем. Така сегментація мережі надає численні переваги з точки зору гнучкості та керованості мережі. З одного боку, це дозволяє поєднати переваги системної віртуалізації та хмарних обчислень, а з іншого боку, створити реалізацію централізованого інтелекту, що забезпечує чітку видимість у мережі для легкого керування та обслуговування мережі. У звичайній інфраструктурі (рис. 1.1 і таблиця 1.1.) впровадження, конфігурація та усунення неполадок вимагають втручання висококваліфікованих фахівців з мереж та системних інженерів із високим рівнем технічної експертизи, а також операційні витрати, пов’язані із забезпеченням і керуванням великими мережами різних постачальників.

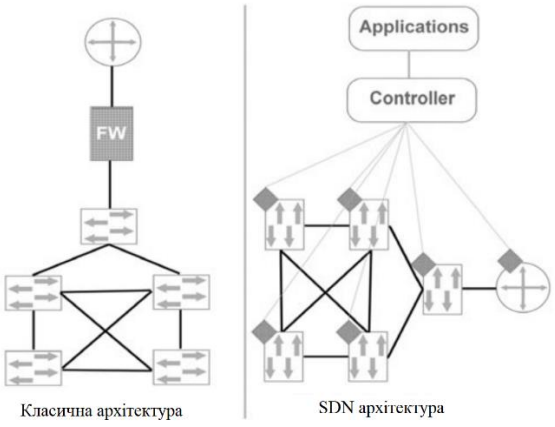


Рис. 1.1.Класична архітектура зліва і архітектура SDN справа

Таблиця 1.1.

Порівняння підходу SDN з класичними мережами

Характеристики	SDN мережа	Класична мережа
Програмованість	✓	
Централізований контроль	✓	
Схильний до помилок в конфігурації		✓
Складний контроль мережі		✓
Гнучкість мережі	✓	
Покращена продуктивність	✓	
Легке впровадження	✓	
Ефективна конфігурація	✓	
Покращене управління	✓	

Фактично, різноманіття та складність [2] елементів мережі роблять їх обслуговування дуже витратним, а базову інфраструктуру менш надійною у разі частих збоїв мережі, особливо якщо резервні плани не передбачені в інфраструктурі мережі.

Оскільки SDN відокремлює прийняття рішень щодо маршрутизації та пересилання даних в мережевих елементах (наприклад, маршрутизаторах, комутаторах та точоках доступу) від плану даних, адміністрування та управління мережею стають простішими, оскільки план управління взаємодіє лише з інформацією, що стосується логічної топології мережі, маршрутизації трафіку і так далі. Натомість план даних оркеструє мережевий трафік відповідно до встановленої конфігурації в плані управління.

У SDN операції управління централізовані в контролері, який визначає політики мережі. Багато платформ контролера мають відкритий код, наприклад

Floodlight [3], OpenDayLight [4] та Beacon [17]. Управління мережею може здійснюватися на різних рівнях (рівні додатків, контролю та плану даних). Наприклад, постачальники послуг можуть виділяти ресурси клієнтам через рівень додатків, налаштовувати та змінювати політики мережі та логічні структури на рівні контролю, а також налаштовувати фізичні елементи мережі на рівні даних.

Програмно-визначені мережі набули популярності лише у останні роки. Однак концепція цього підходу розвивалася з середини 1990-х років. Ethane [6] (архітектура управління) та OpenFlow [7] (протокол потоку мережі) породили реальну реалізацію SDN. OpenFlow є протоколом, який надає стандартизований спосіб керування трафіком і описує, як контролер спілкується з мережевими пристроями, такими як комутатори та маршрутизатори. Пристрої, що підтримують OpenFlow, складаються з двох логічних компонентів: таблиці потоків, яка визначає, як обробляти та пересилати пакети в мережі, і відкритого програмного інтерфейсу (API) OpenFlow, який керує обмінами між комутатором/маршрутизатором та контролером.

1.2. Архітектура програмно-конфігурованих мереж

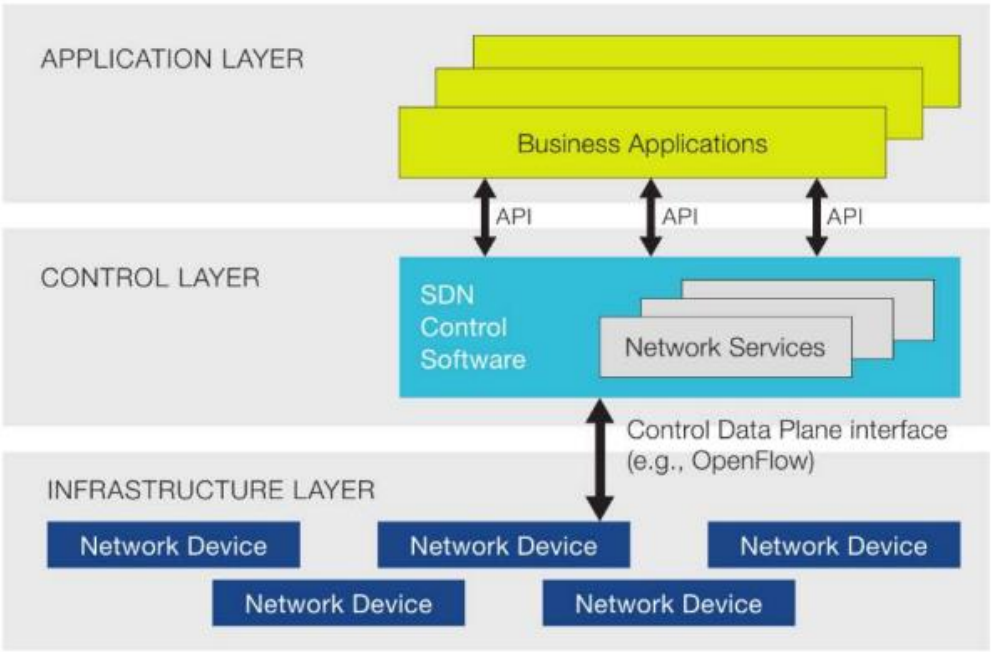


Рисунок 1.2.Схематичне зображення архітектури SDN

Архітектура SDN зазвичай має три компоненти або групи функціональних можливостей:

Мережні пристрої SDN керують можливостями пересилання та обробки даних для мережі. Це включає в себе переадресацію та обробку маршруту даних. Приклад: Open vSwitch (протокол OpenFlow)

Контролер SDN

Контролер SDN - це логічний елемент, який отримує інструкції або вимоги від рівня додатків SDN та передає їх мережевим компонентам. Контролер також отримує інформацію про мережу від апаратних пристроїв та повертає до додатків SDN абстрактне уявлення про мережу, включаючи статистику та події про те, що відбувається.

Мережевий контролер SDN має ключове значення, оскільки він відповідає за управління елементами мережевої інфраструктури та потоками даних у мережі. Надійність мережі SDN напряму залежить від надійності самого контролера.

Приклад контоллерів: RYU, Open Daylight, ONOS, Floodlight тощо

SDN додатки

Додатки SDN — це програми, які взаємодіють з контролером SDN через інтерфейс прикладного програмування (API). Ці програми можуть включати управління мережею, аналітику або бізнес-додатки, які використовуються для запуску великих центрів обробки даних. Це місце для досліджень, інновацій, нових ідей тощо.

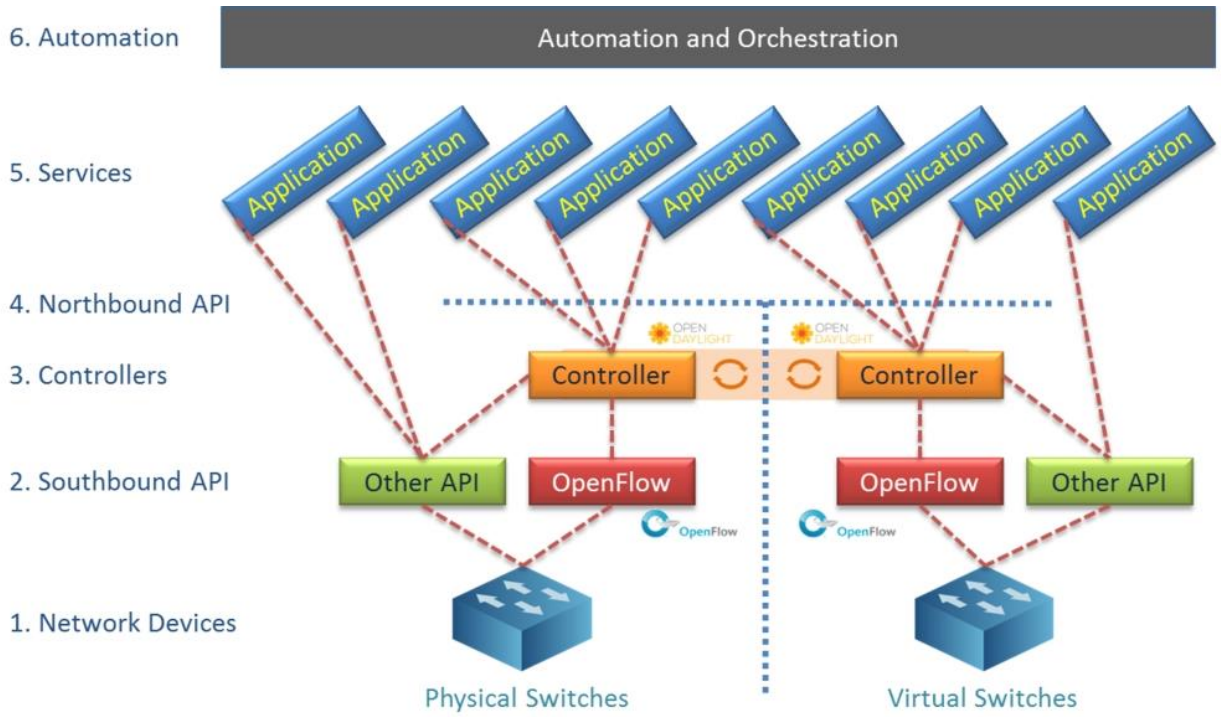


Рис. 1.3.Схематичне зображення взаємодії додатків з іншими компонентами мережі

1.3 Встановлення випробувального стенду SDN

Налаштовуємо тестове середовище SDN для відпрацювання сценаріїв використання OpenFlow за допомогою контролера RYU SDN.

Мінімальні технічні характеристики нашого ПК:

ОС: Ubuntu 20.04

Процесор: 2 ядра +

Оперативна пам'ять: 4 ГБ +

Жорсткий диск: 15 ГБ+

Виконуємо наведені нижче команди в Терміналі UBUNTU

```
sudo apt update
sudo apt install python3 python3-pip xterm iperf hping3 net-tools wireshark apache2-utils curl
sudo apt install mininet
sudo pip3 install ryu
sudo pip3 install mininet
sudo cp /usr/bin/python3 /usr/bin/python
ryu-manager --version
sudo mn --version
```

Щоб перевірити версію python:

python3 --version або python –version

```
Python 3.8.5
```

Щоб перевірити:

ovs-vsctl –version

```
ovs-vsctl (Open vSwitch) 2.13.1
DB Schema 8.2.0
```

Перевірка:

Відкриваємо 4 термінали:

1. У Терміналі 1

```
sudo wireshark
```

І запускаємо захоплення для "loopback" або "будь-якого" інтерфейсу.

2. У Терміналі 2

```
ryu-manager ryu.app.simple_switch_13
```

3. У Терміналі 3

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4
```

Ми отримаємо запит Mininet. У запиті Mininet введимо команду pingall

pingall

Журнал (Logs):

```
abakalyn@test sudo mn --controller=remote,ip=127.0.0.1 --mac --
switch=ovsk,protocols=OpenFlow13 --topo=single,4
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** 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 ...
*** Starting CLI:
mininet> pingall
*** 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)
mininet>
```

4. У Терміналі 4

sudo ovs-vsctl show

sudo ovs-ofctl -O OpenFlow13 dump-flows s1

Logs:

```
abakalyn@test-vm:~$ sudo ovs-vsctl show
[sudo] password for suresh:
a315e8b4-dd3f-42f6-b84a-e967e02660a4
    Bridge "s1"
        Controller "tcp:127.0.0.1:6653"
            is_connected: true
        Controller "ptcp:6654"
        fail_mode: secure
        Port "s1-eth4"
            Interface "s1-eth4"
```

```
        Port "s1"
            Interface "s1"
                type: internal
        Port "s1-eth1"
            Interface "s1-eth1"
        Port "s1-eth3"
            Interface "s1-eth3"
        Port "s1-eth2"
            Interface "s1-eth2"
    ovs_version: "2.13.1"
abakalyn@test-vm:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=115.430s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:01
actions=output:"s1-eth1"
 cookie=0x0, duration=115.421s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02
actions=output:"s1-eth2"
 cookie=0x0, duration=115.410s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01
actions=output:"s1-eth1"
 cookie=0x0, duration=115.404s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:03
actions=output:"s1-eth3"
 cookie=0x0, duration=115.391s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01
actions=output:"s1-eth1"
 cookie=0x0, duration=115.380s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04
actions=output:"s1-eth4"
 cookie=0x0, duration=115.370s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:02
actions=output:"s1-eth2"
 cookie=0x0, duration=115.368s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:03
actions=output:"s1-eth3"
 cookie=0x0, duration=115.361s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:02
actions=output:"s1-eth2"
```

```
 cookie=0x0, duration=115.359s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:04
actions=output:"s1-eth4"
 cookie=0x0, duration=115.346s, table=0, n_packets=3, n_bytes=238,
priority=1,in_port="s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:03
actions=output:"s1-eth3"
 cookie=0x0, duration=115.344s, table=0, n_packets=2, n_bytes=140,
priority=1,in_port="s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:04
actions=output:"s1-eth4"
 cookie=0x0, duration=164.572s, table=0, n_packets=58, n_bytes=4276, priority=0
actions=CONTROLLER:65535
abakalyn@test-vm:~$
```

5. Перевіримо повідомлення OpenFlow у Wireshark

Зупиняємо сканування у Wireshark,

У фільтрі введимо "openflow_v4", щоб побачити повідомлення OPENFLOW. (Рис. 1.3)

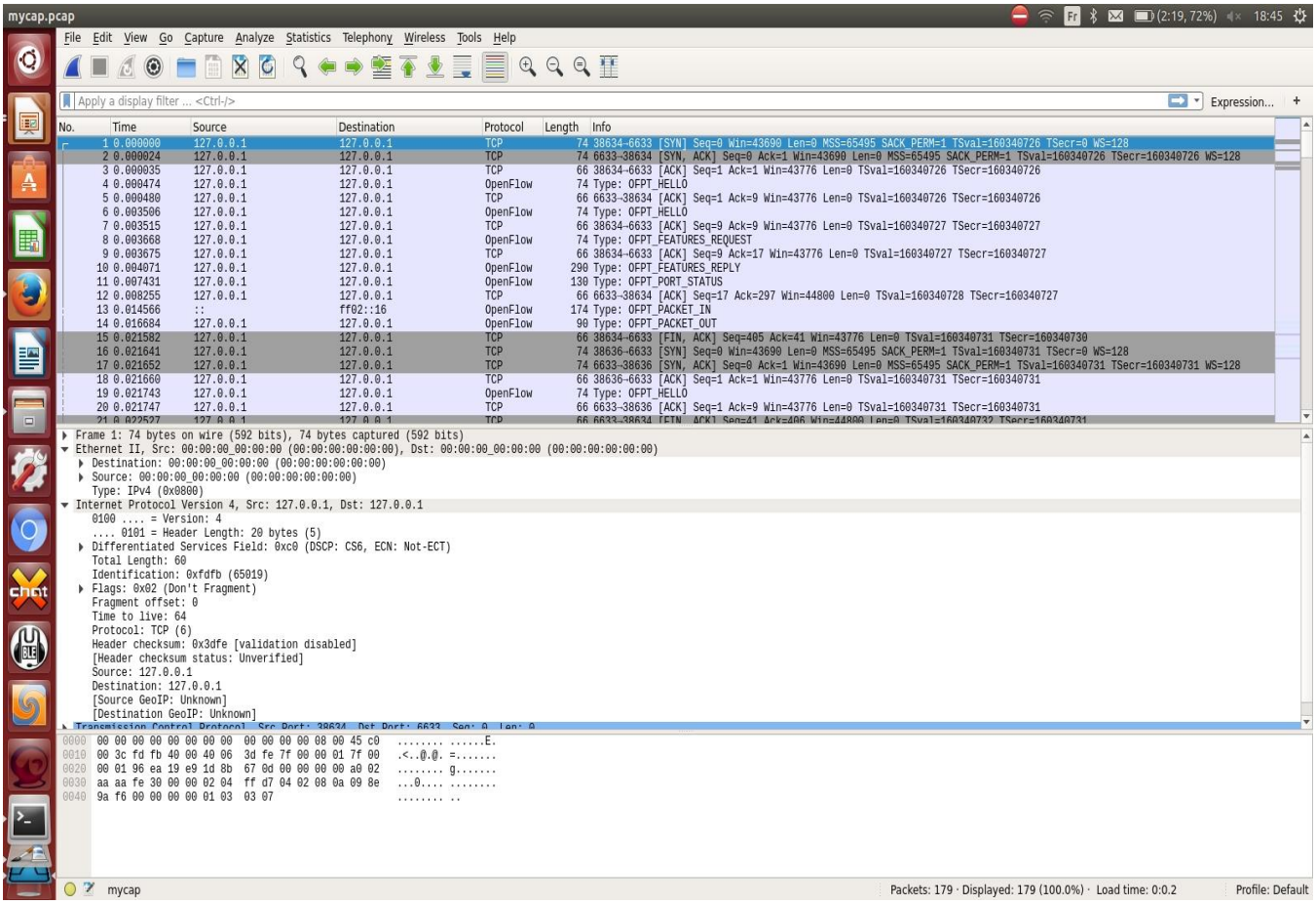


Рис. 1.4.Повідомлення OpenFlow у Wireshark

1.4 Комутатор OpenFlow

Комутатор OpenFlow - це комутатор даних з підтримкою OpenFlow, який зв'язується по каналу OpenFlow з зовнішнім контролером. Він виконує пошук і пересилку пакетів відповідно до однієї або декількох таблиць потоків і групових таблиць. Комутатор OpenFlow зв'язується з контролером, і контролер управляє комутатором через протокол комутатора OpenFlow. Вони можуть бути або засновані на протоколі OpenFlow, або сумісні з ним.

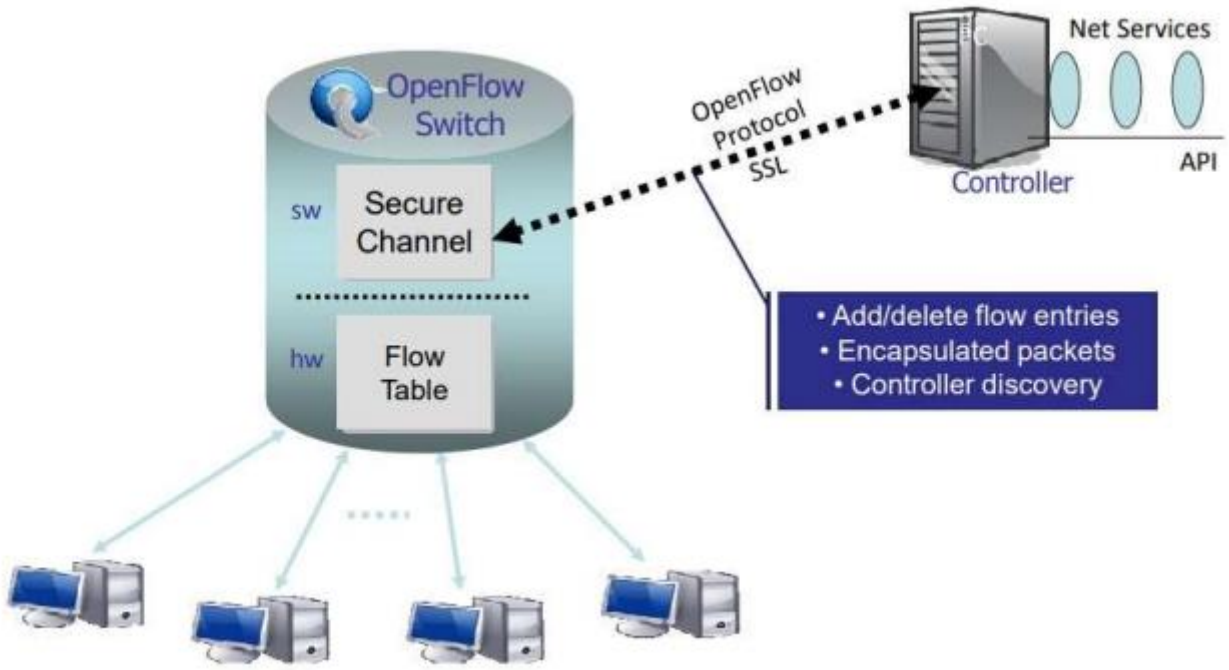


Рис. 1.5.OpenFlow комутатор в мережі SDN

Комутатор OpenFlow може функціонувати тільки при спільній роботі трьох основних елементів:

- таблиць потоків (Flow Tables), встановлених на комутаторах,
- контролера,
- пропрієтарного протоколу OpenFlow для безпечної взаємодії контролера з комутаторами.

Таблиці потоків встановлюються на комутаторах. Контролери спілкуються з комутаторами по протоколу OpenFlow і накладають політики на потоки. Контролер може встановлювати шляхи через мережу, оптимізовані для конкретних характеристик, таких як швидкість, мінімальна кількість стрибків або зменшена затримка.

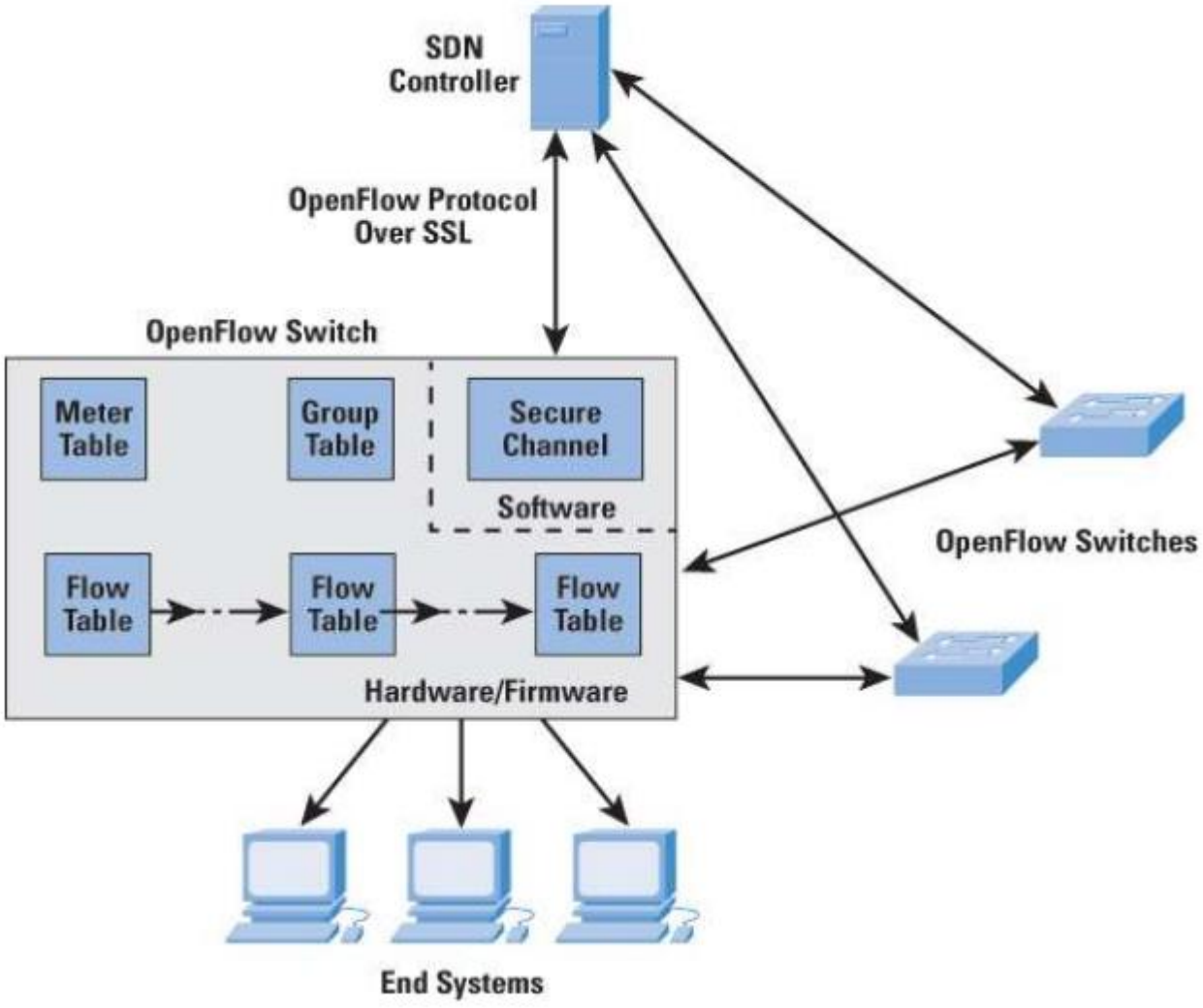


Рис. 1.6.Використання протоколу OpenFlow в мережі SDN

Запускаємо та налаштовуємо OpenFlow комутатор

1. Запуск лінійної топології мінінету

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
switch=ovsk,protocols=OpenFlow13 --topo=linear,4
```

2. Вмикаємо захоплення у Wireshark
3. Застосунок RYU L3

```
ryu-manager ryu.app.simple_switch_13
```

4. Перевіряємо потоки OpenFlow

```
sudo ovs-ofctl -O OpenFlow13 dump-flows s1
```

Принцип роботи:

1. Комутатор налаштований з IP-адресою контролера SDN та версією протоколу OpenFlow.
2. Комутатор встановлює зв'язок з контролером SDN.
3. Контролер SDN встановлює стандартне правило OpenFlow (TABLE MISS ENTRY) в таблиці потоків комутатора.
4. Правило TABLE MISS Entry OpenFlow відповідає всім пакетам і відсилає їх до КОНТРОЛЕРА. Найнижчий пріоритет у таблиці (0).
5. Коли пакет даних від хоста надходить до комутатора, він буде зіставлений з TABLE MISS ENTRY, і пакет буде відправлений до контролера (повідомлення PACKET IN).
6. Контролер отримує пакет і будує логіку комутатора на основі цих пакетів.
7. Контролер додає потоки OPENFLOW до комутатора.
8. Тепер шлях обробки даних у комутаторі побудований за допомогою потоків. Тому наступного разу, коли пакет прийде, він буде порівняний з таблицею потоків і пересланий на відповідний порт

РОЗДІЛ 2
MININET

2.1 Загальні відомості про mininet

Mininet - це емулятор мережі, який створює мережу віртуальних хостів, комутаторів, контролерів та зв'язків. Хости Mininet працюють на стандартному програмному забезпеченні мережі Linux, а їх комутатори підтримують OpenFlow для високогнучкого налаштування маршрутизації та програмно-визначеної мережі.

Mininet підтримує наукові дослідження, розробку, навчання, прототипування, тестування, налагодження та будь-які інші завдання, які можуть бути корисними завдяки наявності повноцінної експериментальної мережі на ноутбуці або іншому комп'ютері.

Mininet:

- Надає просте та доступне тестове середовище для розробки застосунків OpenFlow.
- Дозволяє кільком розробникам незалежно та одночасно працювати над однією топологією.
- Підтримує системні тести на рівні системи, які можна легко повторювати.
- Дозволяє проводити складні тести топологій без необхідності під'єднувати фізичну мережу.
- Має інтерфейс командного рядка (CLI), який розуміє топологію та OpenFlow для налагодження або запуску тестів на рівні мережі.
- Підтримує довільні користувацькі топології та має базовий набір параметризованих топологій.
- Може використовуватися безпосередньо без програмування, але також надає простий та розширюваний Python API для створення та експериментів з мережами.

Mininet пропонує простий спосіб отримати правильну поведінку системи (і, наскільки це підтримується вашим обладнанням, продуктивність) та експериментувати з топологіями.

Мережі Mininet виконують реальний код, у тому числі стандартні мережні додатки Unix/Linux, а також реальне ядро Linux та мережевий стек (включаючи будь-які розширення ядра, які вам доступні, поки вони сумісні з просторами імен мережі).

Через це код, який ви розробляєте і тестуєте на Mininet для контролера OpenFlow, конфігурованого комутатора або хосту, може перейти на реальну систему з мінімальними змінами для тестування в реальних умовах, оцінки продуктивності та впровадження. Це означає, що дизайн, який працює в Mininet, зазвичай може безпосередньо перейти на апаратні комутатори для пересилання пакетів з лінійною швидкістю.[8]

2.2 **Основні операції**

1. Перевірити версію Mininet

```
mn --version
```

2.Очистити існуючі мости (bridges) ovs та простори імен (namespaces):

Примітка: іноді ми помиляємося і закриваємо оболонку Mininet або Mininet впадає. Але компоненти топології продовжують існувати. Для очищення таких об'єктів використовується команда очищення (cleanup).

```
mn -c
```

2. Наша перша топологія (одиночна)

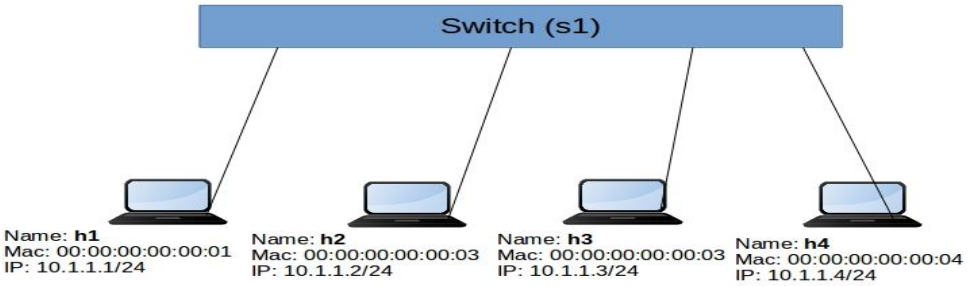


Рис. 2.1.Топологія з одним перемикачем і 4 вузлами.

Контролер RYU SDN

```
ryu-manager ryu.app.simple_switch_13
```

Топологія Mininet

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
switch=ovsk,protocols=OpenFlow13 --topo=single,4
```

Ключ	Опис
--controller	Тип контролера: локальний/віддалений та IP-адреса віддаленого контролера
--mac	Початкова MAC-адреса: 00:00:00:00:00:01
-i	Підмережі IP для топології
--switch	Тип комутатора (ovsk - модуль ядра openvswitch) і версія OpenFlow.
--topo	Тип топології (лінійна, мінімальна, зворотня, одиночна, торус або дерево) та параметри.

Приклад:

```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 4 links
....
*** Stopping 1 switches
s1
*** Stopping 4 hosts
h1 h2 h3 h4
*** Done
completed in 2.604 seconds
abakalyn@test-vm:~$
```

4. Основні команди оболонки Mininet:

nodes - Показує список вузлів (хостів, комутаторів) у поточній топології.

net - Показує інформацію про мережу, включаючи з'єднання між вузлами.

Pingall - Відправляє ping між усіма вузлами в мережі для перевірки доступності.

exit або quit - Вихід з оболонки Mininet.

Help - Показує список доступних команд або довідкову інформацію про команду.

links - Відображає список зв'язків між вузлами (хостами, комутаторами) у поточній мережній топології.

dump - Вивід, що містить інформацію про всі вузли (хости та комутатори), їх стан та поточні параметри, такі як адреси MAC та IP, таблиці маршрутизації, налаштування OpenFlow (якщо використовується), статуси з'єднань між вузлами та інше.

Ми можемо увійти до кожного хоста за допомогою команди 'xterm'

```
mininet>xterm h1
```

Відкриється термінал xterm для основної системи. Тепер ми можемо виконати команду всередині цього терміналу

Абож ми можемо безпосередньо виконати з оболонки Mininet.

```
mininet><hostname> command
```

Наприклад

```
mininet>h1 ifconfig
mininet>h1 ping h2
mininet>h1 ip route
```

5. Лінійна топологія

Лінійна топологія - це структура мережі, де кожен комутатор має одного хоста, і всі комутатори з'єднані у лінію.

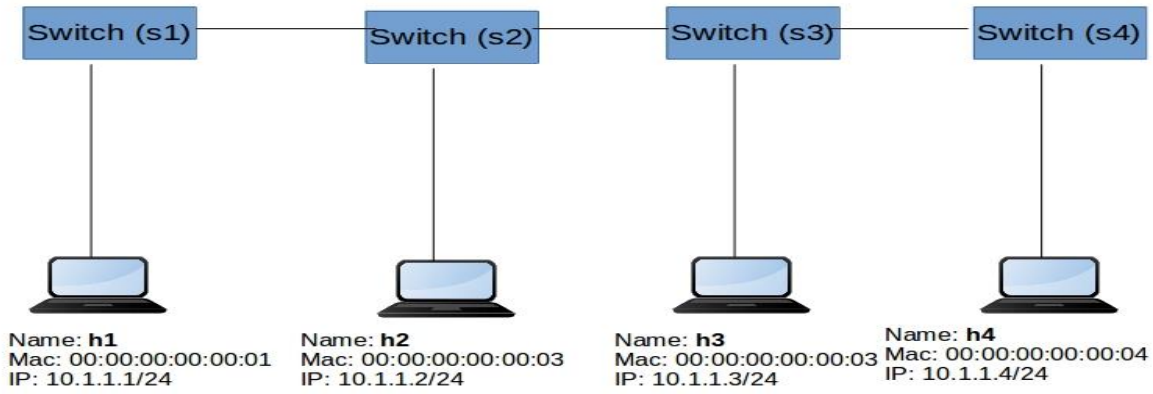


Рис. 2.2.Лінійна топологія

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
switch=ovsk,protocols=OpenFlow13 --topo=linear,4
```

6. Деревоподібна топологія

Топологія дерева - структура мережі, де комутатори та хости організовані у формі дерева. У цій топології комутатори з'єднані між собою у вигляді гілок та стовбурів, утворюючи ієрархічну мережу.

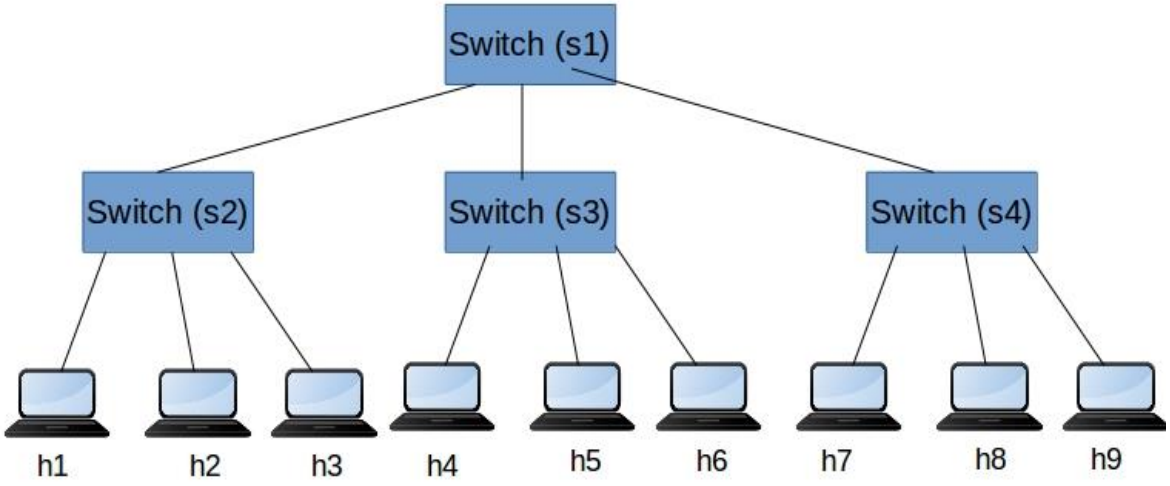


Рис. 2.3.Деревоподібна топологія

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
topo=tree,depth=2,fanout=3
```

Fanout: кількість дочірніх вузлів, до яких підключений кожен комутатор.

Depth: глибина дерева, означає кількість рівнів (висоту) структури дерева.

2.3 Запуск тестів трафіку

1.Тести трафіку TCP/UDP

Налаштуємо лінійну топологію за допомогою параметрів xterms (відкриваємо термінал для кожного вузла)

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
switch=ovsk,protocols=OpenFlow13 --topo=linear,4  -x
```

Тест трафіку TCP від h1 до h4

Запускаємо сервер IPERF TCP на h4

```
iperf -s
```

Запускаємо IPERF TCP-клієнт на h1

```
iperf -c 10.1.1.4 -i 10 -t 30
iperf -c 10.1.1.4 -i 10 -b 10m -t 30
iperf -c 10.1.1.4 -i 10 -P 10 -t 30
```

Ключ	Опис
-c	режим клієнта
-i	інтервал звітності
-t	тривалість тесту в секундах
-b	пропускну здатність 10m означає 10 Мбіт/с
-P	кількість паралельних потоків

Тест трафіку UDP від h1 до h4

Запускаємо сервер IPERF UDP на h4

```
iperf -u -s
```

-u : означає udp

Запускаємо IPERF UDP-клієнт на h1

```
iperf -u -c 10.1.1.4 -b 10m -i 10 -t 30
iperf -u -c 10.1.1.4 -b 10m -i 10 -P 10 -t 30
```

Тести HTTP-трафіку

Запускаємо простий веб-сервер Python на h4

```
python -m SimpleHTTPServer 80
```

З h1, доступ до веб-сервера

```
curl http://10.1.1.4/
```

Утиліта curl використовується як веб-клієнт для доступу до веб-сервера.

Якщо ми хочемо змоделювати 1000-х користувачів, які звертаються до веб-сервера в один і той же час (завантаження), ми можемо використовувати інструмент ab (Apache bench)

```
ab -n 500 -c 50 http://10.1.1.4/
```

-c 50 означає паралельний запит за секунду (50 запитів за секунду)

-n 500 означає загальний запит на цей тест (500 запитів)

Запис власної топології в Mininet.

Mininet надає доступ до Python API. Ми можемо створити власну топологію за допомогою Python API всього кількома рядками коду.

Імпортуємо необхідні бібліотеки python

```
from mininet.topo import Topo

from mininet.net import Mininet
```

Пишемо клас визначення топології

```
class SingleSwitchTopo(Topo):
  def build(self):
    s1 = self.addSwitch('s1')


    h1 = self.addHost('h1')
    h2 = self.addHost('h2')
    h3 = self.addHost('h3')
    h4 = self.addHost('h4')


    self.addLink(h1, s1)
    self.addLink(h2, s1)
    self.addLink(h3, s1)
    self.addLink(h4, s1)
```

Важливі API визначення топології

```
addSwitch
addHost
addLink
```

Запускаємо топологію:

- Створюємо об'єкт топології
- Створюємо об'єкт Mininet with Topology
- Запускаємо мінінет

```
if __name__ == '__main__':
    topo = SingleSwitchTopo()
    c1 = RemoteController('c1', ip='127.0.0.1')
    net = Mininet(topo=topo, controller=c1)
    net.start()
```

Запускаємо:

Запускаємо контролер RYU SDN

```
ryu-manager ryu.app.simple_switch_13
```

Запустіть файл топології Mininet

```
sudo python <topology file name>
```

Проведимо тести/операцію тощо.

Розділ 3
OPENFLOW

3.1 Загальні відомості про OpenFlow

OpenFlow - це протокол взаємодії між мережевими пристроями, такими як комутатори і маршрутизатори, і централізованим контролером (SDN-контролером), який являє собою мережеву операційну систему, встановлену на виділеному фізичному сервері, в програмно-конфігурованій мережі.

Цей протокол відокремлює програмне забезпечення мережевих пристроїв від їх базового обладнання і пропонує стандартизований метод постачання централізованої, програмованої мережі, яка може швидко адаптуватися до змінних вимог до мережі. Специфікація OPENFLOW охоплює компоненти та основні функції комутатора, а також протокол комутатора OpenFlow для управління комутатором OpenFlow з віддаленого контролера OpenFlow.

3.2 Компоненти комутатора

Логічний комутатор OpenFlow складається з однієї або декількох таблиць потоків і групової таблиці, які виконують пошук і пересилання пакетів, а також одного або декількох каналів OpenFlow на зовнішній контролер (рис. 3.1).

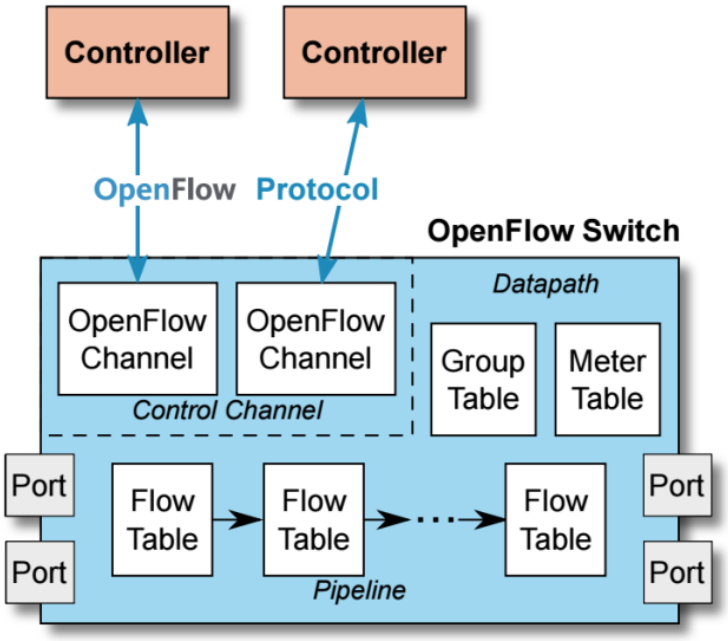


Рис.3.1.Основні компоненти комутатора

- Комутатор спілкується з контролером, а контролер керує комутатором за допомогою протоколу комутатора OpenFlow.
- Використовуючи протокол комутатора OpenFlow, контролер може додавати, оновлювати та видаляти записи потоків у таблицях потоків як реактивно (у відповідь на пакети), так і превентивно.
- Кожна таблиця потоків у комутаторі містить набір записів потоків; кожний запис потоку складається з полів відповідності, лічильників та набору інструкцій для застосування до відповідних пакетів.
- Пошук відбувається в першій таблиці потоків і може продовжуватися до додаткових таблиць потоків у конвеєрі.
- Записи потоків відповідають пакетам у порядку пріоритету, використовуючи перший підходящий запис у кожній таблиці.
- Якщо знайдено відповідний запис, виконуються інструкції, пов'язані із конкретним записом потоку.
- Якщо у таблиці потоків не знайдено відповідного запису, результат залежить від конфігурації запису потоку для випадку "відсутності в таблиці": наприклад,

пакет може бути пересланий до контролера через канал OpenFlow, відкинутий або продовжити шлях до наступної таблиці потоків.

- Дії, включені в інструкції, описують пересилання пакетів, модифікацію пакетів та обробку таблиці груп.
- Записи потоків можуть пересилати до порту. Це зазвичай фізичний порт, але це також може бути логічний порт, визначений комутатором (наприклад, групи агрегації зв'язку, тунелі чи інтерфейси зворотного зв'язку), або зарезервований порт, визначений цією специфікацією.

3.3 Канал OpenFlow

- Канал OpenFlow є інтерфейсом, що з'єднує кожний логічний комутатор OpenFlow з контролером OpenFlow. Через цей інтерфейс контролер налаштовує та керує комутатором, отримує події від комутатора та відправляє пакети через комутатор.
- Керуючий канал комутатора може підтримувати один канал OpenFlow з одним контролером або кілька каналів OpenFlow, що дозволяє використовувати кілька контролерів.

Канал OpenFlow зазвичай шифрується за допомогою TLS (Transport Layer Security)

- , але може працювати безпосередньо через TCP.
- Номер порту за замовчуванням: 6653

3.4 Таблиця потоку OpenFlow

Кожен запис таблиці потоків ідентифікується своїми полями відповідності та пріоритетом: поля відповідності та пріоритет разом ідентифікують унікальний запис потоку у конкретній таблиці потоків. Таблиця потоків складається з записів потоків.

Таблиця 3.1

Основні компоненти запису потоку в таблиці потоків

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Основні компоненти запису потоку в таблиці потоків:

Match Fields (Поля відповідності): Визначають умови, які пакет повинен відповідати, щоб бути обробленим цим записом потоку.

Priority (Пріоритет): Вказує на важливість або пріоритетність цього запису потоку порівняно з іншими записами в тій самій таблиці потоків.

Instructions (Інструкції): Включають дії, які виконуються над пакетами, що відповідають цьому запису потоку, такі як пересилання пакету, зміна пакету або переадресація.

Counters (Лічильники): Вимірюють певні параметри або статистику, пов'язану з цим записом потоку, такі як кількість пакетів, що відповідають цьому запису.

У контексті OpenFlow та записів потоків можуть бути деякі інші атрибути, які використовуються для керування потоками даних:

Timeouts (Таймаути): Часовий інтервал, після якого запис потоку може бути видалений з таблиці потоків, якщо він не був використаний. Це дозволяє управляти пам'яттю та ефективністю таблиці потоків.

Cookie: Унікальний ідентифікатор або токен, який використовується для ідентифікації конкретного запису потоку або для пов'язання його з певними параметрами чи діями.

Flags (Прапорці): Додаткові біти або прапорці, які можуть вказувати певні характеристики або стан запису потоку, наприклад, можливості виконання певних операцій або обробки пакетів.

Приклад потоків із MAC відповідністю

```
 cookie=0x0, duration=4.742s, table=0, n_packets=2, n_bytes=196,
priority=1,in_port="s1-eth2",dl_src=00:00:00:00:11:12,dl_dst=00:00:00:00:11:11
actions=output:"s1-eth1"
 cookie=0x0, duration=4.738s, table=0, n_packets=1, n_bytes=98,
priority=1,in_port="s1-eth1",dl_src=00:00:00:00:11:11,dl_dst=00:00:00:00:11:12
actions=output:"s1-eth2"
 cookie=0x0, duration=5.781s, table=0, n_packets=29, n_bytes=3102, priority=0
actions=CONTROLLER:65535
```

Приклад потоків із відповідністю IP

```
 cookie=0x0, duration=12.927s, table=0, n_packets=2, n_bytes=196,
priority=1,ip,nw_src=192.168.1.1,nw_dst=192.168.1.2 actions=output:"s1-eth2"
 cookie=0x0, duration=12.918s, table=0, n_packets=2, n_bytes=196,
priority=1,ip,nw_src=192.168.1.2,nw_dst=192.168.1.1 actions=output:"s1-eth1"
 cookie=0x0, duration=12.959s, table=0, n_packets=37, n_bytes=3844, priority=0
actions=CONTROLLER:65535
```

Приклади потоків з відповідністю портів TCP/UDP

```
abakalyn@test-vm:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=3.933s, table=0, n_packets=238752, n_bytes=11069190464,
priority=1,tcp,nw_src=192.168.1.2,nw_dst=192.168.1.1,tp_src=37304,tp_dst=5001
actions=output:"s1-eth1"
 cookie=0x0, duration=3.906s, table=0, n_packets=192421, n_bytes=12699810,
priority=1,tcp,nw_src=192.168.1.1,nw_dst=192.168.1.2,tp_src=5001,tp_dst=37304
actions=output:"s1-eth2"
 cookie=0x0, duration=31.495s, table=0, n_packets=43, n_bytes=4309, priority=0
actions=CONTROLLER:65535
abakalyn@test-vm:~$
```

Поля відповідності використовуються для порівняння з пакетами. Вони включають порт входу та заголовки пакетів, а також, за потреби, інші поля конвеєра, наприклад, метадані, вказані попередньою таблицею.

Узгодження OpenFlow

При отриманні пакета комутатор OpenFlow виконує такі функції:

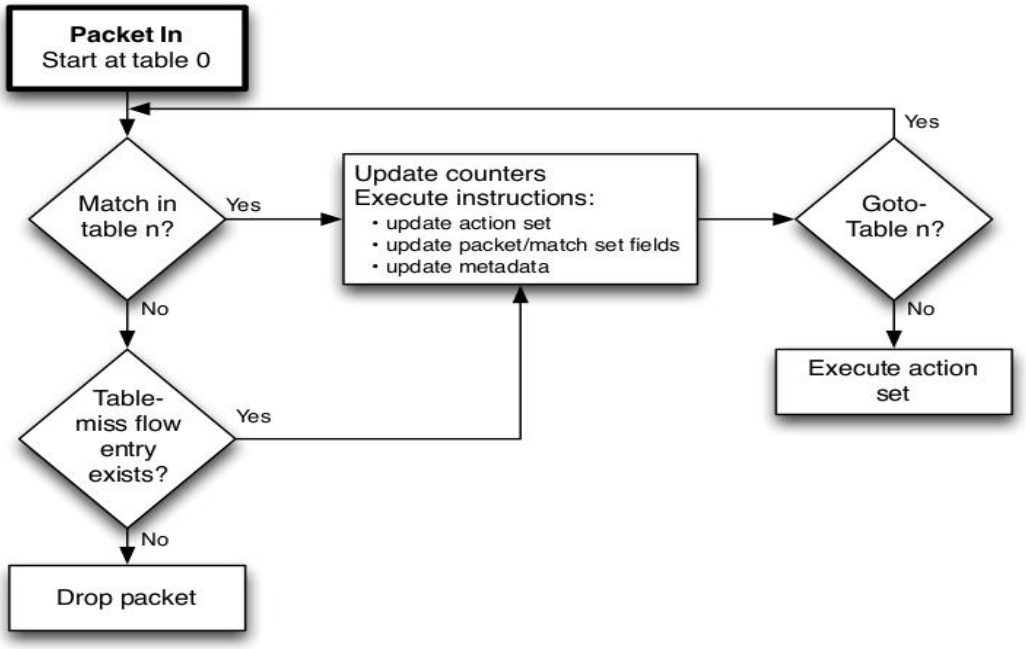


Рис.3.2.Схема руху пакетів через OpenFlow комутатор

Комутатор розпочинає з пошуку в першій таблиці потоків, і в залежності від обробки конвеєра, може виконувати пошук у інших таблицях потоків.

```
Switch input port. */
Switch physical input port. *
Ethernet destination address. */
Ethernet source address. */
Ethernet frame type. */
VLAN id. */
VLAN priority. */
IP DSCP (6 bits in ToS field). */
IP ECN (2 bits in ToS field). */
IP protocol. */
IPv4 source address. */
IPv4 destination address. */
TCP source port. */
TCP destination port. */
UDP source port. */
UDP destination port. */
SCTP source port. */
SCTP destination port. */
ICMP type. */
ICMP code. */
ARP opcode. */
ARP source IPv4 address. */
ARP target IPv4 address. */
ARP source hardware address. */
ARP target hardware address. */
IPv6 source address. */
IPv6 destination address. */
IPv6 Flow Label */
ICMPv6 type. */
ICMPv6 code. */
Target address for ND.
Source link-layer for ND. */
Target link-layer for ND. */
MPLS label. */
MPLS TC. */
MPLS BoS bit. */
PBB I-SID. */
Logical Port Metadata. */
IPv6 Extension Header pseudo-field */
```

• Поля відповідності існують у двох типах: поля відповідності заголовків і поля відповідності конвеєра.

• Поля відповідності заголовків - це поля відповідності, що відповідають значенням, витягнутим із заголовків пакетів. Більшість полів відповідності заголовків прямо

відображаються на конкретне поле в заголовку пакета, визначене протоколом Datapath.

• Усі поля відповідності заголовків мають різний розмір, передумови та можливості маскування.

• Поля відповідності конвеєра - це поля відповідності, які відповідають значенням, прикріпленим до пакета для обробки конвеєром та не пов'язані з заголовками пакета, такі як META_DATA, TUNNEL_ID.

Кожен запис потоку має пов'язані з собою час очікування (idle_timeout) та жорсткий час очікування (hard_timeout).

```
 cookie=0x0, duration=7.619s, table=0, n_packets=3, n_bytes=238, idle_timeout=10,
hard_timeout=30, priority=1,in_port="s1-
eth2",dl_src=00:00:00:00:11:12,dl_dst=00:00:00:00:11:11 actions=output:"s1-eth1"
 cookie=0x0, duration=7.605s, table=0, n_packets=2, n_bytes=140, idle_timeout=10,
hard_timeout=30, priority=1,in_port="s1-
eth1",dl_src=00:00:00:00:11:11,dl_dst=00:00:00:00:11:12 actions=output:"s1-eth2"
 cookie=0x0, duration=8.652s, table=0, n_packets=33, n_bytes=3527, priority=0
actions=CONTROLLER:65535
```

Hard_timeout: Якщо поле hard_timeout не дорівнює нулю, комутатор повинен зафіксувати час прибуття запису потоку, оскільки в майбутньому може знадобитися видалення цього запису. Ненульове значення поля hard_timeout призводить до видалення запису потоку після вказаної кількості секунд, незалежно від того, скільки пакетів він відповів.

idle_timeout: Якщо поле idle_timeout не дорівнює нулю, комутатор повинен зафіксувати час прибуття останнього пакета, пов'язаного з потоком, оскільки в майбутньому може знадобитися видалення цього запису. Ненульове значення поля idle_timeout призводить до видалення запису потоку, якщо він не відповів жодному пакету протягом вказаної кількості секунд. Комутатор повинен реалізувати збільшення часу життя потоку і видаляти записи потоків з таблиці потоків, коли вичерпується один з їх таймаутів.

Три типи повідомлень:

1. Контролер до комутатора

Повідомлення від контролера до комутатора ініціюються контролером та використовуються для прямого керування або перевірки стану комутатора.

- Feature Request
- Packet Out
- Modify Flow Table
- Modify Group Table
- Modify Meter Table
- OpenFlow Switch Description Request
- OpenFlow Port Description Request
- OpenFlow Statistics Request (Flow, Port, Flow table, Aggregate, Group, Meter, Queue)
- Role Request
- Barrier Request

2 Асинхронні

Асинхронні повідомлення ініціюються комутатором і використовуються для оновлення контролера про події в мережі та зміни стану комутатора.

- Packet In
- Flow Removed

2 Симетричні

Симетричні повідомлення ініціюються як комутатором, так і контролером і відправляються без запиту.

- Hello Message
- Echo Message

Повідомлення під час Налаштування Топології

1. Hello
2. Feature request/Response
3. Switch/Port Description Request/Response

4. Modify Flow Entry (To install table Miss entry)
5. Packet IN (Switch to Controller)
6. Packet Out (Controller to Switch)
7. Modify Flow Entry (Install a flow)
8. Echo

Повідомлення "Hello":

Повідомлення "Hello" обмінюються між комутатором та контролером під час запуску з'єднання.

• Комутатор надсилає повідомлення Hello OpenFlow (включає номер версії) контролеру.

• Контролер відповідає повідомленням Hello, якщо версія підтримується.

• Якщо між контролером та комутатором використовується різна версія OpenFlow, повідомлення Hello буде неуспішним. В контролері буде відображено подібне повідомлення про помилку.

Повідомлення "Echo":

Повідомлення запиту/відповіді на ехо можуть надсилатися як від комутатора, так і від контролера і повинні повертати відповідь на ехо. Вони використовуються головним чином для перевірки з'єднання між контролером та комутатором і можуть бути використані для вимірювання його затримки або пропускної здатності. (За замовчуванням: інтервал 5 секунд)

1 . Комутатор надсилає запит на ехо контролеру.

2 . Контролер відповідає відповіддю на ехо.

3.5 Порти OpenFlow:

Фізичні порти OpenFlow - це порти, визначені комутатором, які відповідають апаратному інтерфейсу комутатора. У віртуалізованому середовищі це представляє віртуальний інтерфейс.

Наприклад: "s1-eth1"

Логічні порти - це абстракції на вищому рівні, які можуть бути визначені в комутаторі за допомогою методів, що не відносяться до OpenFlow (наприклад, групи агрегації посилань, тунелі, петлеві інтерфейси).

Наприклад: "vxlan0"

Зарезервовані порти OpenFlow визначені цим специфікацією. Вони вказують на загальні дії пересилання, такі як відправлення до контролера, розповсюдження або пересилання за допомогою методів, що не відносяться до OpenFlow, наприклад, "нормальна" обробка комутатора. FLOOD, ALL, CONTROLLER, IN PORT, LOCAL, NORMAL.

Розділ 4: Контролери

4.1 Контролер Ryu

Ryu — це програмно-визначений мережевий фреймворк на основі компонентів. Ryu надає програмні компоненти з чітко визначеним API, які полегшують розробникам створення нових програм для управління мережею та контролю. Ryu підтримує різні протоколи для управління мережевими пристроями, такі як OpenFlow, OVSDB (Open vSwitch Database), BGP (Border Gateway Protocol). Що стосується OpenFlow, то Ryu повністю підтримує розширення 1.0, 1.2, 1.3, 1.4, 1.5 та Nicira. Весь код знаходиться у вільному доступі під ліцензією Apache 2.0.

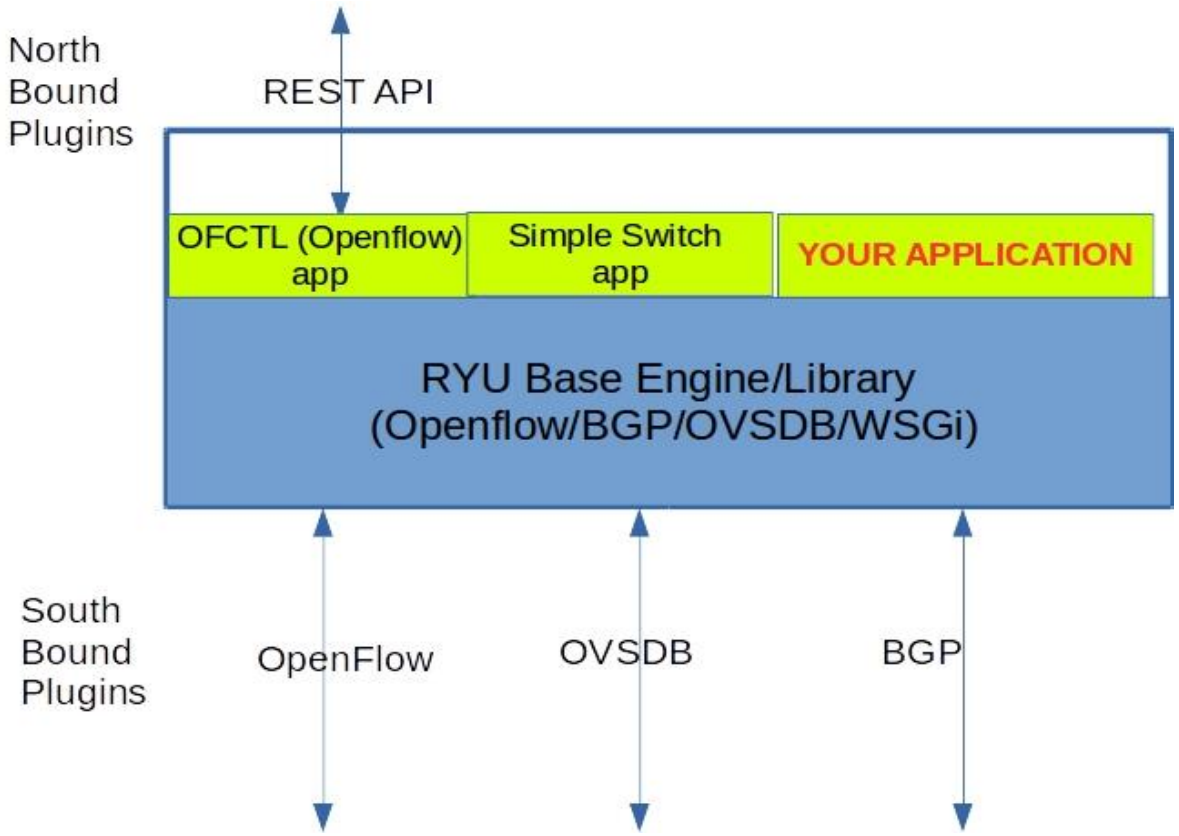


Рис.4.1.Архітектура RYU

Для запуску додатку RYU виконайте наступну команду в терміналі:

ryu-manager ryu.app.application-name

Наприклад:

ryu-manager ryu.app.simple_switch_13

```
abakalyn@test-vm:~$ ryu-manager ryu.app.simple_switch_13
loading app ryu.app.simple_switch_13
loading app ryu.controller.ofp_handler
instantiating app ryu.app.simple_switch_13 of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
```

Перевірка стану порту OpenFlow

RYU Manager слухає порти OpenFlow (6653).

netstat -ap | grep 6653

```
abakalyn@test-vm:~$ netstat -ap | grep 6653
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:6653            0.0.0.0:*               LISTEN      19190/python
suresh@suresh-vm:~$
```

4.2 Реактивні/проактивні потоки

Реактивні потоки:

- Коли новий пакет потрапляє до комутатора і не відповідає жодному існуючому потоку, комутатор відправляє його до контролера.
- Контролер перевіряє пакет і створює логіку.
- Встановлює потік для цієї сесії (відповідність) в комутаторі, використовуючи повідомлення Packet IN / Packet Out.

Проактивні потоки:

Контролер OpenFlow заздалегідь встановить таблиці потоку для всіх збігів трафіку.

Просте проактивне застосування концентратора:

Запускаємо топологію Mininet

sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4

```
abakalyn@test-vm:~/sdn$ sudo mn --controller=remote,ip=127.0.0.1 --mac --
switch=ovsk,protocols=OpenFlow13 --topo=single,4
[sudo] password for abakalyn:
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** 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 ...
*** Starting CLI:
mininet>
```

Запускаємо програму RYU hub

ryu-manager hub.py

```
abakalyn@test-vm:~/sdn$ ryu-manager hub.py
loading app hub.py
loading app ryu.controller.ofp_handler
instantiating app ryu.controller.ofp_handler of OFPHandler
instantiating app hub.py of SimpleSwitch13
```

Перевіряємо потоки

sudo ovs-ofctl -O OpenFlow13 dump-flows s1

```
abakalyn@test-vm:~/sdn$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
[sudo] password for abakalyn:
 cookie=0x0, duration=16.055s, table=0, n_packets=5, n_bytes=350, priority=0
actions=FLOOD
adakalyn@test-vm:~/sdn$
```

Виконуємо пінг між хостами в Mininet

```
mininet> pingall
*** 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)
mininet>
```

Знову спостерігаємо за потоками

```
abakalyn@test-vm:~/sdn$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=654.092s, table=0, n_packets=72, n_bytes=5040, priority=0
actions=FLOOD
abakayn@test-vm:~/sdn$
```

4.3 Відкриття топології

RYU має механізм виявлення топології. RYU використовує LLDP (Link Layer Discovery Protocol) для виявлення комутаторів і зв'язків. Можливість виявлення топології буде увімкнена за допомогою параметра "--observe-links".

Функція виявлення топології використовується в багатьох додатках, таких як визначення найкоротшого шляху, уникнення циклів, багатопроменевість з балансуванням навантаження тощо.

Виявлення комутаторів та зв'язків:

• Контролер RYU генерує пакети LLDP (Datapath ID, номер порту) для кожного комутатора та кожного номера порту.

• Пакети LLDP досягають іншого кінця з'єднання (іншого порту комутатора).

• Цей пакет надсилається до контролера.

• Контролер розшифровує пакет та отримує інформацію про комутатор та зв'язок з пакета LLDP (дані відправника (Datapath ID, номер порту) та дані отримувача (Datapath ID, номер порту)).

Виявлення хостів базується на вхідних пакетах. Зазвичай, машини-хости продовжують відправляти деякі широкомовні/мультимедійні пакети (деякі служби викликають це). Контролер RYU використовує ці пакети для виявлення хостів.

Процес виявлення топології може зайняти кілька секунд. Тому ми використовуємо потік RYU та виводимо інформацію про топологію через 10 секунд. Ми припускаємо, що протягом цих 10 секунд процес виявлення топології буде завершено.

Додаємо заголовок:

```
from ryu.topology.api import get_switch, get_link, get_host
```

Ініціюємо потік:

```
    hub.spawn(self.myfunction)
```

Виведення інформації про виявлення топології:

```
  def myfunction(self):
    self.logger.info("started new thread")
    hub.sleep(10)

    switch_list = get_switch(self.topology_api_app, None)
    self.switches = [switch.dp.id for switch in switch_list]
    links_list = get_link(self.topology_api_app, None)
    self.links = [(link.src.dpid, link.dst.dpid, {'port': link.src.port_no}) for link in links_list]
    host_list = get_host(self.topology_api_app, None)
    self.hosts = [(host.mac, host.port.dpid, {'port': host.port.port_no}) for host in
host_list]
    self.logger.info("*********Topology Information************")
    self.logger.info("Switches %s", self.switches)
    self.logger.info("Links %s", self.links)
    self.logger.info("Hosts %s", self.hosts)
```

Використовуємо три API: get_switch, get_link, get_host. Ці API надають знайдені деталі про комутатор, зв'язок та хост у списку об'єктів.

У наступному рядку ми обробляємо список комутаторів, витягаючи лише ідентифікатори (id) комутаторів.

```
self.switches = [switch.dp.id for switch in switch_list]
```

Аналогічно до інших зв'язків та хостів.

Запускаємо лінійну топологію Mininet

```
sudo mn --controller=remote,ip=127.0.0.1 --mac -i 10.1.1.0/24 --
switch=ovsk,protocols=OpenFlow13 --topo=linear,4
```

Запускаємо контролер Ryu з параметрами "--observe-links

```
ryu-manager --observe-links topology_discovery.py
```

Інформація про топологію буде надрукована в консолі RYU.

```
*********Topology Information*************
Switches [1, 2, 3, 4]
Links [(2, 3, {'port': 3}), (3, 2, {'port': 2}), (3, 4, {'port': 3}), (2, 1, {'port': 2}), (4, 3, {'port': 2}),
(1, 2, {'port': 2})]
Hosts [('00:00:00:00:00:03', 3, {'port': 1}), ('00:00:00:00:00:02', 2, {'port': 1}),
('00:00:00:00:00:01', 1, {'port': 1}), ('00:00:00:00:00:04', 4, {'port': 1})]
```

4.4 Мультиконтролери

Відмова контролера в мережі SDN є серйозною проблемою, що впливає на всю топологію мережі. Різноманітні причини відмови контролера: загрози безпеці, апаратні/програмні проблеми, помилки людини і т.д. Використання одного контролера в мережі SDN - це великий ризик, висока ймовірність відмов, не масштабоване і не має високої стійкості до помилок.

Специфікація OpenFlow підтримує середовище Multiple controllers. Користувач може налаштувати кілька контролерів у комутаторі.

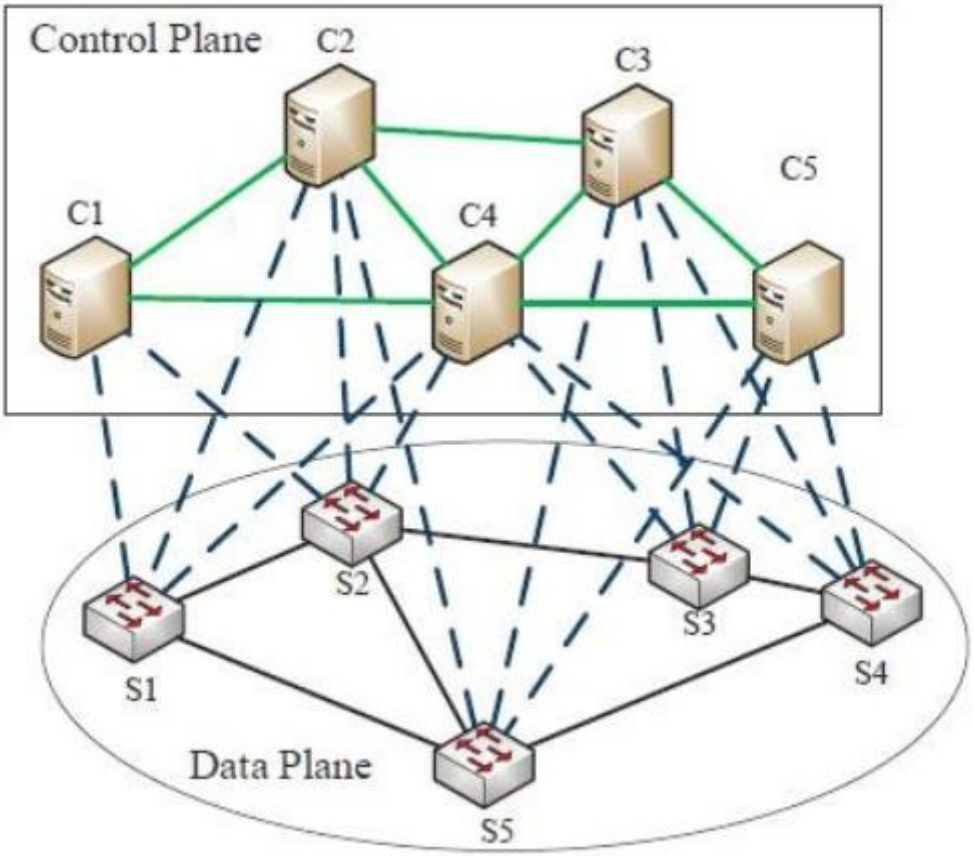


Рис.4.2.Архітектура з використанням мультиконтроллерів

4.5 Ролі контролерів

Рівна роль означає, що всі контролери, налаштовані на комутаторі, мають повний контроль для оновлення/зміни потоків. Комутатор повинен надсилати повідомлення PACKET IN до всіх контролерів. Також комутатор обробляє повідомлення PACKET OUT, ОНОВЛЕННЯ ПОТОКУ та інше від усіх контролерів.

Контролер 1

```
ryu-manager --ofp-tcp-listen-port 6653 flow_timeout.py
```

Контролер 2

```
ryu-manager --ofp-tcp-listen-port 6654 flow_timeout.py
```

Топологія в mininet

```
sudo python topology.py
```

Запускаємо захоплення Wireshark для трасування пакетів.

Pingall

Ми можемо спостерігати, що пакети приходять з обох контролерів.

Роль ведучого означає, що контролер, який є майстром, буде відповідальним за управління планом даних OpenFlow комутатора. Комутатор буде надсилати контрольні повідомлення лише майстер-контролеру.

Роль підлеглого полягає в тому, що контролер виконує роль резервного для майстра. Він також отримує повідомлення HELLO і KEEPALIVE. Проте він не може надсилати та отримувати контрольні повідомлення.

Коли контролер стартує, він надсилає запит на призначення ролі (ROLE REQUEST) з роллю MASTER, інші контролери повинні відправити роль SLAVE. Комутатор буде спілкуватись з MASTER.

Один контролер буде виступати у ролі ведучого (Master Role), інший - у ролі підлеглого або резервного (Slave або Backup Role).

1. Запустіть захоплення Wireshark для трасування пакетів на інтерфейсі замикання (loopback interface).
2. Mininet topology

```
sudo python topology.py
```

3. Запустити -контролер.

```
ryu-manager --ofp-tcp-listen-port 6653 l3switch_master.py
```

4. Запустіть резервний контролер.

```
ryu-manager --ofp-tcp-listen-port 6654 l3switch_slave.py
```

5. Перевіряємо траси в терміналах контролерів. Бачимо, що тільки на терміналі MASTER є траси пакетів.
6. Виконуємо команду "pingall".
7. Тільки контролер MASTER керує комутаторами.
8. Аналізуємо повідомлення ROLE в трасах Wireshark.

У середовищі з кількома контролерами (зокрема, головний/підлеглий), дані (побудовані контролером, наприклад, структура mac_to_port) потрібно синхронізувати між контролерами.

Існують різноманітні механізми, такі як бази даних, бази даних в пам'яті, посередники повідомлень тощо.

РОЗДІЛ 5:

ВПРОВАДЖЕННЯ ТА ТЕСТУВАННЯ

Нам потрібно проводити різноманітні тести трафіку між хостами для перевірки топології мережі та продуктивності застосунків SDN.

Деякі з тестів включають:

1. TCP-тести між двома хостами.

2. Тести UDP.

3. Тести UDP з різним розміром пакетів (64 байти, 1024 байти і т. д.), різною швидкістю пакетів (50 пакетів/с, 100 пакетів/с і т. д.).

4. Змінна швидкість трафіку (10 пакетів/с протягом перших 60 с, потім 100 пакетів/с протягом решти тесту).

5. Потокове відео з використанням відеофайлу mp4 як джерела відео. Буде використано VLC для генерації відеопотоку, який буде приймати клієнт за допомогою VLC Video Player.

5.1 Тести TCP

IPERF - широко використовуваний інструмент для генерації трафіку для проведення TCP-тестів. Тести TCP, як правило, використовуються для вимірювання пропускної здатності. Затримка, джиттер, втрати пакетів не можуть бути виміряні за допомогою тестів TCP. Для всіх наших тестів трафіку ми будемо використовувати просту топологію та додаток l4 switch.

1. Створення топології. (Рис.5.1)

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13
--topo=single,4
```

2. Запуск додатоку l4 switch контролера RYU.

```
ryu-manager l4_switch.py
```

3. У оболонці Mininet виконуємо команду ping h1 до h2.

```
mininet>h1 ping -c 5 h2
```

4. Перевіряємо потоки.

```
sudo ovs-ofctl -O OpenFlow13 dump-flows s1
```

5. Подивитися номери портів хоста та комутатора.

```
mininet>links
mininet>ports
```

6. Перевірка статистики комутатора.

```
sudo ovs-ofctl -O OpenFlow13 dump-ports s1
```

Тепер все готово для запуску наших тестів трафіку.

Тест трафіку з h1 до h4:

Мета: Генерувати TCP-трафік з h1 до h4. (Виміряти пропускну здатність від h1 до h4)

h1 - відправник.

h4 - отримувач.

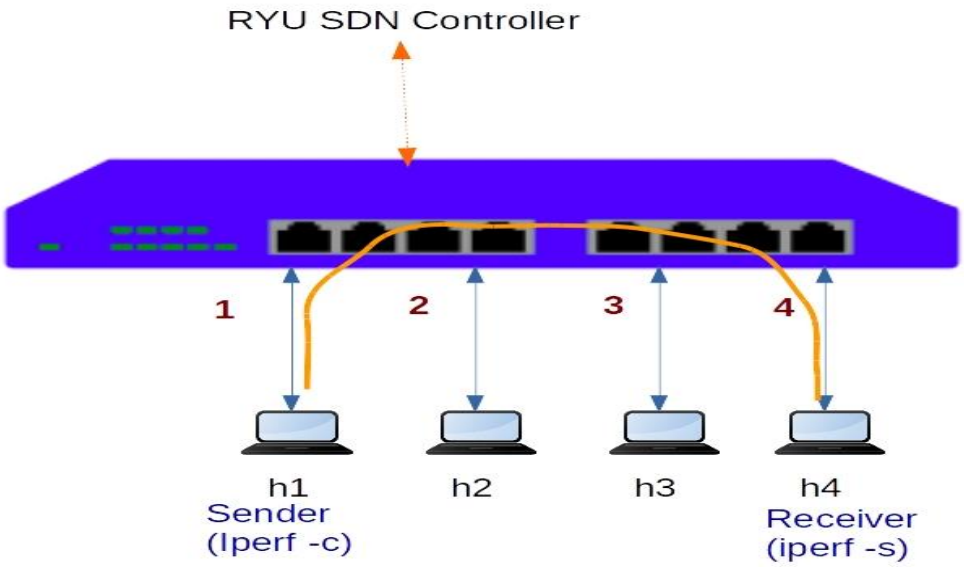


Рис.5.1.Схема для тестів TCP/UDP

1. Запустіть сервер IPERF у h4

```
h4 iperf -s
```

2. Запустіть клієнт IPERF у h1 і підключіться до h4

```
h1 iperf -c h4
```

3. Проаналізуйте результати за потоками.

```
sudo ovs-ofctl -O OpenFlow13 dump-flows s1
```

Ми спостерігаємо трафік у обох напрямках.

Трафік з h1 до h4 дуже великий (в гігабітах на секунду). Це трафік даних.

```
Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 15 second test, tos 0
[ ID] Interval           Transfer     Bitrate
[  5]   0.00-1.00   sec   117 MBytes   981 Mbits/sec
[  5]   1.00-2.00   sec   113 MBytes   949 Mbits/sec
[  5]   2.00-3.00   sec   113 MBytes   949 Mbits/sec
[  5]   3.00-4.00   sec   113 MBytes   949 Mbits/sec
[  5]   4.00-5.00   sec   113 MBytes   949 Mbits/sec
[  5]   5.00-6.00   sec   113 MBytes   949 Mbits/sec
[  5]   6.00-7.00   sec   113 MBytes   949 Mbits/sec
[  5]   7.00-8.00   sec   113 MBytes   949 Mbits/sec
[  5]   8.00-9.00   sec   113 MBytes   949 Mbits/sec
[  5]   9.00-10.00  sec   113 MBytes   949 Mbits/sec
[  5]  10.00-11.00  sec   113 MBytes   948 Mbits/sec
[  5]  11.00-12.00  sec   113 MBytes   950 Mbits/sec
[  5]  12.00-13.00  sec   113 MBytes   949 Mbits/sec
[  5]  13.00-14.00  sec   113 MBytes   949 Mbits/sec
[  5]  14.00-15.00  sec   113 MBytes   949 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
Test Complete. Summary Results:
[ ID] Interval           Transfer     Bitrate
[  5]   0.00-15.00  sec  1.66 GBytes   951 Mbits/sec                  sender
[  5]   0.00-15.04  sec  1.66 GBytes   949 Mbits/sec                  receiver
CPU Utilization: local/sender 1.1% (0.3%u/0.8%s), remote/receiver 9.0% (2.5%u/6.5%s)

iperf Done.
```

Трафік з h4 до h1 дуже малий. Це трафік підтвердження TCP.

4. Проаналізуйте результати за портами.

```
sudo ovs-ofctl -O OpenFlow13 dump-ports s1
```

h1 ---port1, port4---h4

Прямий трафік:

• h1 передає. Порт1 отримує.

• Порт4 передає, h4 отримує.

Підтвердження:

• h4 передає. Порт4 отримує.

• Порт1 передає. h1 отримує.

5.2 Тести UDP

UDP-тести є досить гнучкими за своєю природою, оскільки вони обходять розміри пакетів (64 байти) та кількість пакетів (10 пакетів/с), які ви хочете надіслати. Це означає, що користувач може контролювати пропускну здатність UDP-трафіку.

Пропускну здатність, затримку, джиттер, втрату пакетів можна виміряти за допомогою UDP-тестів.

IPERF не забезпечує особливої гнучкості в тестах UDP.

1. Запускаємо IPERF UDP Server в h3

```
h3 iperf -u -s
```

2. Запустіть клієнт IPERF на h1, підключившись до h4 та генеруючи пропускну здатність 10 Мбіт/с.

```
h1 iperf -u -c h3 -b 10m
```

```
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 5001 connected with 10.0.0.3 port 35898
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[ 23]  0.0- 1.0 sec  1.11 MBytes  9.34 Mbits/sec   4.400 ms    0/  794 (0%)
[ 23]  1.0- 2.0 sec  1.13 MBytes  9.44 Mbits/sec   3.620 ms    0/  803 (0%)
[ 23]  2.0- 3.0 sec  1.11 MBytes  9.30 Mbits/sec   1.433 ms    0/  791 (0%)
[ 23]  3.0- 4.0 sec  1.11 MBytes  9.35 Mbits/sec   1.477 ms    0/  795 (0%)
[ 23]  4.0- 5.0 sec  1.13 MBytes  9.51 Mbits/sec   4.383 ms    0/  809 (0%)
[ 23]  5.0- 6.0 sec  1.14 MBytes  9.57 Mbits/sec   2.398 ms    0/  814 (0%)
[ 23]  6.0- 7.0 sec  1.12 MBytes  9.43 Mbits/sec   2.366 ms    0/  802 (0%)
[ 23]  7.0- 8.0 sec  1.11 MBytes  9.35 Mbits/sec   3.131 ms    0/  795 (0%)
[ 23]  8.0- 9.0 sec  1.14 MBytes  9.54 Mbits/sec   1.795 ms    0/  811 (0%)
[ 23]  9.0-10.0 sec  1.14 MBytes  9.54 Mbits/sec   1.402 ms    0/  811 (0%)
[ 23]  0.0-10.1 sec  11.3 MBytes  9.43 Mbits/sec   4.604 ms    0/ 8075 (0%)
```

5.3 Тестування трафіку відеопотоку

Я використовую медіаплеєр VLC для тестування потокового відео. (Рис.5.2)

Порядок встановлення

```
sudo apt-get install vlc vlc-nox
```

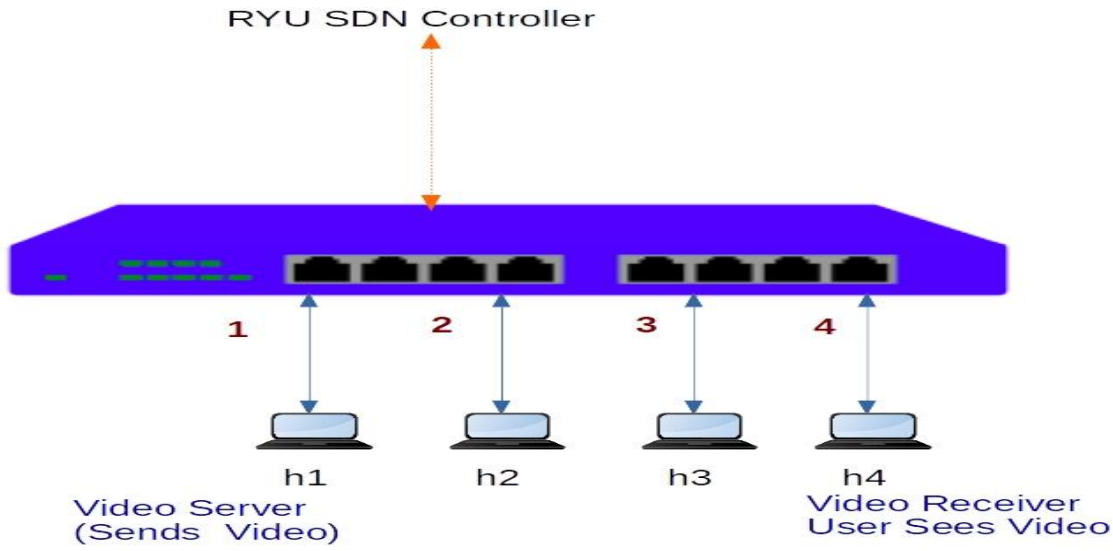


Рис.5.2.Схема для тесту стрімингу відео

1. Створіть топологію.

```
sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13
--topo=single,4
```

2. Запустіть додаток l4 switch контролера RYU.

```
ryu-manager l4_switch.py
```

3. Завантажте відео зразка з Інтернету (або використовуйте свій власний відеофайл).
4. h4 є приймачем відео. Запустіть приймач відео на оболонці h4 xterm.

```
mininet> xterm h4

su - user-name

vlc rtp://@10.0.0.4:9001
```

5. h1 - це відправник відео. Запустіть відправника стрімінгу відео в оболонці h1 xterm.

```
mininet> xterm h1

su - user-name

cvlc  /home/abakalyn/cat3.mp4  --sout '#rtp{proto=udp,mux=ts,dst=10.0.0.4,port=9001}'
```

5.4 Встановлення

Для використання Ryu з Mininet ми:

• Встановлюємо Ryu і Mininet на нашу систему.

• Створюємо топологію за допомогою Mininet. Це можна зробити, написавши скрипт на Python, який визначає топологію, таку як кількість та типи комутаторів і хостів.

• Запускаємо Mininet з створеною топологією. Це створить віртуальну мережу, яку можна використовувати для тестування реалізації SDN.

• Пишемо застосунок Ryu, який реалізує бажану функціональність SDN, таку як маршрутизація або балансування навантаження. Це можна зробити за допомогою API Ryu, яке надає набір функцій для управління мережею.

• Запускаємо контролер Ryu і підключаємо його до віртуальної мережі, створеної Mininet. Це дозволить застосунку Ryu контролювати трафік мережі.

• Проведимо тестування вашої реалізації SDN, відправляючи трафік між хостами у мережі Mininet і спостерігаючи поведінку контролера Ryu.

Загалом, використання Ryu з Mininet може бути потужним інструментом для тестування та розробки застосунків SDN у віртуальному середовищі.

1. **Установка тестового середовища SDN**

Щоб встановити Ryu та Mininet на Ubuntu, скористайтеся цією командою:

```
sudo apt update
sudo apt install python3 python3-pip xterm iperf hping3 net-tools wireshark apache2-utils curl
sudo apt install mininet
sudo pip3 install ryu
sudo pip3 install mininet
sudo cp /usr/bin/python3 /usr/bin/python
```

2. **Код Python для Custom Topology:**

Кроки наведені нижче (для 6 хоста, 6 комутаторів і 1 контролера)

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.node import Controller, OVSSwitch

class CustomTopo(Topo):
    def __init__(self):
        Topo.__init__(self)

        # Create switches
        s1 = self.addSwitch('s1', cls=OVSSwitch)
        s2 = self.addSwitch('s2', cls=OVSSwitch)
        s3 = self.addSwitch('s3', cls=OVSSwitch)
        s4 = self.addSwitch('s4', cls=OVSSwitch)
        s5 = self.addSwitch('s5', cls=OVSSwitch)
        s6 = self.addSwitch('s6', cls=OVSSwitch)

        # Create hosts
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')
        h3 = self.addHost('h3')
        h4 = self.addHost('h4')
        h5 = self.addHost('h5')
        h6 = self.addHost('h6')

        # Connect hosts to switches
        self.addLink(h1, s1)
        self.addLink(h2, s2)
        self.addLink(h3, s3)
        self.addLink(h4, s4)
        self.addLink(h5, s5)
        self.addLink(h6, s6)

        # Connect switches to each other
        self.addLink(s1, s2)
        self.addLink(s2, s3)
        self.addLink(s3, s4)
        self.addLink(s4, s5)
        self.addLink(s5, s6)

if name == '__main__':
    topo = CustomTopo()
    net = Mininet(topo=topo, controller=Controller, switch=OVSSwitch)
    net.start()
    net.pingAll()
    net.stop()
topos = {'10': CustomTopo}
```

Запускаємо контролер SDN RYU

```
ryu-manager
ryu.app.simple_switch_13
```

Запускаємо файл топології Mininet

```
sudo mn --custom 10.py --topo 10 --controller remote
```

5.5 Mini-edit

Побудова топології мережі

Крок 1. Швидкий доступ до MiniEdit розташований на Робочому столі комп'ютера. (Рис.5.3) Запустіть MiniEdit, клацнувши на його ярлику.

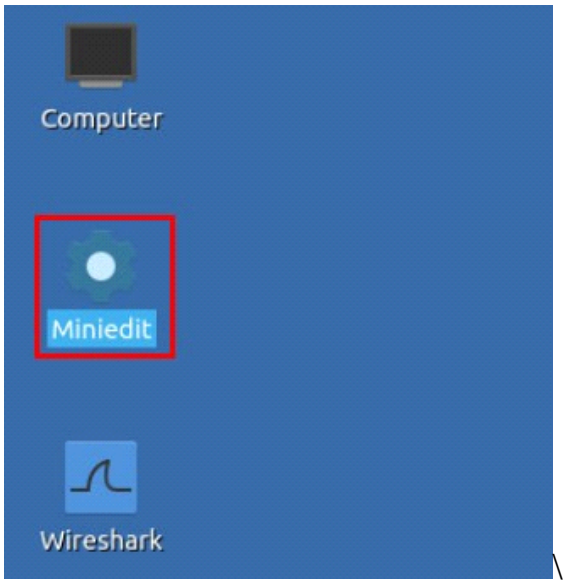


Рис.5.3.Ярлик програми Miniedit

MiniEdit запуститься, як показано нижче. (Рис.5.4)

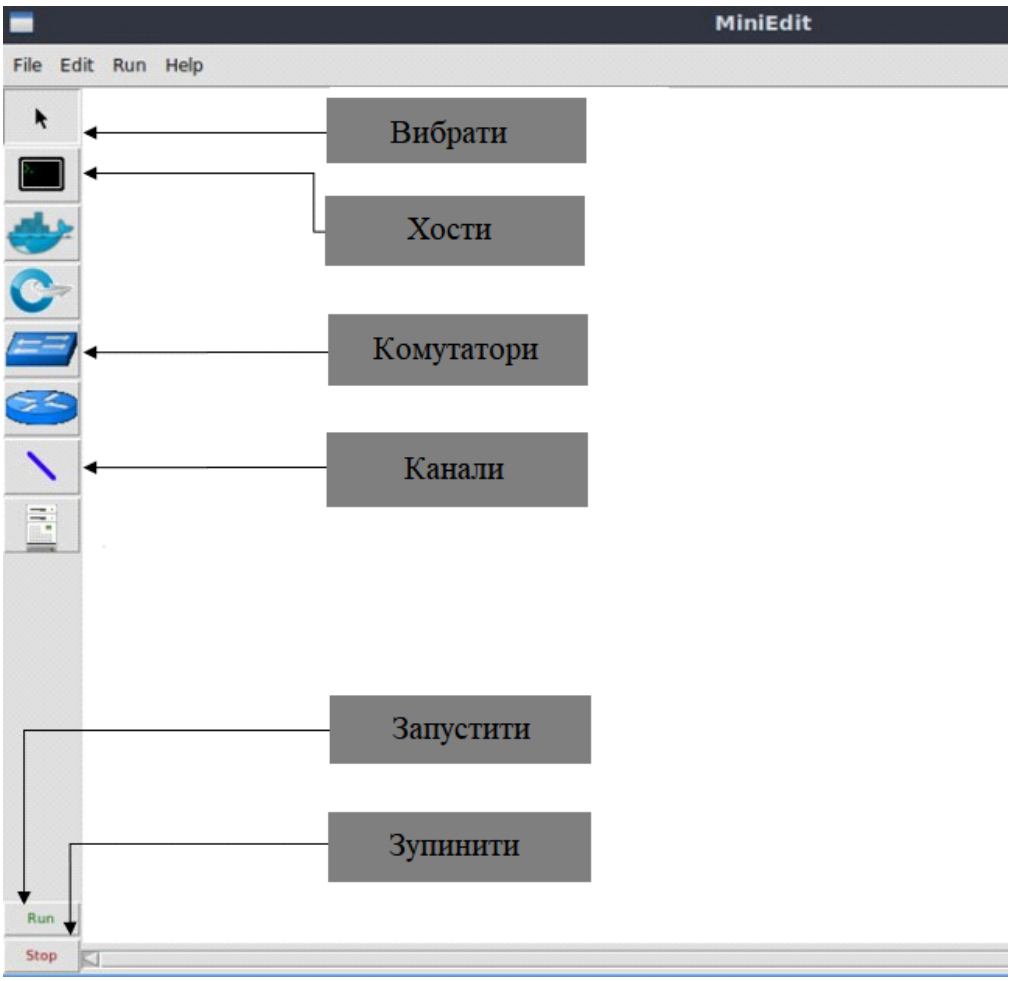


Рис.5.4.Графічний інтерфейс користувача MiniEdit (GUI)

Основні кнопки у цій лабораторії:

• Select: дозволяє вибирати / переміщувати пристрої.

• Host: дозволяє додавати новий хост до топології.

• Legacy switch: дозволяє додавати новий застарілий комутатор до топології.

• Link: з'єднує пристрої в топології (головним чином комутатори та хости).

• Run: запускає емуляцію.

• Stop: зупиняє емуляцію.

Будуємо топологію, із шістьма хостами і шістьма комутаторами, в MiniEdit, як показано нижче. (Рис.5.5)

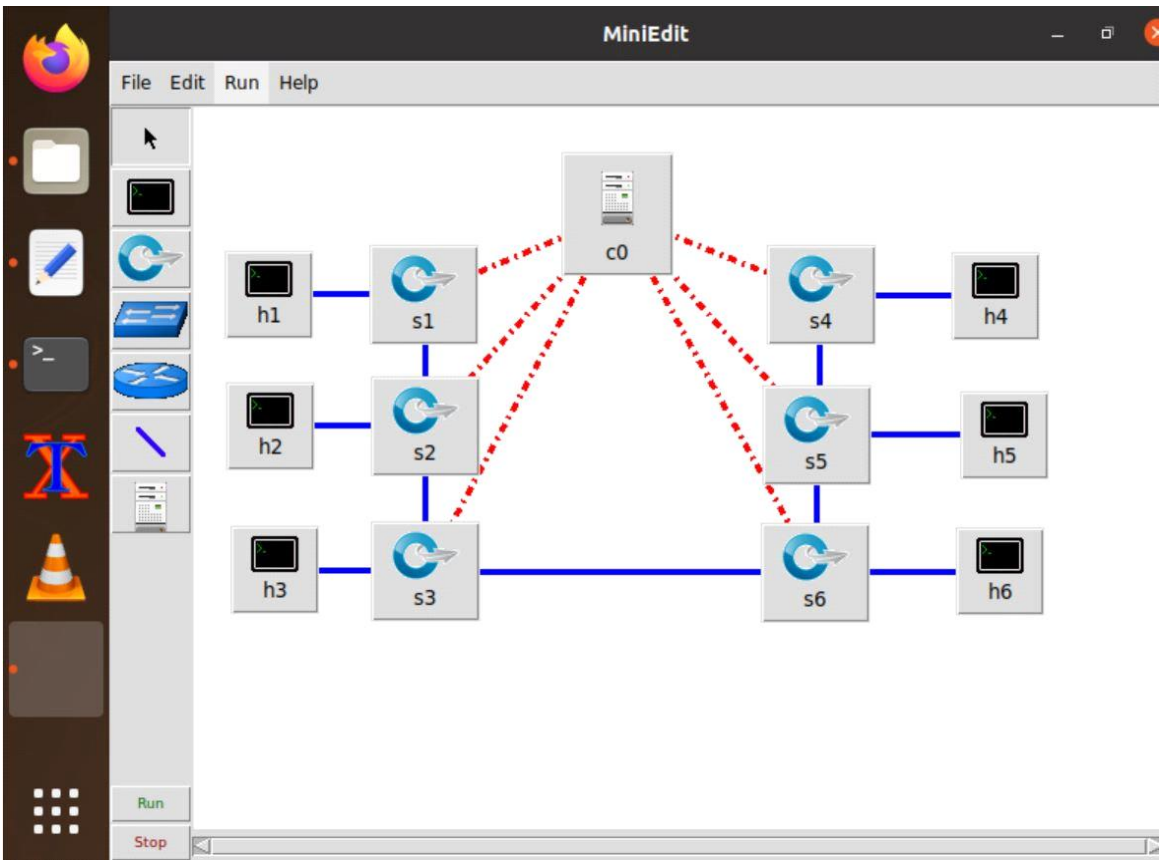


Рис.5.5.Схема топології мережі

Розділ 5.6 Впровадження

Для впровадження потокового відео через SDN ми використовували такі інструменти та технології:

Mininet: емулятор мережі, що дозволив нам створити віртуальну топологію мережі та тестувати нашу SDN-додаток.

VLC Media Player: медіаплеєр, який ми використовували для потокової передачі відео від джерела до приймача.

Контролер SDN Ryu: контролер, який ми використовували для управління мережею та виконання правил OpenFlow на комутаторах.

Для запуску нашого застосунку SDN слід дотримуватись цих кроків:

1) Запускаємо віртуальні машини та комутатори за допомогою програмного забезпечення віртуалізації, такого як VirtualBox або VMware.

2) Налаштовуємо топологію мережі яку буде керувати застосунок SDN, використовуючи клас Custom Topo у коді.

3) Запускаємо програмне забезпечення контролера Ryu, виконавши наступну команду в вікні терміналу:

```
ryu-manager --verbose ryu.app.simple_switch_13
```

4) Розгортаємо застосунок SDN на контролері, виконавши наступну команду в іншому вікні терміналу

```
python sdn_application.py
```

5) Перевіряємо роботу застосунку SDN, відправляючи трафік через мережу та переконуючись, що він правильно керується застосунком. Це можна зробити за допомогою інструментів тестування мережі, таких як ping, або через спеціалізований інструмент для тестування SDN, наприклад, Mininet. Виконуємо наступну команду для перевірки зв'язку між усіма хостами:

```
sudo mn --topo custom --controller remote
```

6) У CLI Mininet виконуємо наступну команду для перевірки зв'язку між h1 та h6:

```
h1 ping h6
```

```
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Отже застосунок SDN правильно керує мережею.

Стрімінг:

У цьому проекті ми плануємо використовувати h1 та h6 для трансляції відео.

Ми відкриваємо 6 вузлів для перевірки підключення за допомогою команди xterm(Рис.5.6)

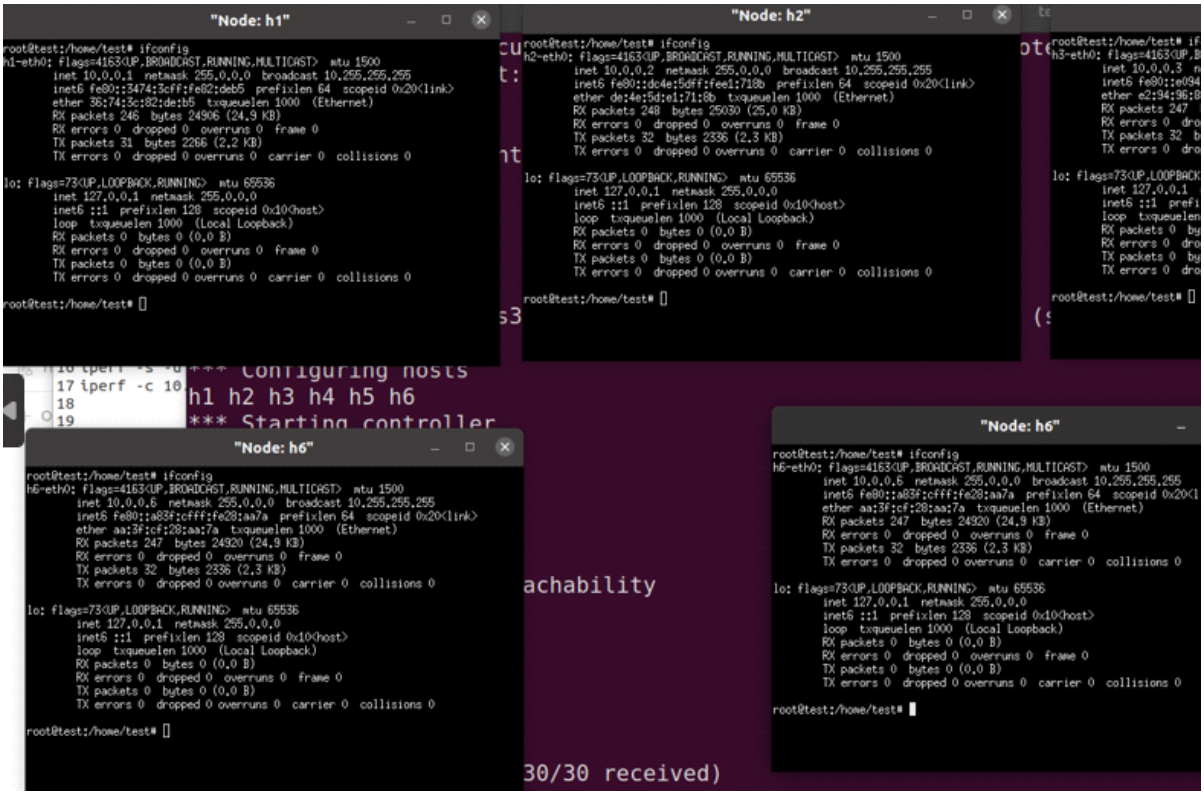


Рис.5.6.Вікна терміналів

Ми використовуємо наступну команду для запуску VLC на хості 1 та 6:

```
vlc-wrapper
```

На хості 1 у VLC Media Player ми клікаємо на меню stream > file > додаємо відео, яке ми хочемо транслювати нитискаємо на "Stream".

У налаштуваннях призначення перевіряємо опцію "Відображати локально" та додаємо "RTP/MPEG Transport Stream". Після цього, натиснувши "Далі", у рядку "Адреса" вводимо адресу, яку ви хочете використовувати для потоку, та порт, до якого підключений хост на комутаторі.

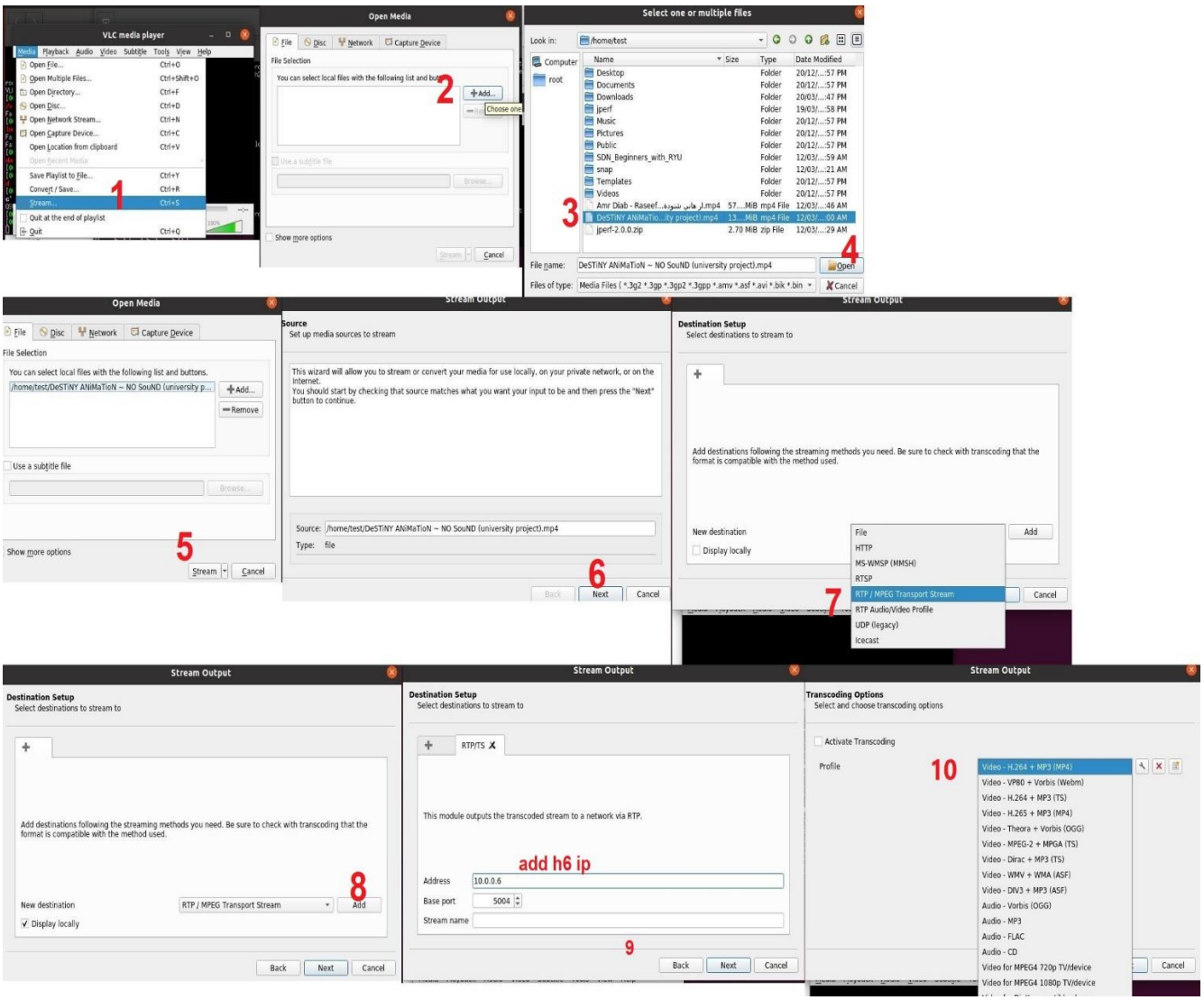


Рис.5.7.Налаштування VLC Media Player

На хості 6 у VLC Media Player ми клікаємо на меню Відкрити потік з мережі в полі URL введимо "rtp://@:5004", переконайтеся, що номер порту співпадає з тим, який використовується на хості 1.

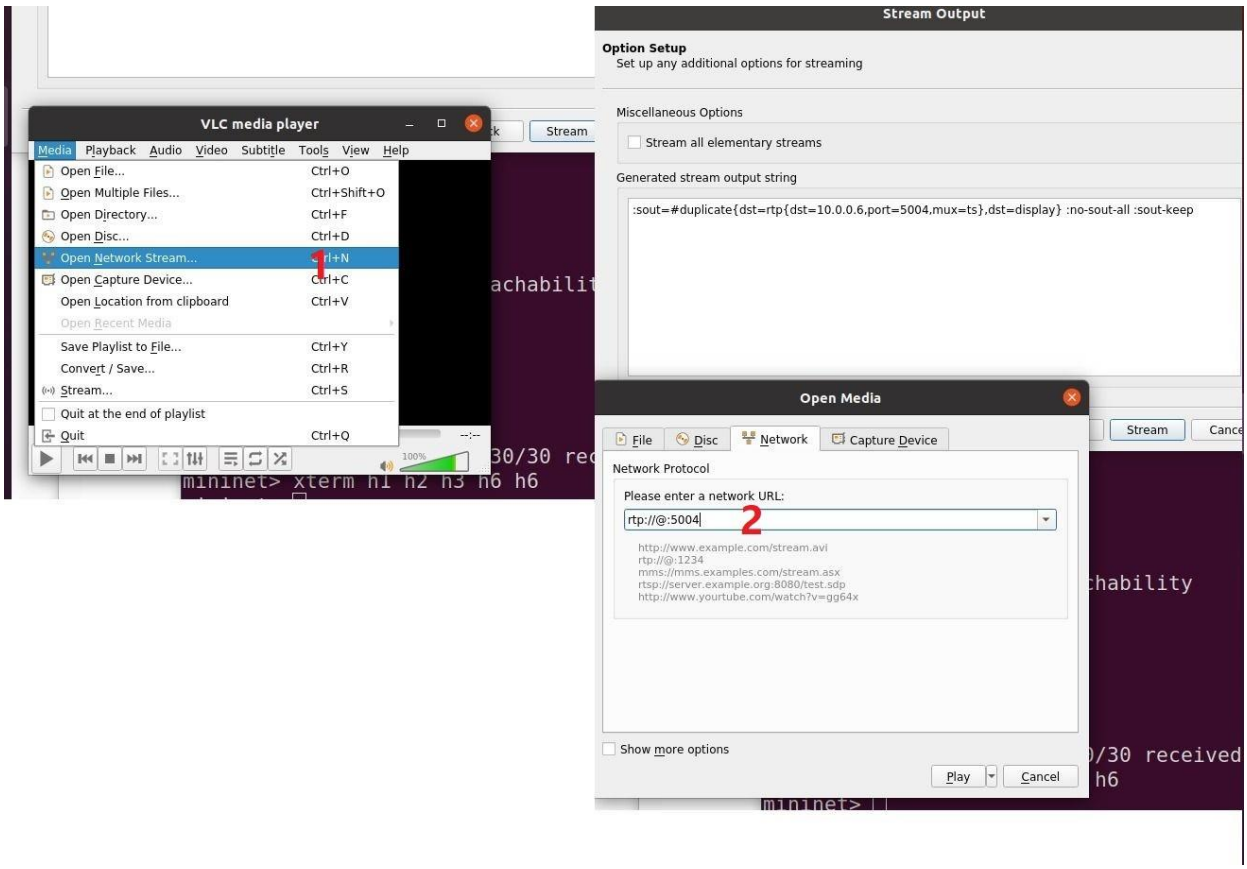


Рис.5.8.Налаштування VLC Media Player

Після виконання цих кроків відео почне транслюватися, і ви зможете переглянути його на h1 та h6. (Рис.5.9)

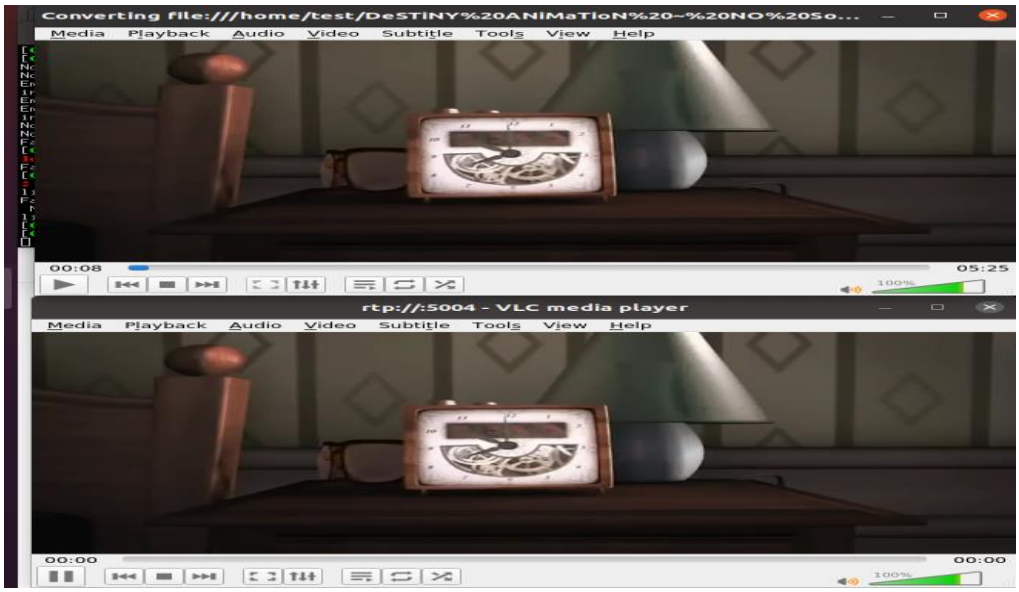
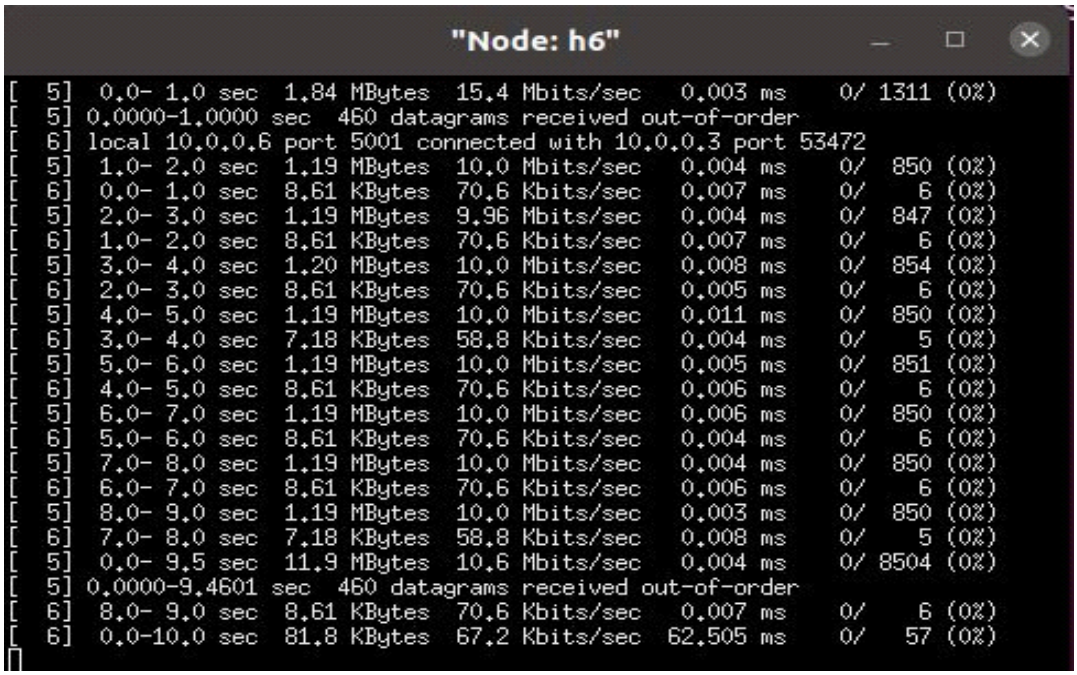


Рис.5.9.Стріминг відео

Тепер ми можемо проаналізувати наш потік і переконатися, що немає затримок і все працює чудово. Нам потрібно зібрати дані, використовуючи iperf. Це буде виглядати приблизно так:



Також наш трафік буде відображатися в програмі Wireshark (Рис.5.10)

Wireshark — це аналізатор мережевих пакетів/протоколів.

Аналізатор мережевих пакетів намагатиметься захопити мережні пакети та намагатиметься відобразити ці пакетні дані якомога детальніше.

Wireshark, мабуть, один з кращих аналізаторів пакетів з відкритим вихідним кодом, доступних на сьогоднішній день для UNIX і Windows.

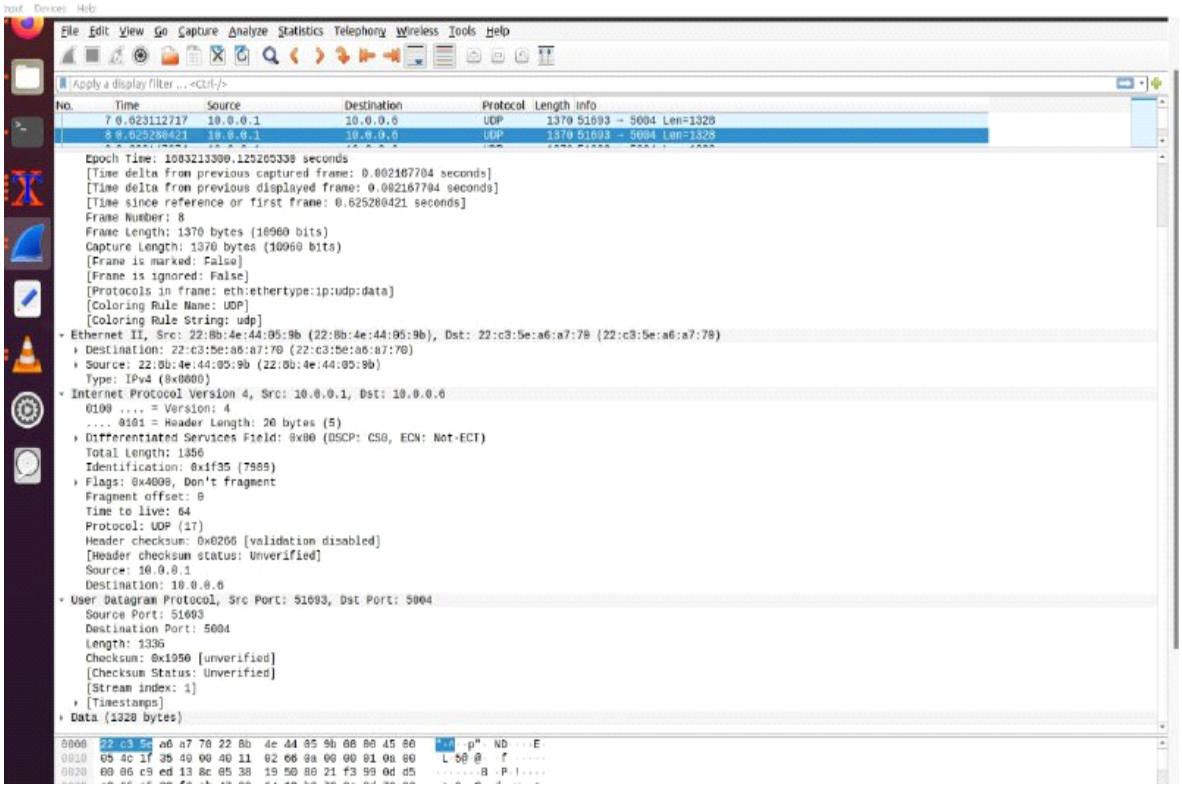


Рис.5.10.Трафік в програмі Wireshark

РОЗДІЛ 6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1. Аналіз впливу техногенних чинників на навколишнє природне середовище

Людська діяльність активно трансформує навколишнє середовище, в результаті чого відбуваються зміни, що спричиняють порушення біосфери та створюють штучне середовище, відоме як техногенна або техносфера. Згідно з науковими даними, майже всі області, де перебуває людина, вже підпадають під вплив цієї техногенної сфери. Людською діяльністю створена техносфера охоплює практично увесь світ і навіть виходить за межі планети у космос.

Техногенне середовище, або техносфера, є наслідком впливу людської діяльності на навколишнє середовище. Вона виникає в результаті антропогенних факторів.

У цьому техногенному середовищі людина постійно виконує дві основні функції:

- Забезпечує своє комфортне перебування у житловому оточенні.
- Створює та використовує системи захисту від негативного впливу техногенного середовища.

Вплив техносфери на навколишнє середовище та організм людини може бути як прямим, так і непрямим, спричиняючи негативні наслідки [8].

6.2. Принцип роботи базових станцій і стільникових пристроїв та їх негативний вплив на довкілля

У сучасному світі, з урахуванням швидкого розвитку технологій, майже неможливо уникнути впливу електромагнітного випромінювання (ЕМП). Телефонні

трубки та базові станції мобільних операторів становлять джерела ЕМП у сфері мобільного зв'язку.

Спосіб впливу цих джерел ЕМП на організм людини різний. Стільникові телефони, як джерела ЕМП, мають особливу рису - максимально наближеність до голови користувача на відстань від двох до п'яти сантиметрів у не контрольованих умовах. ЕМП може впливати на головний мозок, периферичні рецепторні зони вестибулярного та слухового аналізаторів, а також на сітківку очей. Негативні наслідки випромінювання стільникових телефонів можуть також стосуватися оточуючих людей, коли користувач розмовляє по телефону.

Електромагнітні поля, які генеруються базовими станціями, мають імпульсний характер. Це залежить від часу доби, насиченості покриття зони базовими станціями та їх кількості в певній області. Базові станції покривають всю зону функціонування мобільного зв'язку техногенним електромагнітним полем.

Оскільки ці базові станції зазвичай розташовані в місцях, де люди постійно перебувають, це призводить до постійного впливу на людину низькоінтенсивного радіочастотного електромагнітного поля, що діє цілодобово.

Згідно з висновками екологів і лікарів-гігієністів, всі види електромагнітного випромінювання мають вплив на здоров'я та працездатність людей, і можуть мати серйозні наслідки. З огляду на широке поширення, вплив електромагнітних полів на людину є значно більш небезпечним порівняно з радіацією. Електричні поля промислової частоти оточують людину цілодобово через випромінювання від електропроводки, освітлювальних систем, побутових електроприладів, ліній передачі електроенергії та інших джерел.

Енергетичне навантаження від електромагнітних випромінювань постійно зростає у промисловості і в побуті через стрімке розширення джерел фізичних полів електромагнітного походження і збільшення їх потужностей. Людина не може фізично відчути електромагнітне поле, але воно може впливати на зменшення адаптаційних резервів, зниження імунітету, працездатності. Його вплив може спричиняти синдром хронічної втоми та збільшувати ризик захворювань. Особливо

небезпечний цей вплив на дітей, підлітків, вагітних жінок та людей із ослабленим здоров'ям.

Так, електромагнітне поле може впливати на заряджені частинки та струми, викликаючи перетворення енергії поля на інші форми енергії на рівні клітини. Це впливає на фізіологічні процеси та може мати важливі наслідки для клітинних функцій.

Цитогенетичні дослідження, які вивчали хромосомні аберації, показали достовірне збільшення клітин з порушеннями в експериментальній групі у порівнянні з контролем. Також виявлено збільшення хромосомних аберацій при опроміненні електромагнітним полем повітряно-сухого насіння та проростків салату. Аналіз клітин крові корів з ферми показав підвищену кількість генетичних ушкоджень і випадків аномального гематопоезу [9].

Дослідження показали, що навіть слабкі електромагнітні поля, які не викликають теплового ефекту, можуть впливати на зміни в живих тканинах. Особливе увагу приділяли біологічному впливу різних електронних пристроїв, таких як стільникові телефони та комп'ютерні блоки. В ході цих досліджень виявлено, що вплив цих джерел може призводити до погіршення процесу регенерації тканин.

Експерименти, проведені у минулому, підтверджують вплив електромагнітного поля на нервову систему. Дослідження показали, що електромагнітне поле може впливати на різні аспекти нейронної активності, такі як мембрани нейронів, процеси синтезу в нервових клітинах та умовно-рефлекторна діяльність. Експерименти також підтвердили, що дія електромагнітного поля може впливати на процеси передачі інформації в більш складні структури мозку та викликати порушення короткочасної пам'яті.

Електромагнітне випромінювання може мати вплив на імунну систему людини. Накопичені дані свідчать про те, що електромагнітне поле може спричиняти різноманітні порушення у процесах імуногенезу. Зміни в характері інфекційних процесів, розлади білкового обміну, зниження рівня альбумінів і підвищення гамма-глобулінів у крові — це лише деякі з виявлених наслідків впливу

електромагнітного поля на організм. Також відомо, що електромагнітне поле може викликати алергічні реакції або поглиблювати вже існуючі у хворих алергіків при контакті з цим полем.

Електромагнітне поле може впливати на статеву систему людини. Наукові дослідження показали, що під впливом електромагнітного випромінювання може відбуватися зниження функції сперматогенезу у чоловіків та зміни у менструальному циклі у жінок. Також відомо, що електромагнітне випромінювання може уповільнювати ембріональний розвиток та сприяти виникненню вроджених вад у новонароджених. Помітно також зменшення лактації у годуючих матерів.

Електромагнітні поля мають вплив на рослини. Дослідження показують, що як слабкі, так і сильні електромагнітні поля можуть значно змінювати морфологію, фізіологію, біохімію та біофізику багатьох видів рослин. Вони впливають на ріст, розвиток та розмноження рослин.

Ефекти слабких електромагнітних полів на організми вивчаються в різних дослідженнях. Вони показують, що навіть електромагнітні поля з інтенсивністю, що не спричиняє теплового ефекту, можуть впливати на живі тканини. Дослідження, зокрема з використанням стільникових телефонів та комп'ютерів, проводилися в наукових центрах. Вони аналізували шкідливість цих електронних засобів як у робочому, так і вимкненому стані, включаючи й період без живлення[10].

Результати проведених досліджень з оцінки впливу стільникових телефонів, комп'ютерів та інших радіоелектронних засобів на різні організми, як у робочому, так і у вимкненому стані, виявилися тривожними і свідчать про негативний вплив на біологічні об'єкти. Зафіксовані наслідки включають:

- Зниження рухової активності та виживання мікроорганізмів.
- Збільшення смертності мікроорганізмів.
- Погіршення регенерації тканин.
- Порушення ембріонального і личинкового розвитку.
- Зниження біохімічних реакцій та порушення метаболізму.

- Зниження енергетичного потенціалу у всіх життєво важливих системах організму.

6.3.Методи та засоби захисту навколишнього середовища від впливу техногенних чинників

Для зниження впливу електромагнітного випромінювання на персонал та мешканців, що перебувають у зоні дії радіоелектронних засобів, необхідно вжити ряд захисних заходів. Ці заходи можуть включати в себе організаційні, інженерно-технічні та профілактичні заходи, спрямовані на зменшення впливу електромагнітних полів.

Органи санітарного нагляду покладають основну відповідальність за впровадження організаційних та інженерно-технічних заходів. Вони спільно з санітарними лабораторіями підприємств та установ, які використовують електромагнітне випромінювання, вживають заходи для гігієнічної оцінки будівництва та реконструкції об'єктів, які використовують радіозасоби. Також вони контролюють впровадження нових технологічних процесів та обладнання, яке використовує електромагнітне випромінювання. Органи санітарного нагляду здійснюють поточний контроль за об'єктами, які використовують джерела випромінювання, проводять організаційно-методичні заходи з підготовки фахівців та забезпечують інженерно-технічний нагляд.

На етапі проектування важливо врахувати оптимальне розташування джерел електромагнітного випромінювання та об'єктів, які їм піддаються, з метою мінімізації інтенсивності випромінювання. Оскільки повністю уникнути впливу електромагнітного поля неможливо, важливо зменшити можливість доступу людей до зон з високою інтенсивністю ЕМП та скоротити тривалість їх перебування в таких зонах.

Інженерно-технічні методи та засоби захисту грають ключову роль в забезпеченні безпеки від електромагнітного випромінювання. Колективний захист

ґрунтується на аналізі поширення радіохвиль у конкретній місцевості та може охоплювати групу будівель, район чи навіть населений пункт. Економічно доцільним є використання природних бар'єрів, таких як рельєф місцевості, насадження лісу чи будівлі нежитлового призначення, для створення колективного захисту.

Встановлення антени на високогір'ї може значно зменшити інтенсивність поля, яке опромінює населені райони у кілька разів. Схожий ефект можна досягти шляхом коригування діаграми напрямленості антени, особливо високонаправлених, наприклад, за рахунок збільшення її висоти. Однак високі антени є складнішими в установці, вимагають більших витрат і менше стійкі до погодних умов. Також слід зазначити, що ефективність такого захисту зменшується з відстанню.

При використанні екранів для захисту від випромінювання важливо враховувати, як хвиля слабшує під час проходження через екран, наприклад, лісову площу. Рослинний покрив може використовуватись як екранування. Спеціальні конструкції екранів, такі як відбивні або радіо-поглинальні щити, є дорогими, менш ефективними та рідко використовуються.

Локальний захист, який широко використовується, базується на радіозахисних матеріалах. Вони забезпечують високий поглинання енергії випромінювання у матеріалі та відбивають його від поверхні. Металеві листи та сітки з високою провідністю використовуються для відбивання енергії як частина захисних заходів. Застосування металізованих шпалер, захисних сіток на вікнах або металізованих штор допомагає захистити приміщення від зовнішнього випромінювання.

Засоби індивідуального захисту застосовують у випадках, коли інші захисні методи непридатні або менше ефективні: при перетині зон з підвищеною інтенсивністю випромінювання, ремонтних або налагоджувальних роботах у надзвичайних ситуаціях, а також під час короткочасного контролю та зміни інтенсивності опромінення. Однак такі засоби можуть бути незручними у використанні, обмежувати робочі можливості та погіршувати гігієнічні умови.

Одяг, що використовується для захисту тіла, містить металізовані тканини та матеріали, що поглинають радіовипромінювання. Металізовані тканини складаються з бавовняних або капронових ниток, обгорнутих металевим дротом у формі спіралі. Ця тканина, схожа на металеву сітку, здатна зменшити вплив випромінювання на 20-30 децибел. При зшиванні деталей захисного одягу, важливо забезпечити ізольований контакт провідників. Для цього використовують електропровідні розчини чи клеї, що забезпечують герметизацію швів і створюють гальванічний контакт або підвищують ємнісний зв'язок між провідниками, які не мають контакту.

Очі захищаються спеціальними окулярами, які мають скло з нанесеною внутрішньою провідною плівкою з двоокису олова. Гумова оправа окулярів може мати запресовану металеву сітку або бути обклеєною металізованою тканиною. Використання цих окулярів дозволяє зменшити вплив випромінювання у діапазоні надвисоких частот на 20-30 децибел, що забезпечує додатковий захист очей.

Раніше рекомендували використовувати рукавички та бахіли для захисту від електромагнітних випромінювань. Проте сучасні стандарти вважають їх непотрібними, оскільки допустимі значення щільності енергетичного потоку для рук і ніг значно вищі, ніж для загального тіла.

Колективні та індивідуальні засоби захисту можуть забезпечити тривалу та безпечну роботу персоналу на об'єктах, де працюють з радіоелектронними системами.[11]

Боротьба з шумом в джерелі його виникнення є найбільш ефективним способом. Це означає створення малошумних механічних передач та розробку методів для зменшення шуму в підшипникових вузлах та вентиляторах. Такі інновації спрямовані на зменшення шумового викиду вже на самому початку його виникнення, що сприяє створенню менш шумних умов на робочих майданчиках.

Зниження шуму через звукопоглинання полягає у розміщенні шумових джерел всередині оболонки, де внутрішні стінки обробляються спеціальним звукопоглинальним матеріалом. Ця конструкція повинна мати високу здатність

поглинати звук, не заважати обслуговуванню обладнання, та не порушувати естетику робочого простору. Одним із варіантів цього методу є кабіна, де розміщується найбільш шумне обладнання та працівник. Внутрішні стінки кабіни покриті звукопоглинальними матеріалами, що спрямовані на зниження рівня шуму всередині приміщення, а не просто відокремлення джерела шуму від іншої частини робочого приміщення.

Звукоізоляція, як метод зниження шуму, використовується для відокремлення джерела шуму від основного приміщення чи ізоляції найбільш шумних об'єктів. Це досягається за допомогою спеціальної звукоізоляційної стіни або перегородки. Також, шумові джерела можуть бути встановлені в окремій кабіні. В ізольованому приміщенні чи кабіні рівень шуму залишається високим, проте він впливає на меншу кількість людей. Крім того, звукоізоляція може бути досягнута шляхом використання спеціальних кабін для операторів, які керують технологічним процесом. Також, ефективність звукоізоляції може бути забезпечена за допомогою екранів та ковпаків, які захищають робоче місце та людину від прямого звукового впливу, але не зменшують рівень шуму в приміщенні.

Акустична обробка приміщення — це застосування звукопоглинальних матеріалів на стелях та верхній частині стін з метою зниження відбитих звукових хвиль. Це може включати встановлення звукопоглинальних щитів, конусів, кубів, а також використання резонаторних екранів, тобто штучних поглиначів. Ефективність такої обробки залежить від властивостей матеріалів, їх розташування, форми та розмірів приміщення, місць розташування джерел шуму. У приміщеннях з низькою стелею (до 6 м) цей метод має більший ефект. Акустична обробка може знизити рівень шуму на до 8 дБА.[12]

Запланування заходів щодо зменшення шуму виробничих об'єктів та обладнання — це крок, який слід виконувати на етапі проектування. Важливо виділити та перемістити шумне обладнання до окремих приміщень, де знаходиться мінімальна кількість працівників, щоб зменшити вплив підвищеного рівня шуму на

персонал. Такий підхід дозволяє здійснювати заходи зі зниження шуму з ефективним використанням коштів, обладнання та матеріалів.

Спрямованість робіт щодо зниження шуму передбачає вперше складання карт шуму та спектрів обладнання та приміщень, на основі яких приймаються важливі рішення стосовно способів роботи над шумом.

РОЗДІЛ 7

ОХОРОНА ПРАЦІ

На всіх етапах праці, незалежно від професійної сфери, важливо вирішувати питання щодо безпеки та охорони праці. Забезпечення безпечних умов праці в значній мірі залежить від правильної оцінки ризиків, пов'язаних з небезпечними аспектами виробництва. Адаптація організму до змін, які можуть бути однаково складними, може мати походження в різних чинниках. Це можуть бути умови на робочому місці, інтенсивне фізичне чи психічне навантаження, психоемоційний стрес або їх комбінація. Цей програміст працює у відділі інформаційних технологій, що розташований у центральному корпусі авіакомпанії.

Тому саме програміст із ІТ-відділу авіапідприємства, який розробляє програмний комплекс для моніторингу мережевого програмного забезпечення на виявлення дефектів та збоїв, а також для діагностики та ідентифікації проблем у робочому мережевому обладнанні через аналіз спектральних графіків сигналу, виступає основним суб'єктом у сфері охорони праці.

7.1.Аналіз небезпечних і шкідливих факторів, що впливають на програміста

Організація робочого простору для програміста визначається параметрами приміщення. Приміщення має розміри 8 метрів у довжину та 5 метрів у ширину, і загальна площа становить 40 квадратних метрів. Висота стелі складає 3.5 метри.

У цьому просторі розташовані 6 робочих місць з комп'ютерами. Кожне місце обладнане робочим столом площею 1.44 квадратних метри, стільцем і персональним комп'ютером, який включає в себе монітор, системний блок, клавіатуру та мишу.

Згідно з [13] необхідно, щоб площа для кожного робочого місця складала не менше 6.0 квадратних метрів, а об'єм - не менше 20.0 кубічних метрів. З урахуванням вказаних вимог, площа та об'єм цього приміщення вистачають для розташування 6 робочих місць з комп'ютерами для операторів.Чинники, які призводять до негайного погіршення здоров’я працівника, називаються небезпечними. Шкідливими чинниками виробничого середовища називаються такі чинники, які безпосередньо або побічно призводять до порушення працездатності або здоров’я працюючих.

На програміста діють наступні шкідливі та небезпечні чинники, відповідно до [14]: мікроклімат, недостатнє освітлення робочого місця, статична електрика.

Мікроклімат робочої зони програміста. Робота програміста відноситься до категорії Іа, Іб - легких робіт, тому повинні дотримуватися наступні вимоги згідно [15]:

Згідно зі стандартами [16], природне освітлення для програмістів, які виконують роботи середньої точності (IV розряд зорових робіт), має коефіцієнт природного освітлення (КПО) у розмірі 1,5%, коли використовується бокове освітлення. Це рекомендовано для забезпечення оптимальних умов роботи. Щодо штучного освітлення, мінімальний рівень освітленості (Емін) повинен становити 300-500 лк для IV розряду зорових робіт. Крім того, коефіцієнт пульсації світлового потоку (Кп) не повинен перевищувати 20%. Це важливо для забезпечення стабільності та зручності роботи програмістів під штучним освітленням. Загальна увага до природного та штучного освітлення допомагає забезпечити комфортні та безпечні умови для виконання зорових завдань програмістами.

Таблиця 5.1

Допустимі значення параметрів неіонізуючих електромагнітних випромінювань

Найменування параметра	Допустимі значення
Напруженість електричної складової електромагнітного поля на відстані 50 см від поверхні монітора ПК	10 Вт/м
Напруженість магнітної складової електромагнітного поля на відстані 50 см від поверхні монітора ПК	0.3 А/м
Напруженість для операторів ПК не повинна перевищувати	20 кВ/м

На відстані 5-10 см від екрана та корпусу монітора електричне напруження не перевищує 6 В/м, що в межах безпеки. Приміщення ІТ відділу за ступенем ризику ураження електричним струмом відноситься до першого класу, тобто є безпечним середовищем без підвищеної небезпеки: сухе, чисте, з нормальною температурою та мінімальними ризиками.

На робочому місці програміста всі деталі устаткування виготовлені з металу, крім корпусу системного блоку комп'ютера, який відповідає стандартам фірми IBM. Згідно з [5], пунктом 5 «Заходи по захисту від статичної електрики», системний блок повинен мати заземлення для виведення статичної електрики. Проте під час огляду було виявлено, що заземлення відсутнє, що не відповідає встановленим нормам. Основні фактори ризику ураження людини електричним струмом на цьому робочому місці включають:

- Дотик до металевих, не провідних частин (корпусу комп'ютера), які можуть опинитися під напругою внаслідок пошкодження ізоляції.

- Неконтрольоване використання електричних пристроїв.

- Відсутність навчання працівників щодо правил безпеки у використанні електрообладнання.

7.2 Організаційні та конструктивно-технологічні заходи для зниження впливу шкідливих виробничих факторів.

Щоб забезпечити оптимальні умови в робочій зоні ІТ відділу незалежно від зовнішніх факторів, використовуються системи регулювання температури, вологості, чистоти та обміну повітря. У холодну пору року застосовується водяне опалення, а в теплу — кондиціювання повітря [18]. Це дозволяє автоматично підтримувати оптимальні умови для роботи незалежно від кліматичних умов зовнішнього середовища.

В ході оцінки освітлення на робочому місці програміста виявлено, що воно не відповідає встановленим нормам. Для поліпшення умов праці рекомендується підвищити загальний рівень освітленості приміщення шляхом встановлення 5 нових світильників, щоб загалом було 36 світлодіодних ламп, як розраховано. Також для забезпечення постійної якості освітлення рекомендується скласти графік регулярного очищення віконних блоків та світильників не рідше, ніж 2 рази на рік [16]. Це допоможе забезпечити оптимальні умови освітлення для комфортної та продуктивної роботи.

Для забезпечення електробезпеки у приміщенні ІТ відділу пропонується використовувати наступні технічні засоби захисту:

- Для зменшення накопичення статичної електрики рекомендується використовувати зволожувачі, нейтралізатори та покриття підлоги антистатичним матеріалом.

- Забезпечити заземлення металевих корпусів устаткування та системних блоків ПК. Опір заземлення повинен бути не більше 4 Ом для електроустановок з напругою до 1000 В згідно з Правилами улаштування електроустановок (ПУЕ).

Організаційними заходами слід проводити своєчасні інструктажі з техніки безпеки [19].

Щодо ергономіки та організації робочого місця, встановлено, що воно відповідає установленим вимогам. Однак, на підставі аналізу навантаження та напруженості роботи, рекомендується скоротити час безперервної роботи за комп'ютером. Рекомендується виконувати перерви, загальна тривалість яких складає 50 хвилин при 8-годинному робочому дні [20]. Це сприятиме зменшенню напруження та покращенню здоров'я при роботі за комп'ютером.

Розрахунок освітленості робочого місця програміста ІТ відділу авіапідприємства на відповідність розряду зорової роботи

Вимірювання показало, що рівень природної освітленості на поверхні, де розташований ПК програміста, становить 200 лк, при освітленості тієї ж поверхні відкритим небосхилом - 20000 лк. Це означає, що коефіцієнт природного освітлення (КПО) складає 1%, що не відповідає нормативним показникам.

У приміщенні для штучного освітлення використовуються світлодіодні лампи T8 G13, які в порівнянні зі звичайними лампами мають декілька важливих переваг. Вони наближені за спектральним складом світла до природного, мають підвищену світлову віддачу (2-5 разів вище, ніж у ламп розжарювання) та довший термін служби (до 10 тисяч годин) [16]. Ці параметри сприяють покращенню умов освітлення в приміщенні та забезпечують більш комфортні умови для роботи.

Формула для розрахунку світлового потоку, що падає на робочу поверхню, виглядає наступним чином: $F=E*S*K*Z/n$

Де:

F - світловий потік, необхідний для робочої поверхні;

E - освітленість, яку ми хочемо отримати (в люксах);

S - площа приміщення (в квадратних метрах);

K - коефіцієнт використання світлового потоку;

Z - коефіцієнт запасу освітленості;

n - кількість світильників.

Обчислимо індекс приміщення за формулою:

i=S/h(A+B)

(де S – площа приміщення, S=40 м2; h – розрахункова висота підвісу, h = 3.3 м; A – ширина приміщення, A = 5 м; B – довжина приміщення, B = 8 м.)

Підставивши значення отримаємо: i=40/3.3(5+8)=0.93. Знаючи індекс приміщення, знаходимо n=0.22. Підставимо всі значення у формулу для визначення світлового потоку F:

F=(300*1.5*40*1.1)/0.22=90000 Лм.

Для освітлення використані світлодіодні лампи з матовим покриттям типу

LRC-T8-S1500G13-220-22,0W, світловий потік яких Fл = 2500 Лм.

N=F/Fл (6.3)

(де N – визначуване число ламп; F – світловий потік, F=90000 Лм; Fл – світловий потік однієї лампи, Fл = 2500 Лм.)

N=90000/2500=36

В приміщенні використовуються світильники типу ЛПО. Кожен світильник комплектується чотирма лампами. Тобто необхідно використовувати 9 світильників із 36 працюючими лампами в них.

У ІТ відділі авіапідприємства, де аналізувалось робоче місце програміста працює 5 світильників з 20 лампами в них, тому рівень штучного освітлення не задовольняє санітарним нормам.

7.3 Пожежна безпека

Приміщення ІТ відділу центрального офісу авіапідприємства за класифікацією вибухопожежної і пожежної небезпеки, згідно з [21], відноситься до категорії Д, яка охоплює "негорючі речовини та матеріали в холодному стані". Такі приміщення включають об'єкти, де розташовані горючі речовини в системах машин,

охолодження та гідроприводу устаткування, з обмеженою кількістю горючих матеріалів (не більше 60 кг на одиницю устаткування) та при тиску не більше 0.2 мПа, а також включають кабелі електропроводки до устаткування та окремі предмети меблів на визначених місцях.

Центральний офіс, який включає ІТ відділ, з погляду пожежної небезпеки будівельних конструкцій, класифікується як категорія К1, що відповідає "малопожежонебезпечним" приміщенням. Тут присутні займисті матеріали, такі як книги, документи, меблі, оргтехніка і т.д., а також важкогорючі речовини, наприклад, сейфи, різне устаткування і т.д., які можуть горіти без вибуху при взаємодії з вогнем.

Це будинки, що мають несучі та огороджуючі конструкції, зазвичай виконані з природних або штучних кам'яних матеріалів, бетону чи залізобетону. У таких будинках для перекриттів можуть використовувати дерев'яні конструкції, які оброблені штукатуркою або важкогорючими листовими, а також плитні матеріали.

Отже,ступінь вогнестійкості будинку Центрального офісу можна визначити як третю (III).

Згідно з [21], на кожному підприємстві, враховуючи рівень пожежної небезпеки, має бути визначений протипожежний режим, який включає організацію технічних засобів протипожежного захисту. Це може включати протипожежний водопровід, насосні станції, системи пожежної сигналізації, автоматичні системи пожежогасіння, системи димовидалення, вогнегасники та інше.

У вашому приміщенні встановлено 1 переносний вуглекислотний вогнегасник типу ВВК-5, що відповідає потребам для даного типу та площі приміщення. Крім того, на стелі встановлено 2 бездротових ІЧ датчики диму Страж М-501, які розраховані на покриття площі 40 м2. Такий захист допомагає забезпечити реагування на можливу пожежу та сприяє збереженню безпеки в приміщенні.

Засоби пожежогасіння та пожежно-охоронної сигналізації. Відповідно до [21]: «3.3. На кожному підприємстві з урахуванням його пожежної небезпеки наказом (інструкцією) повинен бути встановлений відповідний протипожежний режим, у

тому числі визначені: ... порядок організації експлуатації і обслуговування наявних технічних засобів протипожежного захисту (протипожежного водопроводу, насосних станцій, установок пожежної сигналізації, автоматичного пожежогасіння, димовидалення, вогнегасників тощо); ...».

В приміщенні встановлено 1 переносний вуглекислотний вогнегасник типу ВВК-5, який вважається достатнім для ліквідації пожежі в даному типі приміщення та його площі. Крім того, на стелі розташовано 2 бездротових ІЧ датчики диму Страж М-501, призначених для покриття площі 40 м2. Ці датчики призначені для вчасного виявлення диму, що допомагає вчасно спрацювати системам пожежної сигналізації та прийняти необхідні заходи для запобігання розповсюдженню пожежі.

У разі виникнення пожежі, важливо негайно спрацювати протипожежною сигналізацією. Потрібно відключити електроживлення, негайно набрати номер 101 для виклику пожежної команди. Відповідно до плану евакуації, зображеного на Рисунку 5.1, вивести людей з приміщення. Після цього слід приступити до ліквідації пожежі за допомогою вогнегасників.

Якщо пожежа є невеликою і має лише невелике вогнище, можна використати підручні засоби, щоб заборонити доступ повітря до місця загоряння.

Рис. 7.1 План евакуації з приміщення ІТ відділу компанії

Інструкція з охорони праці при роботі з персональним комп’ютером

Вимоги до обладнання робочого місця з ПК уточнюють, що для праці з відеотерміналами важливо забезпечити таке розташування, щоб у поле зору працюючого не потрапляли вікна, освітлювальні прилади або віддзеркалюючі поверхні. Поверхня робочого столу не має бути полірованою. Щоб запобігти відблискам на екрані відеомоніторів, особливо в сонячні дні або влітку, рекомендується розміщувати екран так, щоб світло з вікна падало збоку, переважно зліва.

Загалом, рекомендації до обладнання робочого місця з ПК вказують на важливість правильного розташування обладнання. Екран відеомонітора ПК повинен бути віддалений від очей оператора на 500-700 мм і знаходитися під кутом 10-40 градусів від лінії зору. Краще, якщо екран розташований перпендикулярно до лінії зору.

Комп'ютер слід розміщувати на відстані не менше 1 метра від джерела тепла. Клавіатура повинна бути на поверхні столу або підставці на відстані 100-300 мм від краю, що звернений до користувача, під кутом нахилу 5-15 градусів.

Робочий стіл повинен мати висоту від 680 до 800 мм. Крісло, у свою чергу, має дозволяти операторові регулювати висоту, кут нахилу спинки та висоту сидіння для забезпечення зручної робочої пози.

Щоб запобігти відблискам на екрані від сонячного світла, слід встановити сонцезахисні пристрої на вікнах. Екран відеомонітора краще розміщувати так, щоб світло падало на робоче місце з боку, особливо зліва.

Рекомендації щодо освітлення та догляду за обладнанням ПК дуже важливі. Щодо освітлення, рекомендується використовувати люмінесцентні лампи, але можна також застосовувати лампи розжарювання в світильниках місцевого освітлення. Освітленість робочого місця в горизонтальній площині на висоті 0,8 м від рівня підлоги повинна бути не менш як 400 лк, тоді як вертикальна освітленість

у площині екрану має бути не більш як 200 лк. Це допомагає зменшити напруженість зору, забезпечуючи рівномірну яскравість на робочій поверхні монітора та навколишньому просторі.

Для забезпечення комфорту та безпеки оператора також рекомендується щоденно вологе прибирання та регулярне провітрювання приміщення, особливо під час робочого дня. Чистку екрану від пилу слід проводити щонайменше один раз на день.

Щодо захисту від електромагнітних випромінювань та електростатичних полів, використання захисних екранів може допомогти захистити оператора від негативного впливу монітора.

Вимоги безпеки перед початком роботи.

Перевірте цілісність обладнання, включаючи корпуси системного блоку, монітора, принтера, клавіатури.

Огляньте кабелі живлення та їх підключення.

Підготуйте робоче місце, позбувшись речей, які можуть заважати.

Запуск ПК:

Увімкніть живлення ПК.

Якщо виникають проблеми під час запуску (наприклад, не вдається завантажити систему), повідомте про це свого керівника або відділ інформаційних технологій.

Під час роботи:

Якщо ви помічаєте пошкодження або інші проблеми, повідомте про це керівника перед тим, як продовжувати роботу.

Це базові кроки для забезпечення безпеки під час використання ПК. Важливо завжди слідкувати за станом обладнання та повідомляти про будь-які аномалії чи пошкодження.

ВИСНОВКИ

У ході виконання цієї кваліфікаційної роботи проведено глибоке дослідження інтеграції програмно-конфігурованих мереж (SDN) та потокового відео, поклавши особливий акцент на вивчення можливостей SDN у збільшенні ефективності потокового відео. Основна мета полягала у відповіді на зростаючий запит на високоякісний відеострімінг, враховуючи його швидко розвиваючу складність та вимоги до продуктивності.

У відповідних розділах нашої роботи ми ретельно розглянули основні принципи SDN, проаналізували протоколи потокового відео та вивчили ключову роль протоколу OpenFlow у динамічному управлінні мережею. Ми застосували систему потокового відео з використанням таких інструментів, як Mininet, MiniEdit, Wireshark та VLC Media Player, для моделювання та оцінки продуктивності системи в різних сценаріях.

Результати отриманих експериментів та симуляцій переконливо демонструють переваги системи потокового відео на базі принципів SDN. Інтеграція цих принципів сприяла покращенню якості обслуговування (QoS), зниженню перевантаження мережі та оптимізації розподілу ресурсів. Отримані поліпшення забезпечують вражаючий досвід користувачів у сфері потокового передавання, проявляючись у зменшенні буферизації, підвищеній якості відео та ефективному використанні пропускної здатності.

Результати цього дослідження відзначають потужний потенціал SDN як технології, що здатна перетворити сферу потокового відео. Завдяки програмованості та централізації SDN, оператори мереж та постачальники послуг можуть гнучко реагувати на зміни у мережі, оптимізувати розподіл ресурсів та забезпечувати безперебійний стрімінг для кінцевих користувачів.

Так, зростання попиту на потокове відео вимагає подальших досліджень та розвитку у галузі потокового відео на основі SDN. Майбутні дослідження можуть фокусуватися на додаткових оптимізаціях, таких як маршрутизація з урахуванням вмісту та адаптивна потокова передача бітрейту. Це дозволить ще більше поліпшити досвід користувача та адаптуватися до різноманітних умов мережі.

Шлях до майбутніх досягнень та інновацій у цій галузі пролягає через постійну інтеграцію SDN і потокового відео. Це відкриває перспективу для високоякісних та надійних потокових передач, які безперешкодно досягатимуть користувачів у всьому світі. Така перспектива створює базу для подальших зрушень у розвитку технологій передавання мультимедійного контенту та забезпечує удосконалення досвіду використання цих послуг.

Підсумовуючи, цей випускний проект відіграє значну роль у поглибленні розуміння потокового відео через використання інфраструктури SDN. Він не лише пропонує важливі відомості для науковців та мережевих інженерів, а й стає важливим джерелом знань для студентів, які мають інтерес до цієї сфери. Робота відкриває шлях для подальших досліджень і визначає нові можливості розвитку цієї важливої галузі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Naudts B, Kind M, Westphal FJ, Verbrugge S, Colle D, Pickavet M. Techno-economic analysis of software defined networking as architecture for the virtualization of a mobile network. Software Defined Networking (EWSDN), European Workshop on. IEEE, 2012, с 67–72.

2. Benson T, Akella A, Maltz D. Unraveling the complexity of network management. Proc. 6th USENIX Symp. Networked Syst. Design Implement. USENIX Association, 2009, с. 335–348.

3. Floodlight controller, Floodlight documentation, for developers, architecture. [Online]. Retrieved from: http://www.projectfloodlight.org/floodlight/.

4. OpenDaylight: a Linux Foundation Collaborative Project, 2014. [Online]. Retrieved from: http://www. opendaylight.org.

5. Erickson D. The Beacon OpenFlow controller. Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, 2013, с.13–18.

6. Casado M, Freedman M, Pettit J, Luo J, McKeown N, Shenker S. Ethane: Taking control of the enterprise. Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. ACM, 2007, с. 12.

7. McKeown N, Anderson T, Balakrishnan H, et al. OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review 2008; с.69–74.

8. Прогнозування екологічних ризиків з використанням аналізу ієрархів та теорії нечітких множин: міжнародна науково-практична конференція «І-й всеукраїнський з'їзд екологів»: Тези доповідей. Україна, м. Вінниця, 4-7 жовтня 2016. – 2016. – С.25

9. Клап Я. А., Яремкевич О. С., Червецова В. Г., Заярнюк Н. Л., Новіков В. П., Дослідження впливу електромагнітних, постійних магнітних та акустичних полів на організм людини // Вісник Нац. ун-ту "Львівська політехніка". – 2016 – № 812. – С. 365–372.

10. Сучасний стан досліджень впливу електромагнітних випромінювань на організм людини [Електронний ресурс]/[А. П. Чорний, В. В. Никифоров, Д. І. Родькін, В. Ю. Ноженко] // Інженерні та освітні технології в електротехнічних та

комп'ютерних системах: щоквартальний науково-практичний журнал. – Кременчук: КрНУ, 2013.

11. Екологія та охорона навколишнього природного середовища: навч. посібник для вузів / В. С. Джигирей. - 6-те вид., випр. і доп. - К. : Знання, 2017. - 422 с

12. Боротьба з шумом на виробництві: Довідник / Під ред. О. Я. Юдіна. – М: Машинобудування, 2015. – 297

13. ДСанПіН 3.3.2-007-98 «Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин»

14. ГН 3.3.5-8-6.6.1-2002 «Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу»

15. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень»

16. ДБН В.2.5-28-2006 «Інженерне обладнання будинків і споруд. Природне і штучне освітлення»

17. НПАОП 0.00-1.29-97 «Правила захисту від статичної електрики»

18. ДСТУ 12.1.005-88 «ССБП. Загальні санітарно-гігієнічні вимоги до повітря робочої зони»

19. ДСТУ Б В.2.5-82:2016 «Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом»

20. ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги»

21. НАПБ А.01.001-2004 «Правила пожежної безпеки в Україні»