НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Інститут телекомунікаційних систем

Кафедра Інформаційно-телекомунікаційних мереж

До захисту допущено:

В.о. завідувача кафедри

_____ Лариса ГЛОБА

«___»_____2021 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційно-комунікаційні технології»

спеціальності 172 «Телекомунікації та радіотехніка»

на тему: «Удосконалений спосіб функціонування гіпервізора NFV в мережах SDN»

Виконав (-ла):
студент (-ка) IV курсу, групи ТІ-72
Оліфіренко Роман Сергійович _____

Керівник:
Професор кафедри ІТМ, д.т.н, с.н.с
Скулиш Марія Анатоліївна _____

Рецензент:
Професор кафедри ТК, д.т.н, проф.,
Лисенко Олександр Іванович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут телекомунікаційних систем

Кафедра Інформаційно-телекомунікаційних мереж

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інформаційно-комунікаційні технології»

ЗАТВЕРДЖУЮ

В.о.завідувача кафедри

_____ Лариса ГЛОБА

«___»_____2021 р.

ЗАВДАННЯ

на дипломну роботу студенту

Оліфіренку Роману Сергійовичу

1. Тема роботи: «Удосконалений спосіб функціонування гіпервізора NFV в мережах SDN», керівник роботи Скулиш Марія Анатоліївна, професор кафедри інформаційно-телекомунікаційних мереж ІТС, професор, д.т.н., с.н.с,, затверджені наказом по університету від «14» квітня 2021 р. № 1007-с2. Термін подання студентом роботи 7 червня 2021 р.

3. Вихідні дані до роботи

4. Зміст роботи

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

6. Дата видачі завдання 1 жовтня 2020 року

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Вибір та обґрунтування напрямку дослідження	01.09.2020 – 30.09.2020	Виконано
2	Дослідження SDN та NFV, і їх зв’язок	01.10.2020 – 30.11.2020	Виконано
3	Аналіз методів віртуалізації у традиційних мережах	01.12.2020 – 28.02.2021	Виконано
4	Аналіз методів віртуалізації у програмно-конфігурованих мережах	01.03.2021 – 15.04.2021	Виконано
5	Опис та постановка задачі	01.04.2021 – 30.04.2021	Виконано
6	Підготовка середовища з метою отримання вихідних даних	1.05.2021 – 15.05.2021	Виконано
7	Аналіз отриманих результатів	16.05.2021 – 01.06.2021	Виконано
8	Підготовка документації	01.06 2021 – 06.06.2021	Виконано

Студент Роман ОЛІФІРЕНКО

Керівник Марія СКУЛИШ

РЕФЕРАТ

Актуальність: Віртуалізація комп'ютерів стає все більш популярною та поширеною в контексті хмарних обчислень. Користувачі можуть створити власні приватні віртуальні машини, розміщені у постачальника послуг, за лічені хвилини, використовуючи такі послуги, як Amazon EC2, Microsoft Azure або Google Cloud Platform. Віртуалізація та ізоляція процесора, пам'яті та елементів зберігання даних у цих середовищах є простою та високо автоматизованою. Однак динамічна конфігурація, віртуалізація та ізоляція мережевих ресурсів все ще вважається "відсутньою ланкою", яка буде з'єднувати всі інші віртуалізовані пристрої».

Сьогодні ми активно користуємося мережевими технологіями. За останні 10 років компанії дають змогу використовувати пакетні рішення для організації транспорту в телекомунікаційних мережах. Це стає можливим за рахунок керованих мереж SDN та технології віртуалізації. У цій роботі представлено та порівняно функції та можливості, які можуть надавати гіпервізори. Результати показують, що гіпервізори роблять мережі SDN більш гнучкими у процесі віртуалізації. Але, з іншого боку, можна з впевненістю сказати, що така гнучкість веде за собою зниження продуктивності

Мета роботи: Проаналізувати існуючі методи віртуалізації та створити рекомендації для покращення працездатності гіпервізора NFV за допомогою використання контролера SDN

Ключові слова: SDN, NFV, FlowVisor, VeRTIGO, OpenVirteX, гіпервізор, програмно-конфігурована мережа, віртуалізація мережевих функцій

ABSTRACT

Relevance: The virtualization of computers is becoming increasingly popular and widespread in the context of cloud computing. Users can set up their own private virtual machines hosted by a service provider in a matter of minutes, using services such as Amazon EC2, Microsoft Azure, or Google's Cloud Platform. Virtualization and isolation of the CPU, memory and storage elements in these environments is straightforward and highly automated. However, dynamic configuration, virtualization, and isolation of network resources is still considered a "missing link that will interconnect all other virtualized appliances"

Today we actively use network technologies. For the last 10 years the company can use a package of solutions for the organization of transport in telecommunication networks. This is made possible by managed SDNs and virtualization technologies. This paper presents and compares the functions and capabilities that hypervisors can provide. The results show that hypervisors make SDNs more flexible in the virtualization process. But, on the other hand, it is safe to say that such flexibility leads to reduced productivity

Purpose: Analyze existing virtualization methods and make recommendations for improving the performance of the NFV hypervisor using the SDN controller.

Keywords: SDN, NFV, FlowVisor, VeRTIGO, OpenVirteX software-configured network, virtualization of network functions, hypervisor.

ЗМІСТ

ВСТУП

Зі зростанням програмно-визначеної мережевої тенденції за останні пару років було розроблено кілька контролерів віртуалізації мережі. Ці контролери, які також називаються мережевими гіпервізорами, керують фізичними мережами, основою яких є SDN. Це зроблено для того, щоб орендарі використовувати одне і те ж апаратне забезпечення переадресації, не ризикуючи зазнати впливу чи впливу інших орендарів. Однак багато областей залишаються невивченими. Ця дипломна робота представляє та оцінює деякі функції, пропоновані мережевими гіпервізорами, такі як повна доступність простору заголовків, ізоляція та прозорі можливості переадресації трафіку для орендарів. Час і пропускна здатність потоку також вимірюються та порівнюються між різними мережевими гіпервізорами.

Оцінюються три різні гіпервізори мережі: FlowVisor, VeRTIGO та OpenVirteX. Ці інструменти віртуалізації оцінюються за допомогою експериментів, проведених на двох різних тестових стендах: емульованому сценарії Mininet та фізичному тестовому стенді з одним перемикачем. Результати показують, що гіпервізори мережі привносять гнучкість та варіативність SDN у віртуалізацію мережі, що полегшує мережевим адміністраторам з точністю визначати спосіб розподілу та розподілу мережі між орендарями. Однак така підвищена гнучкість може спричинити за собою зниження продуктивності, а також створює додаткові ризики взаємодії через відсутність стандартизації методів віртуалізації.

Задачі дослідження:

1. Розглянути методи віртуалізації у традиційних мережах.
2. Проаналізувати методи віртуалізації у програмноконфігурованих-мережах.
3. Для методів Vertigo, OVX, FlowVisor, використовуючи програмне забезпечення MiniNet, створити імітаційну модель для оцінки трьох гіпервізорів.
4. Аналіз отриманих результатів.

Об’єкт дослідження: віртуалізація мережевих функцій в програмно-конфігурованих мережах.

Предмет дослідження: функціонування гіпервізора NFV за рахунок використання засобів контролера SDN.

Наукова новизна: Запропонували удосконалений спосіб функціонування гіпервізора NFV в мережах SDN, який базується на групі методів віртуалізації та дозволяє використати переваги кожного з них.

РОЗДІЛ 1

SDN ОСНОВНІ ПОЛОЖЕННЯ. АРХІТЕКТУРА

1.1 Програмно-конфігуровані мережі

Програмно-конфігуровані мережі (ПКМ) - це нове покоління комп'ютерних мереж, в яких ключовою відмінністю є винос логіки маршрутизації за межі пристрою на окремий виділений сервер. Поява цього покоління пов'язано з ростом трафіку і падінням ефективності поточних комп'ютерних мереж.

Незважаючи на свою розповсюдженість, сучасні мережі, побудовані на основі стеку TCP/IP, не відповідають зростаючим потребам щодо швидкості введення в експлуатацію та швидкості ре конфігурації мереж. Поява програмно-конфігурованих мереж (ПКМ) покликана змінити існуючий стан речей.

Головною відмінністю архітектури ПКМ є видалення логіки керування трафіком з складу програмного забезпечення маршрутизаторів та комутаторів, поява логіки централізованого управління мережею, а також можливість конфігурування мережі.

ПКМ мережа розриває вертикальну інтеграцію, тобто розділяє управлінську логіку (рівень керування) від нижче розташованих маршрутизаторі та комутаторів, котрі здійснюють передачу трафіку (рівень даних).

З появою розподілу цих рівнів на рівні даних здійснюється лише маршрутизація трафіку, а управління зосереджується централізованим контролером (мережевою операційною системою) . що спрощує введення мережевих політик, ре конфігурування мережі та її еволюцію.

Слід зазначити, що логічно централізована програмна модель не означає фізичної централізації системи. Натомість робочі станції ПКМ спираються на фізично розподілені керуючі рівні Розподілення цих рівнів є ключовим у досягненні бажаної гнучкості мережі одночасно розділяючи проблему

керування мережею не більш прості задачі спрощуючи одночасно введення нових абстракцій.

В даний час відбувається кардинальна зміна парадигми телекомунікацій – перехід до мультисервісних мереж, які будуються на принципах комутації пакетів. Мультисервісна мережа, дозволяючи відмовитися від численних накладених вторинних мереж, створює єдину інформаційно-телекомунікаційну структуру, яка підтримує всі види трафіку (дані, голос. відео) та надаю всі види послуг у будь якій точці у будь який час у будь якому обсязі.

В ПКМ рівні управління мережею і передачі даних поділяються за допомогою винесення функцій управління (комутаторами, маршрутизаторами і т. П.) В програми, що працюють на виділеному сервері, званому контролері. Перші концепції таких мереж були сформульовані фахівцями університетів Стенфорда і Берклі ще в 2006 році, а вироблені ними дослідження отримали схвалення не тільки в наукових колах, а й у сфері бізнесу. Ці ідеї тепло зустріли такі виробники мережевого обладнання як, як Cisco, HP і інші. У березні 2011 року був заснований консорціум Open Networking Foundation (ONF). Засновниками цієї спільноти є ряд великих і впливових компаній в сфері IT. Цими компаніями були: Verizon, Yahoo, Microsoft, Google, Deutsche Telekom, і Facebook. Склад ONF швидко поповнився і іншими не менш відомими компаніями, таких як Marvell, Citrix, IBM, NEC, Brocade, Oracle, HP, Dell, Ericsson, і ряд інших. Найперша практична реалізація ПКС була запропонована компанією Nicira, яка нещодавно стало частиною VMware.

Інтерес IT-компаній до ПКМ викликаний тим, що така технологія дозволяє підвищити ефективність мережевого обладнання на 25-30%, знизити на 30% витрати на експлуатацію мереж, дозволяє перетворити управління мережами з творчого проектування в інженерію, підвищити захищеність мережі і дозволити користувачам самостійно з допомогою програм створювати нові сервіси та оперативно завантажувати, і використовувати їх на мережевому обладнанні.

У більшості напрацювань і досліджень ключові моменти в ПКМ пов'язані з програмою Global Environment for Network Innovations (GENI) дослідження майбутнього Інтернету, що включає в себе близько 40 провідних університетів США. Діяльність об'єднаного центру Стенфорда і Берклі, яка провадить різні вивчення, дослідження, експерименти і напрацювання в області Internet2; а також з Сьомої рамкової програми досліджень Європейського Союзу Ofelia і проектом FEDERICA.

Основні концепції ПКМ:

- Винесення процесу управління даних з передавального пристрою на виділений сервер, а процеси передачі даних залишити на передавальних пристроях;
- Уніфікований і незалежний інтерфейс між рівнем управління і рівнем передачі даних;
- логічно централізоване управління мережею, здійснюване за допомогою контролера з встановленої мережевою операційною системою і реалізованими поверх мережевими додатками;
- віртуалізація фізичних ресурсів мережі;

Головна проблема поточних комп'ютерних мереж полягає в тому, що приблизно 20-25% передавальний пристрій витрачає на прокладку маршруту по сегменту мережі. Концепція програмно-конфігурованих мереж полягає в тому, щоб прибрати логіку побудови маршрутів з пристрою передачі даних і залишити йому тільки передачу даних. Новий тип мереж пропонує розміщувати логіку передачі даних на спеціальних серверах, які називаються програмно-конфігуруються контролерами, на яких відбувається обчислення маршруту в рамках сегмента мережі. Дані мережі в першу чергу націлені на центри обробки даних, адже саме вони відповідають за великий медіа контент надається на різних ресурсах.

За рахунок централізованості управління сегментом мережі дана мережа пропонує ряд серйозних переваг:

- Зниження вартості обладнання, за рахунок спрощення обладнання;
- Більш детальне управління мережею;

- Дуже низька ймовірність несанкціонованого доступу до ресурсів мережі;
- Можливість розробки або доопрацювання існуючих інструментів для програмно-конфігуруються мереж для управління ресурсами мереж і потоками даних;

В рамках розвитку концепції даної технології був розроблений спеціальний протокол передачі даних OpenFlow. Цей протокол базується на концепції управління обробки потоків даних. Принцип роботи протоколу простий, якщо комутатор отримує дані для певного потоку, він дивиться в спеціальні таблиці потоків, іменовані як flow tables, в яких вказані дії що необхідно зробити у разі отримання даного потоку, якщо він є новим, то тоді комутатор запитує інформацію у контролера , надалі контролер виробляє створення нового маршруту і заносить його в таблиці потоків пристроїв, що беруть участь в процесі передачі даних даного потоку.

Головне проблемою даного підходу є величезна кількість інформації про просування пакетів, званої forwarding state explosion. Через це навантаження на контролер є величезною і величезні сегменти мережі, що включають в себе більше 1000 передавальних пристроїв, є непосильною ношею з точки зору затримок і стабільності передавальної інфраструктури.

1.2 Концепція SDN-мереж

Програмно-конфігуровані мережі (SDN, Software-Defined Networking) – це набір методів, котрі дозволяють користувачам безпосередньо програмувати, організовувати, контролювати та керувати мережевими ресурсами, що полегшує проектування, налаштування та функціонування мережевих сервісів в динамічному режимі з можливістю масштабування.

Архітектура SDN відрізняється виключною гнучкістю, можливістю роботи з різними типами комутаторів на різних рівнях протоколу.

SDN-комутатор виконує такі функції:

- Інкапсулює та перенаправляє перший пакет потоку в SDN-контролер для визначення необхідності додавання опису потоку до таблиці комутації потоків комутаторів;
- Перенаправляє вхідні пакети відповідно до таблиці комутації потоків та з врахуванням можливої пріоритетності;
- Фільтрує пакети у визначеному потоці за правилами, визначеними контролером, з метою забезпечення безпеки або вимог щодо управління трафіком.
- Контролер SDN визначає потоки, існуючі у площині даних. Кожен потік у мережі має бути спочатку дозволений контролером і лише після цього визначаються маршрути для цього потоку та додаються записи до таблиць маршрутизації потоків комутаторів за маршрутом.

Контролер SDN визначає потоки, існуючі у площині даних. Кожен потік у мережі має бути спочатку дозволений контролером і лише після цього визначаються маршрути для цього потоку та додаються записи до таблиць маршрутизації потоків комутаторів за маршрутом.

Для зв’язку між контролером і комутаторами використовується стандартний протокол і API. Найчастіше для цієї мети використовується протокол OpenFlow. Реалізація такої архітектури у існуючих мережах потребує заміни всього комунікаційного обладнання що потребує значних капітальних витрат та повної зміни політики обслуговування мережі, що є доволі проблематичним на практиці. З цих причин повномасштабне функціонування SDN-мереж реалізовано здебільшого лише на центрах обробки даних.

З точки зору практичної реалізації на теперішній час більш цікавою є концепція віртуалізації мереж під назвою NVF (Network Function Virtualization, віртуалізація функцій мережі – ВФМ). В цій моделі комутатори не абстрагуються від виконання функцій керування мережею. А лише делегують такі функції SDN-контролеру, точніше, деякому набору віртуальних мережевих сервісів.

Підхід NFV передбачає відділення функцій логічних мережевих елементів від забезпечуючої їх апаратної інфраструктури за рахунок віртуалізації таких функцій. Все це дає можливість використання уніфікованого комп'ютерного обладнання, що забезпечує використання широких обчислювальних можливостей для управління смугою пропускання мереж. При цьому ресурси уніфікованого обладнання можуть динамічно перерозподілятися між програмно реалізованими віртуальними мережевими функціями (Virtual Network Functions – NFV)

Незважаючи на більш пізню, в порівнянні з концепцією SDN появу, оператори ринку телекомунікаційних послуг розглядають можливості впровадження рішень NVF у коротко та середньостроковій перспективі. Це пов'язано з такими особливостями концепції:

- Непотрібно достроково виводити з експлуатації обладнання, що успішно використовується;
- Впровадження NVF, на думку учасників ринку, має привести до скорочення капітальних та операційних витрат, пов'язаних з окремими послугами;
- Впровадження NVF може здійснюватися обмежено та поступово за рахунок додання нових послуг що дозволяє розглядати його як економічно обґрунтоване;
- Для рішень NFV, на відміну від рішень SDN, вже існує напрацьована практика застосування, пов'язана з хмарними та іншими рішеннями, що дозволяє розглядати NFV-концепцію як більш зрілу та готову для комерційної експлуатації.

1.4 Рішення NFV

На відміну від рішень SDN, орієнтованих здебільшого на нижні рівні архітектури – передачу даних і управління, рішення, що пропонуються на ринку NVF є, насамперед, функціями NFV. Це відповідає закладеній у концепції NVF можливості обмеженого та поступового впровадження у

відповідності до потреб. В зв’язку з цим існуючи рішення можна кваліфікувати у відповідності до сфери їх застосування у тому числі:

- Платформа NFV – комплекс засобів, що забезпечують функціонування VNF, оркестрацію та управління;
- Функції VNF, віртуалізуючи пристрої мережи пакетного дротового зв’язку (vCPE, vSBS, vBRAS, прикладні та інші шлюзи);
- Функції VNF, віртуалізуючи компоненти мобільних телефонних мереж;
- Функції мереж радіо доступу (Cloud RAN, vRAN, vBS, vNodeB, veNodeB);

  Архітектура NFV включає в себе такі основні елементи
- VNF (Virtual Network Function) – віртуальна мережева функція, наприклад, DNS, DHCP, комутатор, маршрутизатор, балансувальник.
- EMS (Element Management System) – система управління та адміністрування однієї, або декількох VNF.
- NFVI (Network Function Virtualiza tion Infrastructure) – інфраструктура NFV: апаратні та програмні ресурси у фізичному або віртуальному вигляді, на котрих працюють віртуальні мережеві функції VNF, котрі можуть бути розташовані або локально або територіально розподілено
- Hardware Resourses (апаратні ресурси, тобто обчислювальні, мережеві та ресурси зберігання) – це фізична частина інфраструктури NFV – будь-який стандартний комутатор, фізичний сервер, пристрій зберігання, таке інше.
- NFV Orchestrator (оркестратор NFV) – адміністрування інфраструктури NFV програмними ресурсами, створення завершеної послуги з декількох VNF.
- VNF Manager – менеджер, який відповідає за життєвий цикл VNF: інсталяція, активація, масштабування, інсталяція, активація, оновлення і термінація. Може керувати однією або декількома VNF.
- Virtualized Infrastructure Manager менеджер віртуальної інфраструктури) – відповідає за взаємодію віртуальної мережевої функції з апаратними

та програмними ресурсами та інвентаризацію наявних ресурсів, за збір статистики та питання продуктивності мережі.

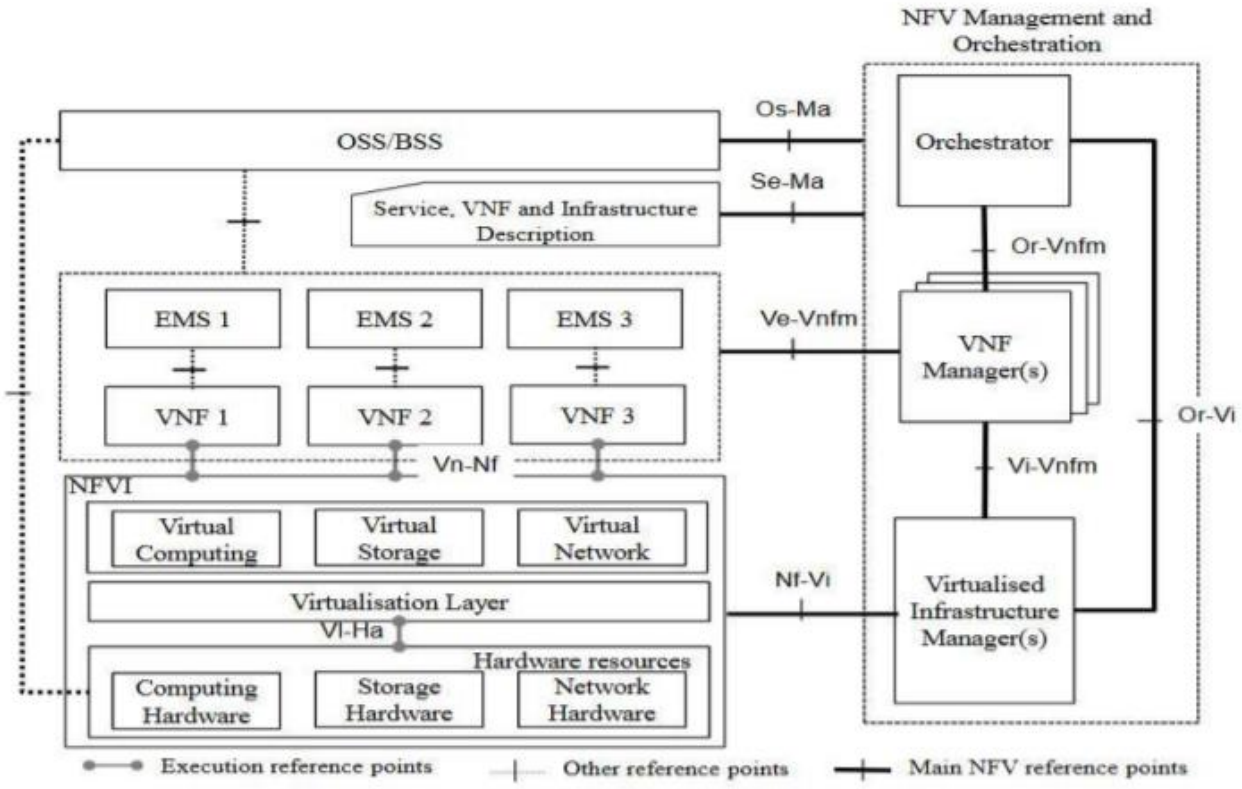

Рис.1.4 Архітектура віртуалізації мережевих функцій

1.5 Переваги та труднощі застосування NFV

Основні цілі, які можуть бути досягнуті в результаті переходу на концепцію NFV:

- прискорення інноваційних процесів в наданих сервісах за допомогою програмного розгортання і впровадження мережевих функцій та наскрізних послуг
- поліпшення експлуатаційної ефективності в результаті спільної автоматизації і скорочення операційних процедур, а також зниження енергоспоживання за рахунок міграції робочих навантажень і відключення невикористаного обладнання;
- стандартизація інтерфейсів між мережевими функціями і їх керуючими об'єктами і можливість надання мережевих елементів різними гравцями (наприклад, VNFaaS-провайдер 1 на базі IaaSпровайдера 2);

- підвищення ефективності капіталів і загальної гнучкості мережевий архітектури завдяки відходу від апаратних реалізацій.

До факторів, що стримують розвиток NFV, слід віднести:

- Стандарти та сумісність. Стандарти NFV недостатньо розроблені, що призводить до різнобою у вендорних реалізаціях та призводить до проблем з сумісністю NFV різних вендорів.
- Недостатня зрозумілість мети з точки зору бізнесу. Перші комерційні реалізації NFV з'явилися лише в 2015 р. Цінність для бізнесу від реалізації NFV поки лежить в теоретичній площині і реальних підтверджуючих кейсів все ще недостатньо.
- Проблеми міграції. Міграція традиційної мережевої інфраструктури до архітектури NFV є складним і багатоступінчастим завданням. В даний час оператори і вендори обладнання поки не мають систематичного досвіду подібних переходів і переконливих «історій успіху».
- Проблеми організаційної структури. На даний момент в структурі практично будь-якого оператора зв'язку департамент інформаційний технологій (IT) і технічний департамент мережі зв'язку оператора (CT) організаційно розділені. Тим часом, NFV і SDN відносяться саме до сфери IT. Тому потрібна не лише трансформація базової мережі оператора, а й його організаційної структури. А це доволі нетривіальне завдання. Однак, після його вирішення, як внутрішня корпоративна мережа оператора, так і його базова мережа, матимуть єдину інфраструктуру, що призведе як зниження накладних витрат, так і до підвищення ефективності бізнесу.
- Моніторинг продуктивності і якісних параметрів мережі. Перші ж реалізації показали, що існуючі системи моніторингу не адаптовані під завдання NFV. Якщо ми ведемо мову про часткову заміну обладнання на стандартні сервера, то і апаратні рішення для моніторингу каналів зв'язку повинні бути теж замінені на програмне рішення для установки на стандартний сервер. Але такі рішення вже з'являються, наприклад, TruSpeed NFV від Viavi Solutions.

Таким чином, на поточному етапі технічні фахівці мають сконцентрувати свої зусилля, по-перше, на стандартизації, подруге, на побудові більш розгалужених моделей з метою розробки ефективних засобів моніторингу та керування мережею.

1.6 Протокол OpenFlow

Протокол OpenFlow - це найбільш часто використовуваний протокол для південного інтерфейсу SDN, який відокремлює площину даних від площини управління. Довідковий документ про OpenFlow вказує на переваги гнучко-налаштованої площини пересилання. Спочатку OpenFlow був запропонований Стенфордським університетом, і зараз він стандартизований ONF. Далі ми оглянемо структуру OpenFlow, а потім опишемо функції, що підтримуються різними специфікаціями.

Архітектура OpenFlow складається з трьох основних концепцій:

(1) Мережа побудована за допомогою комутаторів, сумісних з OpenFlow, які складають площину даних;

(2) площина управління складається з одного або декількох контролерів OpenFlow;

(3) захищений канал управління з'єднує перемикачі з площиною управління. Далі ми обговоримо комутатори та контролери OpenFlow та взаємодію між ними.

Перемикач, сумісний з OpenFlow, є основним пристроєм переадресації, який пересилає пакети відповідно до своєї таблиці потоків. Ця таблиця містить набір записів таблиці потоків, кожен з яких складається з полів відповідності, лічильників та інструкцій. Записи таблиці потоків також називаються правилами потоку або записами потоку. “Поля заголовка” у записі таблиці потоків описують, до яких пакетів застосовується цей запис. Вони складаються із збігу, що підтримує підстановочні символи, над вказаними

полями заголовків пакетів. Щоб забезпечити швидку переадресацію пакетів за допомогою OpenFlow, комутатору потрібна адресна пам'ять з трикомпонентним вмістом (TCAM), що дозволяє швидко шукати збіги з підстановкою. Поля заголовка можуть збігатися з різними протоколами залежно від специфікації OpenFlow, наприклад, Ethernet, IPv4, IPv6 або MPLS. «Лічильники» призначені для збору статистики про потоки. Вони зберігають кількість прийнятих пакетів і байтів, а також тривалість потоку. "Дії" визначають, як обробляються пакети цього потоку. Поширені дії: "вперед", "скинути", "змінити поле".

Програмне забезпечення, відповідає за заповнення та управління таблицями потоків комутаторів. Вставляючи, модифікуючи та видаляючи записи потоку, контролер може змінити поведінку перемикачів щодо переадресації. Специфікація OpenFlow визначає протокол, який дозволяє контролеру вказувати комутаторам. З цією метою контролер використовує захищений канал управління.

У протоколі OpenFlow існують три класи зв'язку: контролер до комутатора, асинхронний та симетричний зв'язок. Зв'язок контролер-комутатор відповідає за виявлення особливостей, конфігурацію, програмування комутатора та пошук інформації. Асинхронний зв'язок ініціюється перемикачем, сумісним з OpenFlow, без будь-якого звернення контролера. Він використовується для інформування контролера про надходження пакетів, зміни стану на комутаторі та помилки. Нарешті, симетричні повідомлення надсилаються без запиту з будь-якої сторони, тобто комутатор або контролер можуть вільно ініціювати зв'язок без запиту з іншої сторони. Прикладами симетричного спілкування є привіт або ехо-повідомлення, які можна використовувати для визначення того, чи є канал управління все ще активним та доступним.

Коли комутатор отримує пакет, він аналізує заголовок пакета, який відповідає таблиці потоків. Якщо запис таблиці потоків знайдено там, де підстановка поля заголовка відповідає заголовку, запис розглядається. Якщо знайдено кілька таких записів, пакети узгоджуються на основі пріоритетності,

тобто вибирається найбільш конкретний запис або шаблон, що має найвищий пріоритет. Потім комутатор оновлює лічильники цього запису таблиці потоків. Нарешті, комутатор виконує дії, зазначені записом таблиці потоків на пакеті, наприклад, комутатор пересилає пакет на порт. В іншому випадку, якщо жоден запис таблиці потоків не відповідає заголовку пакета, комутатор зазвичай повідомляє своєму контролеру про пакет, який буферизується, коли комутатор може буферизувати. З цією метою він інкапсулює або буферизований пакет, або перші байти буферного пакета, використовуючи повідомлення PACKET-IN, і відправляє його контролеру; загальноприйнятим є інкапсулювання заголовка пакета і за замовчуванням кількість байт становить 128. Контролер, який отримує повідомлення PACKET-IN, визначає правильну дію з пакетом і встановлює один або кілька відповідних записів у запитуючому комутаторі. Потім буферизовані пакети пересилаються згідно з правилами; це запускається шляхом встановлення ідентифікатора буфера в повідомленні про вставку потоку або в явних повідомленнях PACKET-OUT. Найчастіше контролер встановлює весь шлях для пакету в мережі, змінюючи таблиці потоків усіх комутаторів на шляху.

Висновки

В данному розділі була розглянута реалізація архітектури SDN-NVF. Також було виявлено, що вона забезпечує наступні переваги: зменшення капітальних витрат і операційних витрат (CAPEX / OPEX) за рахунок зниження вартості обладнання і зниження споживання енергії; скорочення часу «виходу на ринок» для розгортання нових мережевих сервісів, поліпшення віддачі від інвестицій в нові послуги; велика гнучкість для збільшення, зменшення або розвитку послуг; відкритість для ринку віртуальних пристроїв і «чистих» учасників; можливість апробування та впровадження нових інноваційних послуг з меншим ризиком; скорочення різноманітності парку апаратних пристроїв.

РОЗДІЛ 2

МЕТОДИ ВІРТУАЛІЗАЦІЇ

2.1 Традиційні методи

2.1.1 Multiprotocol Label Switching

MPLS - це метод ефективної передачі пакетів даних згідно з Інтернет-протоколів. Тобто використання MPLS буде різними IP-пакетами по широкомасштабних переданих методах. Після концепції MPLS всі потрапляють до запрограмованого IP-пакету. З цією допомогою IP-пакети можуть бути надіслані для упорядкування вузлів без необхідності оцінки непередбачуваного заголовка IP. Згодом MPLS можна розшифрувати як свого роду IP-обмін. Широкий регіон організовує, в якому набори IP відповідно до правила MPLS називаються системами MPLS. Основна ідея MPLS була узагальнена дотепер, що MPLS думав про передачу IP-пакетів як в оптичних рамках, що залежать від оптичного просування мультиплексного поділу довжини хвилі, аналогічно тому, як в рамках синхронної цифрової ієрархії. Ця підсумована передача MPLS називається GMPLS (Узагальнює MPLS).

Багато протокольна архітектура комутації (MPLS) з допомогою ознак міток представляє концепт розділення наборів пакетів у відправленні еквівалентність класів (Прямі виправлення помилок) і проектування кожного на специфічний набір стрибків в мережі. Прямі Виправлення помилок потім наносити на карту в етикетках, або етикетка перемкнула шляхи(LSPs) через мережу. Етикетки можуть потім бути легко використані маршрутизаторами комутації з допомогою ознак міток, щоб прийняти пересилаючи рішення, тому що мережевий рівень аналізується усього лише раз - в мережевому вхідному маршрутизаторі.

2.1.2 MAC-in-MAC

Функціональність магістральних мостів постачальники мережевого обладнання зазвичай називають "MAC-in-MAC" і розширює концепцію Q-in-Q, дозволяючи повну інкапсуляцію трафіку клієнта, включаючи MAC-адресу клієнта (C-MAC). Стандарт IEEE 802.1ah-2008 представив концепцію PBB, яка згодом була включена в стандарт IEEE Std 802.1Q-2014

Рис. 2.1 Кадр MAC-in-MAC.

На рисунку 2.1 показано кадр MAC-in-MAC, в якому є адреса магістралі та адреса джерела . Також можна побачити, що в кадрі MAC-адреси клієнта не можна дізнатись за допомогою перемикачів на магістралі постачальника. Це стає зрозумілим при порівнянні кадру з одинарним та подвійним позначеним кадрами. Магістральна VLAN (B-VLAN) являє собою VLAN у магістралі, яка не залежить від інших тегів VLAN у фреймі клієнта. Тег екземпляра серверної служби (I-TAG) містить 24-розрядне поле, що називається ідентифікатором екземпляра серверної служби (I-SID), яке використовується для ідентифікації унікального клієнта в мостовій мережі магістральної мережі постачальника.

MAC-in-MAC вирішує такі проблеми:

- - Забезпечує більше ідентифікаторів для різних мережевих клієнтів
- Запобігає комутаторам на магістралі вивчати всі клієнтські MAC-адреси, вивчаючи лише адреси магістралі та вихідні адреси;
- Введення чіткої точки розмежування між мережами клієнтів та провайдерів.

2.1.3 VLAN

Віртуальні локальні мережі (VLAN) дозволяють розділити одну розширену локальну мережу на кілька, здавалося б, окремих локальних мереж. Кожній віртуальній локальній мережі присвоюється ідентифікатор (який іноді називають кольором), і пакети можуть переміщатися з одного сегмента в інший, лише якщо обидва сегменти мають однаковий ідентифікатор. Це є наслідком обмеження кількості сегментів у розширеній локальній мережі, які прийматимуть будь-який даний широкомовний пакет. Привабливою особливістю VLAN є те, що можна змінити логічну топологію, не рухаючи жодних проводів і не змінюючи жодної адреси.

Поля в кадрі VLAN є:

- Етертип (2 байти);
- Джерело MAC (6 байт);
- Перевірка CRC (4 байти);
- Призначення MAC (6 байтів);
- Корисне навантаження (змінної довжини)

Причини застосування мереж VLAN:

- Скорочення додаткових витрат процесорів всіх пристроїв за рахунок скорочення кількості пристроїв, які отримують кожен широкомовний фрейм;
- Поліпшення захисту за рахунок скорочення кількості хостів, які отримують копії фреймів при їх лавинній розсилці комутатором (широкомовлення, групова передача і одно адресатні фрейми з невідомим одержувачем).
- Поліпшення захисту хостів, які пересилають важливі дані, за рахунок їх переміщення в окрему мережу VLAN.
- Можливість більш гнучкого об'єднання користувачів в групи (наприклад, по відділах) замість фізичного поділу за місцем розташування.

- Спрощення пошуку проблеми в мережі, оскільки більшість проблем локалізується в області набору пристроїв, які формують широкомовний домен.
- Скорочення додаткових витрат на роботу протоколу розподіленого сполучного дерева (STP) за рахунок обмеження VLAN одним комутатором доступу.

2.1.4 VPN

Віртуальні приватні мережі (VPN) досягають віртуалізації таким чином, що мережевий клієнт дивиться зазвичай тільки частину цілої мережі, якою може бути Інтернет або будь-яка інша мережева інфраструктура. Тільки користувачі, беручи участь можуть послати і отримати трафік в межах VPN.

Хоча VPN часто вважають віртуалізацією мережі з доданими механізмами захисту, такими як шифрування та автентифікація, немає точного визначення того, що VPN повинна мати. У деяких випадках шифрування та автентифікація взагалі не використовуються.

У надійних VPN клієнт вірить, що мережа постачальника безпечна і недоступна для громадськості. Автентифікація чи шифрування не потрібні. Прикладами надійних VPN є VPN рівня 2 на основі MPLS, такі як віртуальна служба псевдо дроту та служба віртуальної приватної локальної мережі. У захищених VPN клієнтські дані повинні бути автентифіковані та зашифровані через мережу постачальника.

VPN, орієнтовані на підключення, використовують віртуальні схеми або тунелі для передачі даних [8], [9]. Наприклад, VPN рівня 2 MPLS або загальна інкапсуляція маршрутизації (GRE) є орієнтованими на з'єднання VPN [10]. VPN без підключення покладаються на розподіл даних клієнта на межі провайдера [8], [9]. Використання тегів VLAN для досягнення зв’язку рівня 2 між двома клієнтськими сайтами можна вважати беззв’язковим, оскільки воно в основному покладається на чистий Ethernet для переадресації [11].

Накладання VPN означає, що клієнт не знає топології мережі, що використовується для VPN, оскільки він не обмінюється інформацією про

маршрутизацію з постачальником. Накладні VPN включають тунелі VPN рівня 2 MPLS, GRE або IPSec [8], [9]. Рівні VPN вимагають від клієнта обміну інформацією про маршрутизацію з постачальником, наприклад у віртуальній VPN-маршрутизації та переадресації (VRF) [9].

Раніше організації або підприємства фізично встановлювали лінії на великі відстані, щоб забезпечити безпечну передачу даних. Однак ця система непрактична для кожного підприємства та повсякденних користувачів через вартість, простір та час, необхідні для таких установок. В останні роки, з експоненціальним зростанням Інтернету, ландшафт телекомунікацій докорінно змінився, і Інтернет став частиною майже всіх аспектів розвиненого світу, включаючи освіту, банківську справу, бізнес та політику. За останні два десятиліття громадський Інтернет виявився вразливим для зловмисників, які шукають конфіденційну інформацію. Найновішим рішенням цієї проблеми стала віртуальна приватна мережа на основі IP (IPVPN). Віртуальну приватну мережу (VPN) можна визначити як спосіб забезпечення безпечного спілкування між членами групи за допомогою використання загальнодоступної телекомунікаційної інфраструктури, збереження конфіденційності за допомогою протоколу тунелювання та процедур безпеки. Системи VPN надають користувачам ілюзію повністю приватної мережі. Віртуальну приватну мережу IP (IPVPN) можна визначити як реалізацію VPN, яка використовує загальнодоступні або спільні ресурси мережі IP для емуляції характеристик приватної мережі на основі IP.

Основною метою VPN є надання підприємствам тих самих можливостей, або навіть кращих, як у приватних мережах, але з набагато меншими витратами. Підприємства отримують вигоду від VPN у зниженні вартості, збільшенні масштабованості та збільшенні продуктивності праці, не порушуючи при цьому безпеку.

VPN повинна забезпечувати автентифікацію, контроль доступу, конфіденційність та цілісність даних для забезпечення безпеки даних

VPN, як правило, повинен підтримувати архітектуру, яка складається з основної локальної мережі в штаб-квартирі підприємства, інших локальних мереж у віддалених офісах, локальних мереж партнерів або компаній-споживачів, а також окремих користувачів, які підключаються з поля. В основному існує два типи VPN, VPN з віддаленим доступом та VPN від сайту до сайту. VPN від сайту до сайту можна поділити на внутрішню мережу VPN та екстранет VPN.

2.2 Віртуалізація в мережах SDN

2.2.1 FlowVisor

FlowVisor - система віртуалізації контролерів OpenFlow. FlowVisor є з точки зору комутатора - контролером, з точки зору контролера - одним або декількома комутаторами. Кожен контролер бачить тільки доступні йому інтерфейси комутаторів, кожен комутатор бачить тільки один загальний контролер. FlowVisor виступає системою двосторонньої віртуалізації OpenFlow протоколу для розмежування доступу декількох власників мережі до мережевої інфраструктури, як показано на рисунку 2.2

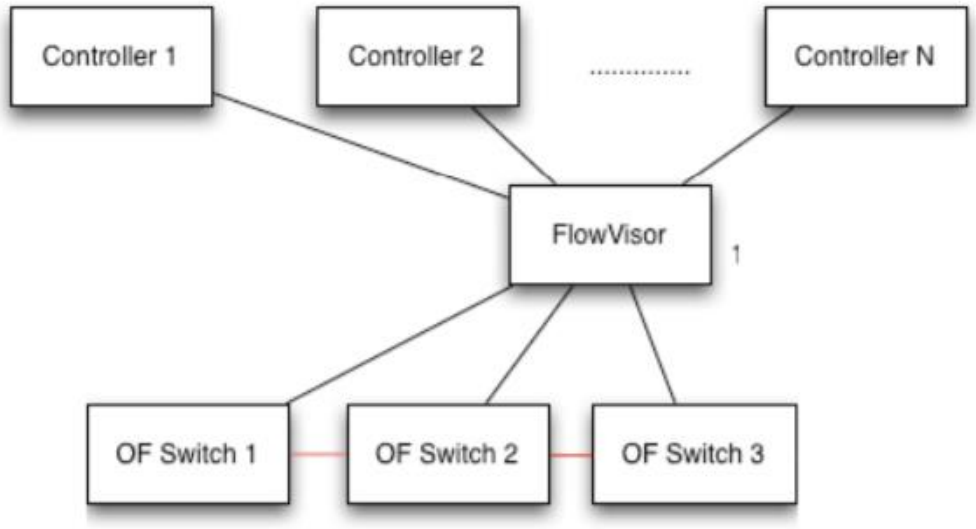

Рис. 2.2 – Схема роботи FlowVisor

Кожен набір інтерфейсів і відповідний контролер в термінології FlowVisor називається slice і управляється окремо. Також замість інтерфейсу

можна додавати в таблицю відповідності будь-які параметри таблиці OpenFlow, наприклад IP або MAC адресу. Недоліком такого підходу є відсутність механізмів спільної роботи декількох контролерів з одним ресурсом (розмежування спільного доступу).

Незважаючи на те, що FlowVisor все ще не в змозі виділити п'ять вимірів мережі, концепція SDN дозволяє йому розділяти та контролювати мережу способами, які раніше були неможливі. Експерименти FlowVisor показують, що він використовує концепції SDN для запровадження централізованого забезпечення політики, тобто FlowVisor має глобальну точку зору на мережу і може скидати або переписувати повідомлення OpenFlow відповідно до налаштованих політик [13]. Крім того, FlowVisor має можливість нарізати простір потоку з більшою гнучкістю, ніж традиційні грубі зернисті традиційні методи віртуалізації мережі. Наприклад, VLAN дозволяють нарізати трафік лише «за вхідним портом або явним тегом», тоді як механізм нарізки FlowVisor може розділити мережу за портами, протоколом, діапазонами адрес тощо [13].

2.2.2 VeRTIGO

VeRTIGO - визначена мережева платформа, призначена для віртуалізації мережі. Заснована на оригінальній системі нарізки мереж OpenVlow FlowVisor, платформа VeRTIGO спрямована на охоплення всіх аспектів віртуалізації мережі: зокрема, вона здатна виставити простий абстрактний вузол в одній крайності та доставити логічно повністю підключену мережу на протилежній кінець. Експериментальні результати показують, що VeRTIGO може надавати гнучкі та надійні послуги віртуалізації мережі для широкого кола випадків використання, незважаючи на збій та / або перевантаження в базовій фізичній мережі.

VeRTIGO [14] розширює FlowVisor концепцією абстрактних вузлів або абстрактних мережевих пристроїв, які складаються з двох основних будівельних блоків: віртуальних посилань та віртуальних портів. Віртуальні посилання об’єднують кілька фізичних вузлів і посилань і абстрагують їх до

контролера оренди як єдине посилання між будь-якими двома портами. Один або кілька віртуальних портів відображаються у фізичний порт відповідно до кількості віртуальних посилань, які повинні використовувати той самий фізичний порт. На рисунку 2.3 показано, як VeRTIGO використовує віртуальні посилання 1 і 2 між SW-A і SW-D для створення надлишкових з'єднань між фізичними портами A і B. Чотири фізичні комутатори представлені у вигляді єдиного абстрактного вузла для контролера оренди з віртуальними портами X та Y зіставляються з фізичними портами A та B відповідно.

VeRTIGO побудований поверх FlowVisor, тому він автоматично надає всі функції нарізки мережі, пропоновані FlowVisor. Тут коротко описані нові модулі, розроблені спеціально для VeRTIGO.

Модуль класифікатора визначає, які повідомлення, що надходять з мережі, обробляються контролерами OpenFlow клієнта. Деякі повідомлення обробляються безпосередньо внутрішнім контролером VeRTIGO, щоб він міг приховати деталі мережі від контролерів орендарів, які контролюють лише частину, яка їм піддається. Наприклад, на рисунку 2.3 повідомлення OpenFlow, генеровані трафіком, що надходить у SW-A з віртуального порту X, обробляються контролером орендаря, тоді як будь-які повідомлення OpenFlow, пов'язані з трафіком між SW-B та SW-D - внутрішня частина віртуальної мережі , приховані VeRTIGO - обробляються та маршрутизуються внутрішнім контролером [14].

Модуль віртуалізатора вузлів відповідає за групування кількох фізичних вузлів у мережі та змушує їх здаватися лише одним абстрактним вузлом для контролера орендаря, як зображено на рисунку 2.3. VeRTIGO мультиплексує повідомлення від декількох фізичних вузлів у мережі та відображає реальні ідентифікатори шляху даних (DPID) у віртуальні DPID [14] абстрактного вузла.

Коли один фізичний порт використовується багатьма різними віртуальними посиланнями, картограф портів обробляє відображення віртуального до фізичних портів. Це також відбувається для портів

абстрактних вузлів, які повинні мати фізичні номери портів, переназначені до номерів віртуальних портів [14].

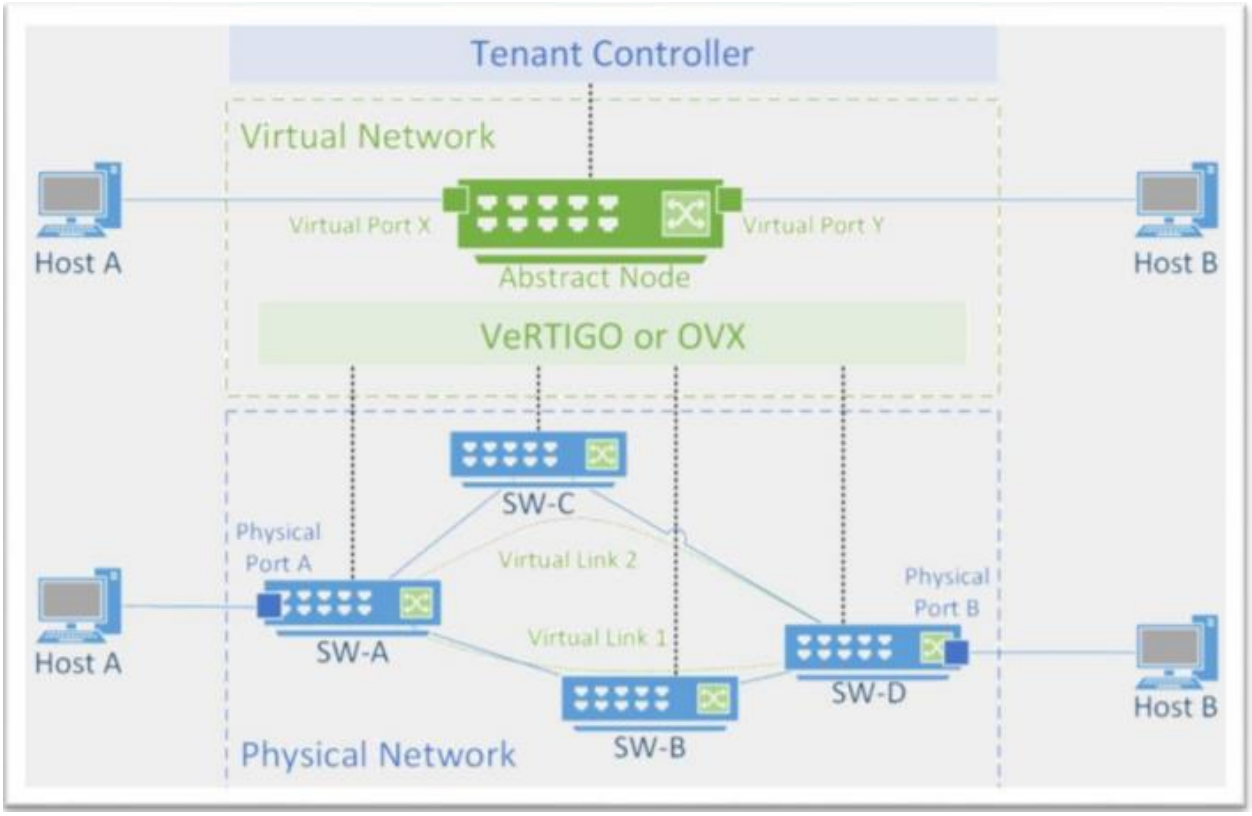

Рис. 2.3 Фізична мережа внизу та відповідна віртуальна мережа.

2.2.3 FlowN

FlowN забезпечує кожному орендаря ілюзію власного адресного простору, топології та контролера. Платформа контролера FlowN використовує технологію баз даних для ефективного зберігання та керування відображеннями між віртуальними мережами та фізичними комутаторами. Замість того, щоб запускати окремий контролер для кожного орендаря, FlowN виконує легку віртуалізацію на основі контейнерів. Експерименти з нашим прототипом FlowN, побудованим як розширення контролера NOX OpenFlow, показують, що наше рішення масштабується до великої кількості орендарів.

Кожна абстракція найбільш підходить для іншого класу орендарів. У міру того, як більше компаній переходить «до хмари», провайдери повинні виходити за рамки простого спільного використання пропускної здатності мережі, щоб підтримувати ширший спектр абстракцій. Завдяки гнучкому рівню віртуалізації мережі хмарний провайдер може підтримувати декілька

абстракцій, починаючи від простої абстракції “одного великого перемикача” (де орендарям не потрібно нічого налаштовувати) і закінчуючи довільними топологіями (де орендарі використовують власну логіку управління). Ключ до підтримки різноманітних абстракцій - це гнучкий рівень віртуалізації, який підтримує довільні топології, ізоляцію адрес і ресурсів та спеціальну логіку управління. Система FlowN забезпечує цей рівень віртуалізації.

FlowN - ефективне та масштабоване рішення віртуалізації, яке будується на основі технології SDN для програмованого управління мережею комутаторів. За допомогою FlowN кожен орендар може вказати власний адресний простір, топологію та логіку управління. Архітектура FlowN використовує досягнення технології баз даних для масштабованого відображення між віртуальною та фізичною мережами. Подібним чином FlowN використовує спільну платформу контролера, аналогічну віртуалізації на основі контейнерів, для ефективної роботи додатків контролерів орендарів. Експерименти з нашим прототипом системи FlowN, побудованою як розширення контролера NOF OpenFlow, показують, що ці два рішення щодо дизайну призводять до швидкого, гнучкого та масштабованого рішення для віртуалізації мережі.

2.2.4 OpenVirteX

OpenVirteX - мережевий гіпервізор, який дозволяє операторам надати форму віртуалізації мережі своїм клієнтам. Ми використовуємо зростаючу увагу серед програмно-визначених мереж (SDN) серед постачальників інфраструктури, спрямованих на спрощення та додання гнучкості процесу надання мереж. Зокрема, OpenVirteX базується на OpenFlow як середовищі SDN, а також для проектування. Як і FlowVisor, OpenVirteX функціонує як проксі-сервер в каналі управління, представляючи мережі OpenFlow орендарям, одночасно керуючи базовою фізичною інфраструктурою за допомогою інтерфейсу. Розкриваючи мережі OpenFlow, OpenVirteX дозволяє орендарям використовувати власну мережеву ОС (NOS) для управління мережевими ресурсами, що відповідають їх віртуальній мережі. Іншими

словами, OpenVirteX створює декілька мереж, визначених віртуальним програмним забезпеченням, з однієї. На відміну від FlowVisor, який просто розподіляє весь простір потоків серед орендарів, OpenVirteX надає кожному орендаря повністю віртуалізовану мережу, що містить топологію, визначену орендодавцем, і повний простір заголовка.

OpenVirteX підтримує внутрішнє представлення фізичної та віртуальної мереж як сукупність програмних комутаторів, посилань та кінцевих хостів. Представлення фізичної топології є основою, на якій відображаються віртуальні мережі. Орендарі можуть запитувати топологію, надаючи відображення між елементами у фізичній топології та бажаною ними віртуальною мережею. Віртуальні топології можуть варіюватися від точної копії або підмножини фізичної мережі, через гігантський віртуальний комутатор, який абстрагує всі фізичні комутатори та посилання, до будь-якої спеціальної топології, як описано орендарем. Можливість контролю абстракції топології дозволяє орендарям спрощувати свої NOS або операторам впроваджувати політики в мережі. Наприклад, орендар може відмовитись від роздільної здатності циклу в своєму NOS, запитуючи безциклову топологію, або оператор може вимагати проходження трафіку через вибрані вузли, змінюючи шлях з'єднання віртуального зв'язку або між портами гігантського комутатора.

OpenVirteX надає орендарям можливість вибору призначення адреси для своїх кінцевих хостів, дозволяючи безлічі потенційно перекриваючих блоків IP-адрес існувати в одній фізичній мережі. Щоб розрізнити хости, OpenVirteX генерує глобально унікальні ідентифікатори орендаря для кожного орендаря та для кожного хоста - фізичні IP-адреси, які кодують членство хоста за допомогою ідентифікатора орендаря. Зіткнень адрес можна уникнути, встановивши правила потоку для перезапису адрес на крайових комутаторах мережі, від адреси, призначеної орендодавцем, до фізичної IP-адреси на краю входу та навпаки на краю виходу. Процес перекладу проілюстрований на малюнку 2. Важливо, що ця процедура невидима для NOS або хостів орендаря, маючи на увазі, що NOS, що використовується для управління віртуальною

мережею, створеною OpenVirteX, не потрібно будь-яким чином змінювати для належної роботи. На додаток до запобігання прозорим псевдонімам адрес, створене процедурою відображення використовує OpenVirteX для демультиплексування повідомлень керування на північ, щоб вони потрапляли до правильних мереж орендаря.

Кожна віртуальна мережа може запускати власний NOS для програмування віртуальних комутаторів. OpenVirteX робить це можливим, відображаючи різні функції управління віртуальною мережею у відповідні фізичні мережеві ресурси. Переклади можуть бути складними, оскільки операція для одного віртуального комутатора може відображатися на декількох комутаторах і посиланнях у фізичній мережі. OpenVirteX використовує свою позицію проксі-сервера, що дозволяє йому перехоплювати та маніпулювати пакетами управління до того, як вони досягнуть призначення, яке OpenVirteX також визначає на основі свого віртуально-фізичного відображення.

2.2.5 AutoSlice

AutoSlice - рівень віртуалізації, який автоматизує розгортання та експлуатацію програмно визначених мережевих (SDN) фрагментів поверх спільних мережевих інфраструктур. AutoSlice дозволяє постачальникам субстратів перепродавати свої SDN кільком орендарям, мінімізуючи втручання оператора. Одночасно орендарям надаються засоби оренди програмованих мережевих фрагментів, що дозволяють розгортати довільні служби на основі принципів SDN. Ми окреслюємо архітектуру площини управління AutoSlice та обговорюємо найскладніші аспекти дизайну площини пересилання з акцентом на масштабованість.

Віртуалізація мережі включає життєздатне рішення для одночасного розгортання та експлуатації ізольованих мережевих фрагментів поверх спільних мережевих інфраструктур [1]. Парадигма SDN, що формується, полегшує розгортання мережевих служб, поєднуючи програмоване комутаційне обладнання, таке як OpenFlow, централізоване управління та

видимість у мережі. Ці помітні властивості SDN можуть дозволити орендарям мережі взяти під контроль свої фрагменти, реалізуючи власні рішення щодо переадресації, політики безпеки та налаштовуючи контроль доступу за необхідності.

Фундаментальним будівельним блоком для віртуалізації SDN є FlowVisor, який дозволяє нарізати таблицю потоків у комутаторах OpenFlow, розділяючи її на так звані простори потоків. Як результат, перемикачами можна одночасно керувати за допомогою декількох контролерів. Тим не менше, створення цілої топології vSDN є нетривіальним, оскільки передбачає численні операції, такі як відображення топологій віртуальних SDN (vSDN), встановлення допоміжних записів потоку для тунелювання та забезпечення ізоляції таблиці потоків. Оскільки ці операції вимагають значних ресурсів для планування та управління, ми прагнемо розробити прозорий рівень віртуалізації або гіпервізор SDN, який автоматизує розгортання та роботу довільних топологій vSDN з мінімальним втручанням оператора субстрату. На відміну від попередніх зусиль з віртуалізації SDN, ми зосереджуємось на аспектах масштабованості дизайну гіпервізора. Крім того, AutoSlice оптимізує використання ресурсів та пом'якшує обмеження таблиці потоків шляхом моніторингу статистики трафіку на рівні потоку. У подальшій частині статті ми обговоримо архітектуру площини управління та пересилання AutoSlice.

Висновки

В данному розділі було розглянуто ключові поняття, необхідні для розуміння типової архітектури та роботи SDN, таких як розділення площин управління та даних, а також те, як контролери контролюють та виявляють мережеві пристрої. Коротко пояснили VLAN, MPLS та VXLAN та VPN, і було показано, що деякі гіпервізори мережі SDN все ще можуть використовувати деякі з цих традиційних технологій для реалізації віртуалізації мережі.

РОЗДІЛ 3

РЕЗУЛЬТАТИ ТА АНАЛІЗ ОТРИМАНИХ ДАНИХ

3.1 Топологія Mininet

Експерименти проводяться на, віртуальному тестовому стенді на основі Mininet для оцінки функціональних аспектів віртуалізації мережі та фізичному тестовому стенді з одним комутатором для аналізу основних аспектів продуктивності віртуалізації мережі.

У всіх експериментах з контролером OpenFlow, що використовується, є Floodlight [19], оскільки це простий у налаштуванні контролер і містить усі функції, необхідні для експерименту з тестованими мережами, а саме виявлення топології та навчання MAC.

Mininet - це мережевий емулятор, він працює, керуючи екземпляром мереж віртуальних хостів, комутаторів та посилань. Віртуальні хости створюються як окремі просторові простори імен у Linux, а віртуальні комутатори, як правило, є Open vSwitches (OVS) [20]. Mininet дозволяє користувачеві використовувати зовнішній контролер OpenFlow для управління перемикачами OVS з підтримкою OpenFlow [21].

Топологія на основі Mininet, зображена на рисунку 3.1, використовується для всіх функціональних експериментів. До кожного комутатора до нього підключено 3 хоста, хоча хости, підключені до комутаторів SW2-SW4, не відображаються для ясності. Крім того, DPID і MAC-адреси були скорочені за допомогою подвійних двокрапок для вираження послідовності нулів. Наприклад, DPID SW3 (00: 00: 00: 00: 00: 00: 00: 03) представлений 00 :: 03, а MAC-адреса хоста C1 (02: 00: 00: 00: 03: 01) відображається як 02 :: 03: 01. Схема адресації віртуальних пристроїв, що використовуються в цій мережі, була розроблена для полегшення налаштування, розуміння та запуску експериментів:

- • Хости були призначені для трьох різних мереж, A, B та C. Мережі A та B (10.0.0.0/8) перекриваються з метою перевірки переваг ізоляції мережі.

Мережа C (3.0.0.0/8) призначена для перевірки можливих конфліктів із методом переписування заголовка, який характерний для OVX;

- Ідентифікатори комутаторів DPID відповідають номерам комутатора 00 :: NN. Наприклад, комутатор SW3 (NN = 03) має DPID 00 :: 03, щоб полегшити пошук та аналіз результатів;
- • MAC-адреси хосту були налаштовані так, щоб відображати мережу, якій вони належать, і комутатор, до якого вони підключені, дотримуючись шаблону 02 :: XX: YY, де XX - це мережа, а YY - комутатор. Наприклад, хост B3 знаходиться в мережі B (представлений XX = 02) і комутатором SW3 (Y Y = 03), тому його MAC-адреса 02: 02: 03.

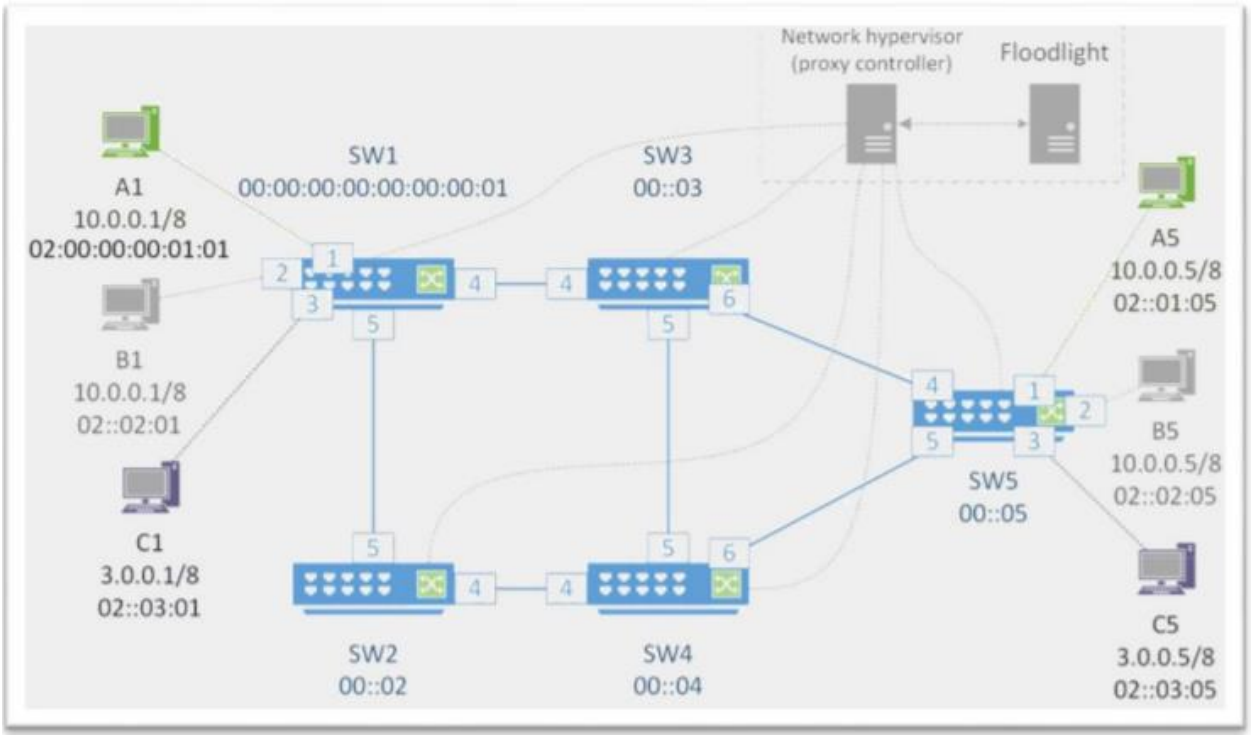

Рис.3.1 Тестовий сценарій Mininet

Рис. 3.1 також показує, що гіпервізор мережі - OVX, VeRTIGO або FlowVisor - підключений до Open vSwitches через інтерфейс зворотної петлі. Той самий інтерфейс зворотного зв'язку використовується для підключення мережевого гіпервізора до Floodlight.

У цій емульованій топології використано п’ять комутаторів, щоб переконатися, що різні методи віртуалізації та функції кожного гіпервізора мережі можуть бути оцінені. Ця мережа може бути використана для

демонстрації нарізки FlowVisor, стійкості OVX до резервних маршрутів, а також можливостей віртуалізації топології OVX та VeRTIGO.

3.2 Ізоляція мережі

Для цього експерименту для перевірки ізоляції мережі серед орендарів використовується топологія Mininet,. По-перше, тестований гіпервізор мережі - FlowVisor, OVX або VeRTIGO - повинен бути налаштований на розділення цієї мережі на три окремі віртуальні мережі, кожна з яких контролюється іншим екземпляром Floodlight.

Очікується, що FlowVisor розділить вихідну мережу, як показано на рисунку 3.2 . Для OVX та VeRTIGO ізоляція мережі повинна виглядати так, як показано на малюнку 18, з великим перемикачем, що представляє п’ять комутаторів. Ізоляцію мережі можна перевірити, помітивши, що хости, що належать до різних зрізів, не можуть спілкуватися. Наприклад, хост A1 не повинен мати можливості відправляти будь-які пакети на будь-який з хостів C1-C5. Крім того, мережевий гіпервізор повинен забезпечити, щоб контролери орендарів мали право контролювати лише їхні мережеві частини.

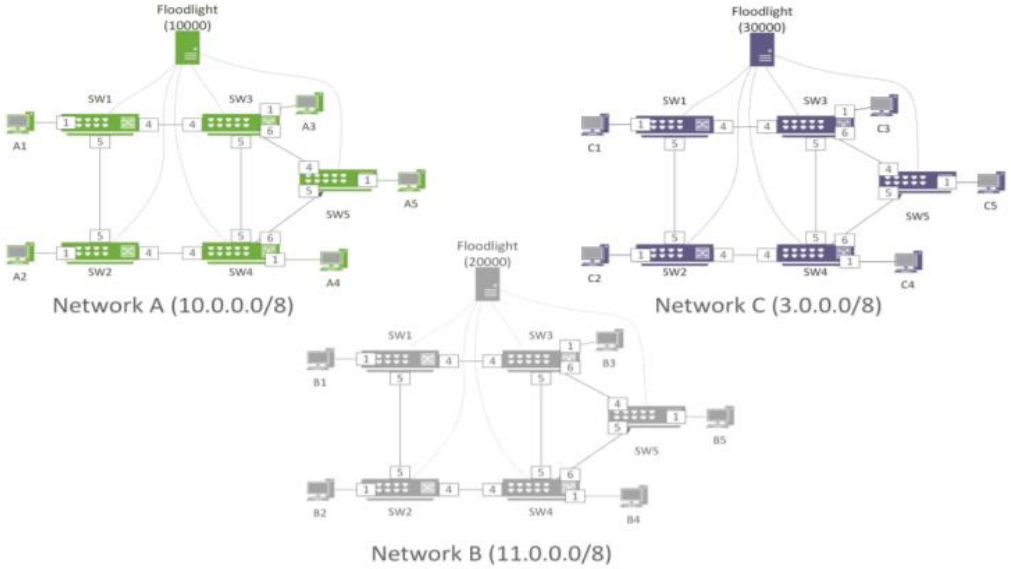

Рис. 3.2 Ізоляція топології за допомогою віртуалізації FlowVisor.

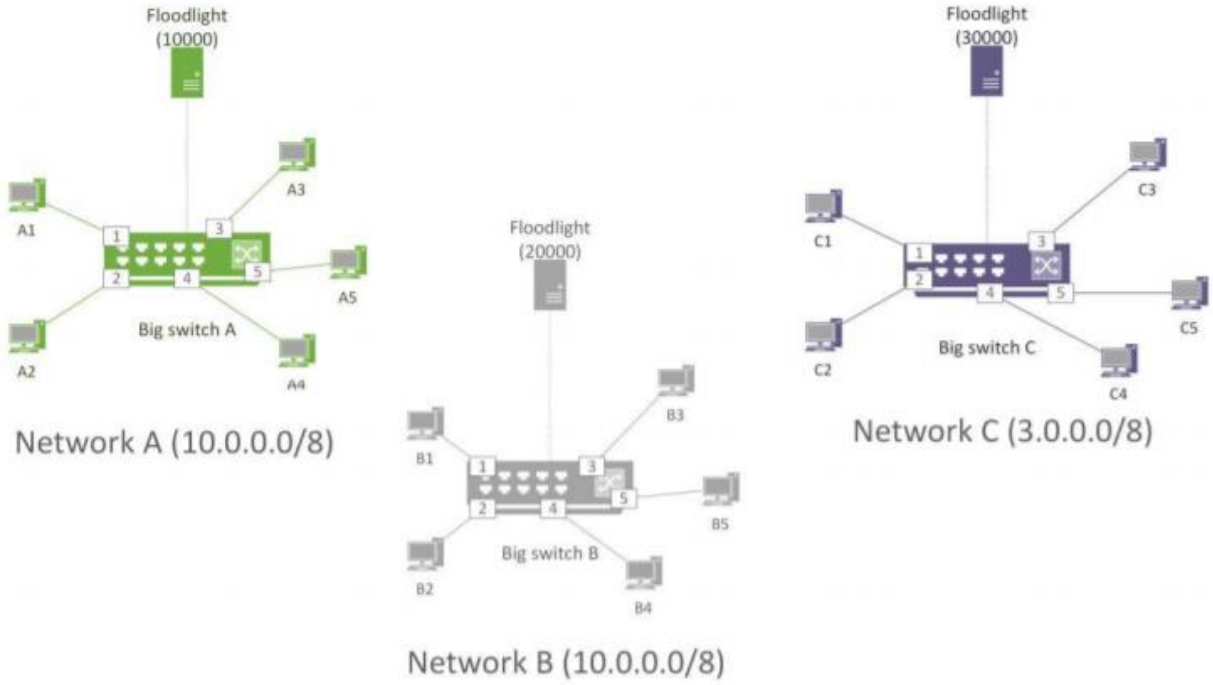

Рис. 3.3 Ізоляція топології за допомогою віртуалізації OVX та VeRTIGO

Графічний інтерфейс Floodlight був використаний для перевірки того, що гіпервізори мережі виставляли правильні подання віртуальної мережі контролерам оренди. Три екземпляри контролера орендаря Floodlight були створені в портах 10000, 20000 та 30000 для управління мережами A, B та C відповідно.

Рисунок 3.4 показує мережу A з точки зору Floodlight, представлену FlowVisor, гіпервізором мережі. Показано п’ять комутаторів з DPID, починаючи з 00 :: 01-05. Усі хости в цій мережі мають MAC-адреси 02 :: 01: XX, і вони правильно відповідають хостам A1-A5. Аналогічні результати можна спостерігати на рисунках 3.5 та 3.6, які містять господарів B1-B5 та господарів C1-C5 відповідно. Це показує, що FlowVisor працює належним чином, дозволяючи ізолювати мережу, розділивши одну мережу на три різні мережі, які працюють незалежно.

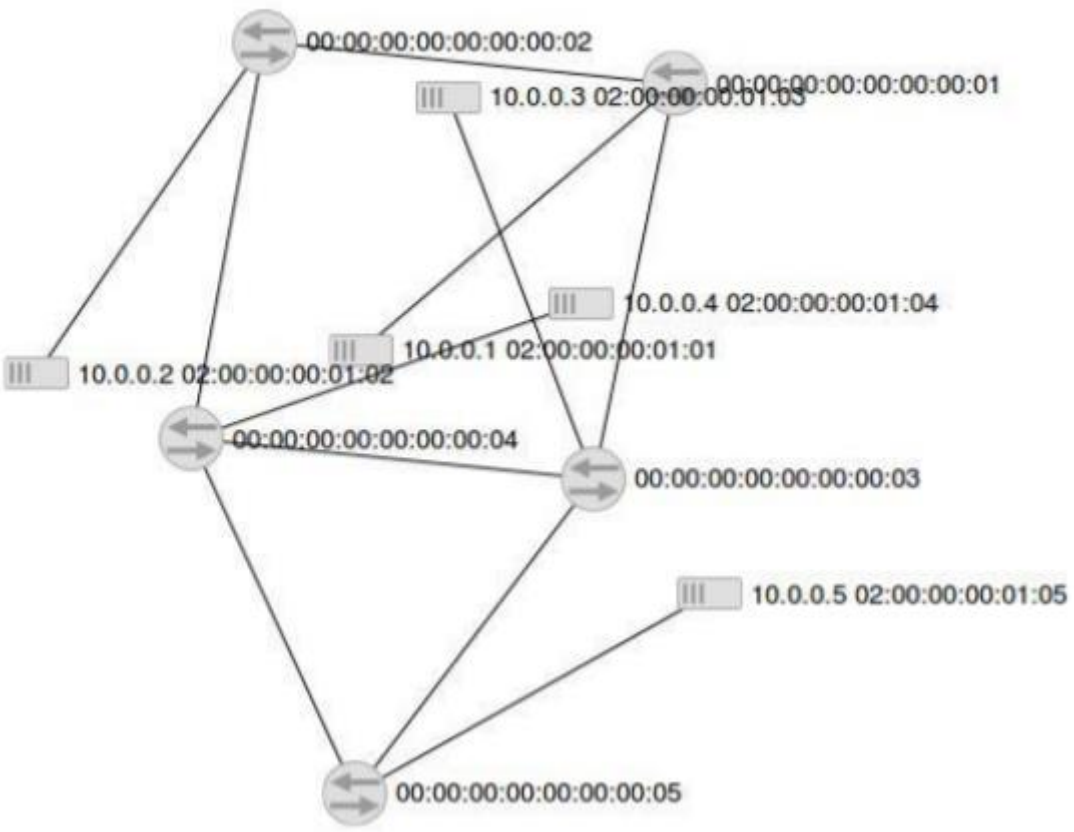

Рис. 3.4 Мережа A, використовуючи FlowVisor як мережний гіпервізор

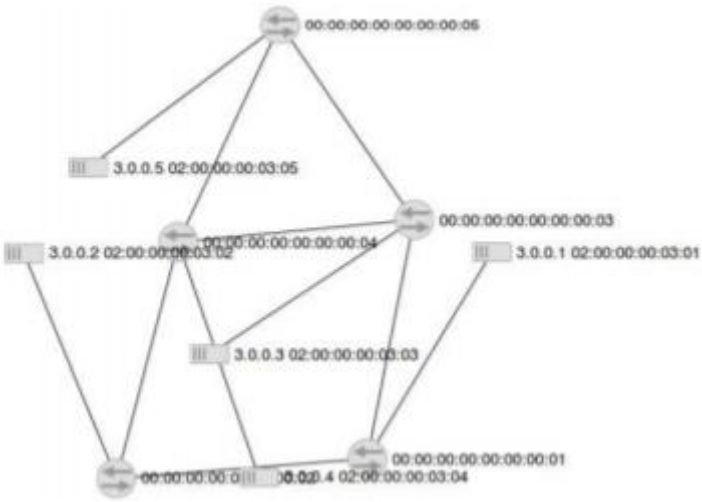

Рис. 3.5 Мережа B

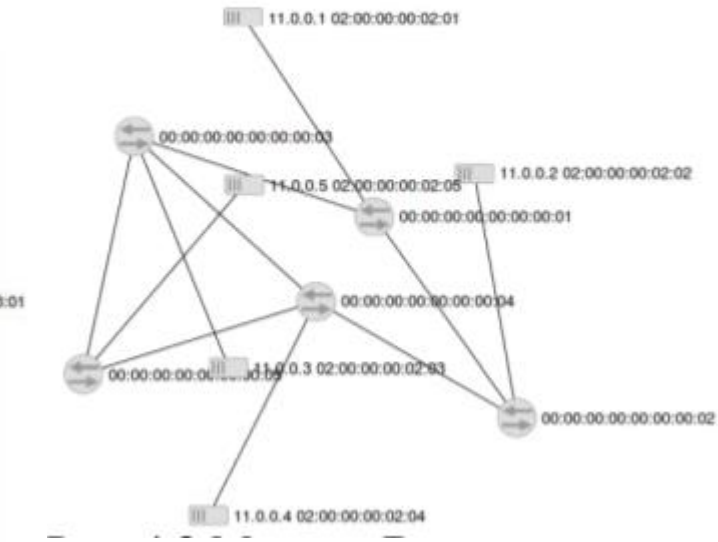

Рис. 3.6 Мережа C

Як спостерігається на рисунку 3.7, примірник Floodlight A «бачить» лише один віртуальний комутатор, представлений OVX. Цей віртуальний комутатор має DPID 00: a4: 05 :: 01 і є лише поданням п’яти комутаторів. Повторюється той самий аналіз, що використовується для FlowVisor, порівнюючи, які хости підключені до кожної мережі. З цього аналізу підтверджено, що хости з MAC-адресою 02 :: 01: XX відображаються лише в мережі A. Те саме можна підтвердити на рисунках 3.8 і 3.9, де хости 02 :: 02: XX з'являються лише в мережі B і хости 02 :: 03: XX лише в мережі C. Отже, OVX також працює належним чином і дозволяє ізолювати мережу, віртуалізуючи єдину мережу на три абстрактні екземпляри, які працюють незалежно

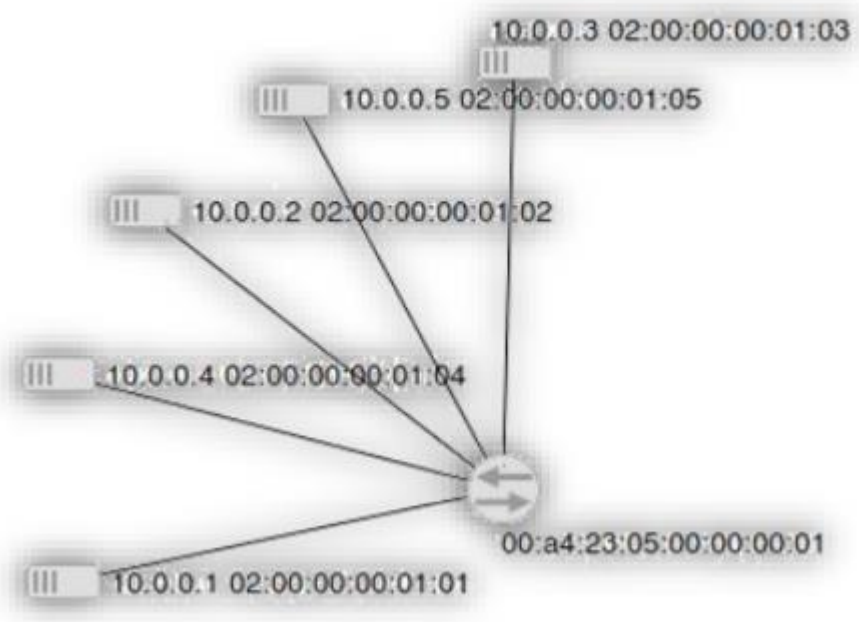

Рис. 3.7 Мережа A з використанням OVX як мережевого гіпервізора.

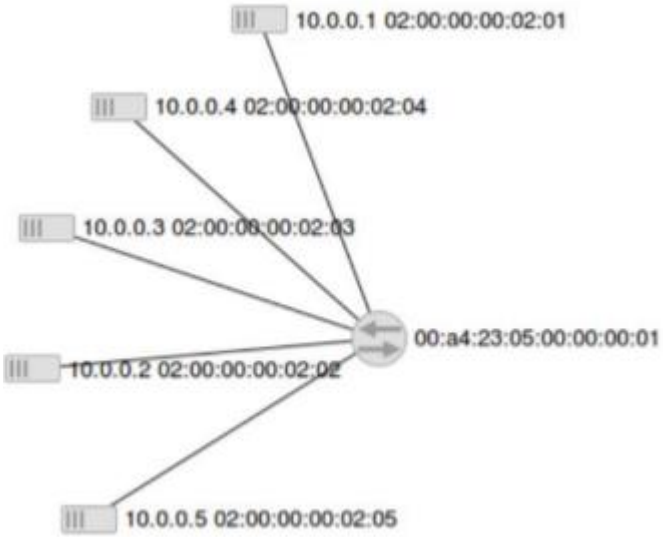

Рис. 3.8 Мережа B з використанням OVX як мережевого гіпервізора

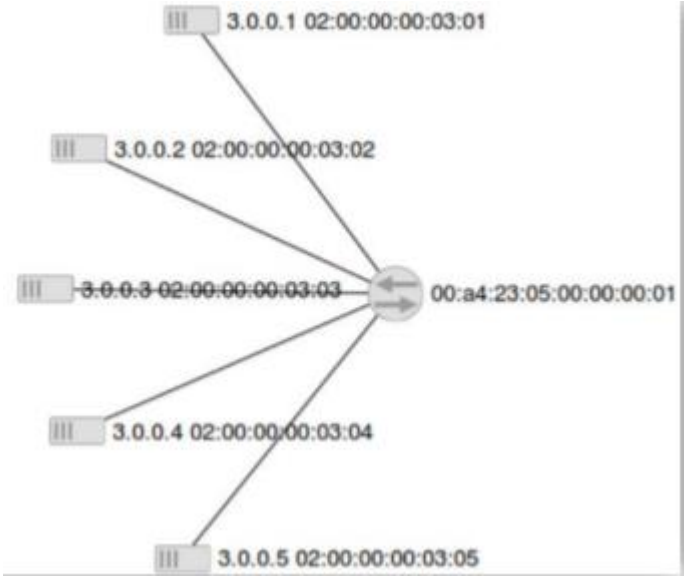

Рис. 3.9 Мережа C з використанням OVX як мережевого гіпервізора

3.3 Прозоре пересилання трафіку

Орендарі мережі можуть захотіти запустити традиційні розподілені мережеві програми у власній віртуальній мережі, такі як механізм виявлення LLDP або протокол маршрутизації спочатку відкритого найкоротшого шляху (OSPF). Як LLDP, так і OSPF покладаються на використання багатоадресних кадрів для зв'язку з сусідніми мережевими пристроями. Віртуалізація мережі зазвичай добре перевірена для стандартного одноадресного IP-трафіку, але деякі багатоадресні кадри та інші спеціальні типи кадрів навряд чи колись перевіряються на прозору переадресацію трафіку в експериментальних гіпервізорах мережі. Тому важливо перевірити, що кілька різних типів трафіку можуть пересилатись віртуальними мережами.

Рис. 3.10 Еталонне прозоре пересилання трафіку.

Один із способів перевірки прозорості переадресації руху полягає у використанні інструменту test_packets.py. Користувач повинен вводити кадри в один мережевий порт, а потім очікувати, що кадри будуть отримані на будь-якому іншому хості призначення в мережі. Очікується, що мережа переадресовує кадри до портів, що належать до правильного фрагмента або віртуальної мережі, не скидаючи та не змінюючи їх. Різні гіпервізори мережі не повинні додатково обмежувати тип трафіку, який зазвичай може пересилатись лише контролером Floodlight.

За допомогою інструменту test_packets.py було перевірено багато типів трафіку, щоб оцінити, чи блокують контролери віртуалізації певні кадри. Узагальнені результати наведені в таблиці 2 та класифіковані на шість різних категорій. Усі категорії є зрозумілими, і більш докладно про те, які пакети були використані для тестування, можна знайти в [51]. На рисунку 3.11 порівнюються результати переадресації трафіку за допомогою стандартного перемикача рівня 2 (стовпець L2), Open vSwitch (OVS), Floodlight (FL), FlowVisor (FV) та OpenVirteX (OVX). Зверніть увагу, що результати VeRTIGO тут не відображаються, оскільки програма VeRTIGO може натрапити на винятки та перестати працювати під час тестів..

Type of traffic	L2	OVS	FL	FV	OVX
IPv4 unicast	+	+	+	+	+
IPv6 unicast	+	+	+	-	-
IPv4 multicast	+	+	+	+	+
L2 multicast	+-	-	+-	+-	+-
IPv4 w/ VLAN	+	+	+	+	+
IPv4 w/ MPLS	+	+	+	+	+

Рис. 3.11 Узагальнені результати пересилання трафіку.

І мережевим гіпервізорам OVX і FlowVisor вдалося переслати все, окрім:

- пакети IPv6;
- Два типи багатоадресних пакетів L2: протокол виявлення рівня зв'язку (LLDP) і протокол охоплюючого дерева (STP).

Контролер Floodlight підтримував переадресацію IPv6, хоча його потрібно було налаштувати за допомогою OpenFlow 1.3, що не підтримується FlowVisor та OpenVirteX. Однак Floodlight все ще не може пересилати кадри LLDP та STP. Це обмеження очікується, оскільки ці кадри перехоплюються програмою контролера для цілей виявлення мережі або обробляються / скидаються Open vSwitches.

Open vSwitch, налаштований як традиційний комутатор рівня 2, успішно вдається переадресувати всі типи трафіку, за винятком пакетів багатоадресної передачі L2, таких як LLDP, STP, протокол агрегування посилань (LACP) та протокол виявлення Cisco (CDP). Ці кадри використовують зарезервовані багатоадресні адреси призначення MAC у діапазонах 01: 80: c2: 00: 00: xx та 01: 00: 0c: cc: cc: xx, і, як правило, обробляються мостами Ethernet. З OVS, налаштованим як самостійний міст, прийнятно, щоб ці кадри були оброблені або скинуті, а не залиті в іншу частину мережі.

Стандартний комутатор рівня 2 - комутатор NETGEAR Fast Ethernet FS108 - також використовувався як базовий для порівняння з іншими результатами. Йому вдалося переадресувати всі типи трафіку, за винятком декількох фреймів багатоадресної передачі L2:

- Операції, адміністрування та обслуговування (OAM), протокол, який використовується для виявлення проблем з підключенням у мережах рівня 2;
- LACP;

Загалом, перевірені мережеві гіпервізори напрочуд добре перенаправляли кілька типів трафіку. Єдині загальні проблеми, що спостерігаються, трапляються з трафіком IPv6, LLDP та STP. IPv6 не підтримується OpenFlow 1.0, який використовується як OVX, так і FlowVisor. LLDP і STP використовуються як

протоколи управління мережами Ethernet. Хоча це обмеження очікується, оскільки ці протоколи призначені для обробки мережевими гіпервізорами, це потенційно серйозне обмеження для клієнтів, які бажають запускати власні протоколи управління рівня 2 у своїх віртуальних мережах.

Загалом, перевірені мережеві гіпервізори напрочуд добре передавали кілька типів трафіку. Єдині загальні проблеми, що спостерігаються, трапляються з трафіком IPv6, LLDP та STP. IPv6 не підтримується OpenFlow 1.0, який використовується як OVX, так і FlowVisor. LLDP і STP використовуються як протоколи управління мережами Ethernet. Хоча це обмеження очікується, оскільки ці протоколи призначені для обробки мережевими гіпервізорами, це потенційно серйозне обмеження для клієнтів, які хочуть запускати власні протоколи управління рівня 2 у своїх віртуальних мережах.

3.4 Час налаштування потоку

Тести на затримку були можливі лише для FlowVisor та OVX. VeRTIGO спричинив скидання перших пакетів потоку під час установки потоку. Тому тест тестування пінгу не міг виміряти час налаштування потоку для VeRTIGO.

Цей експеримент проводився перевіряв випадки встановлення встановлення потоку для:

- Floodlight;
- FlowVisor та Floodlight;
- Немає контролера, який використовує перемикач у традиційному режимі переадресації рівня 2;
- OVX та Floodlight.

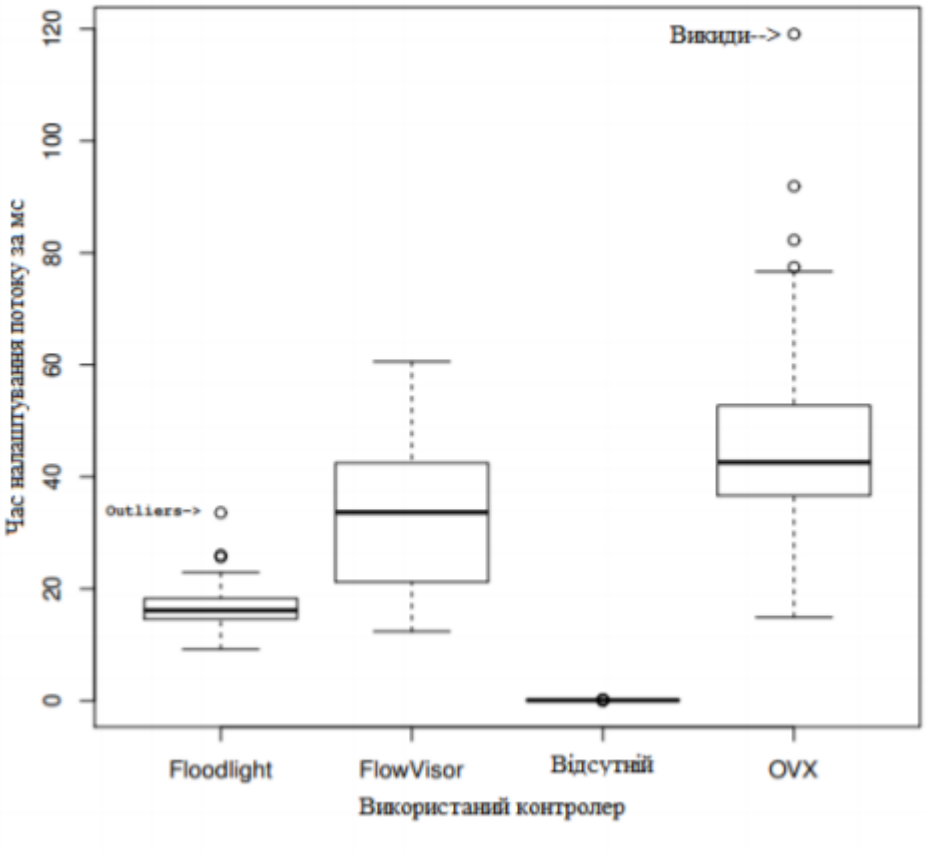

Рис. 3.12. Час затримки першого пінгу

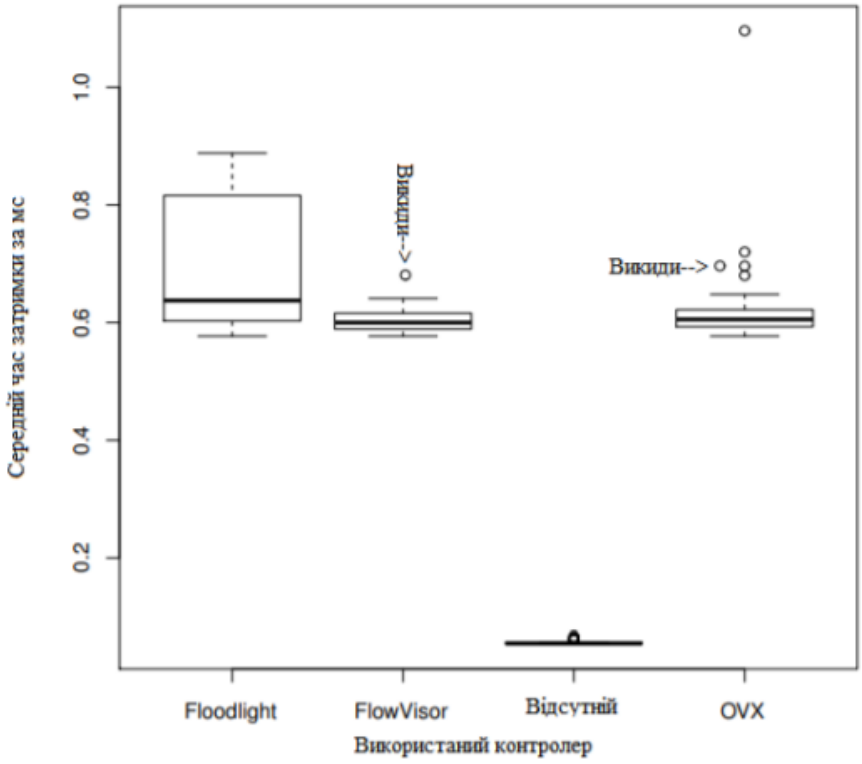

Рис. 3.13 Середній час затримки наступних пінгів

Як показано на рисунку 3.12, час налаштування потоку лише за допомогою Floodlight становив в середньому менше 20 мс, тоді як час налаштування потоку FlowVisor плавав близько 20-40 мс, а час налаштування потоку OVX - близько 40-50 мс. Дисперсія часу встановлення потоку була вищою у випадку OVX, і цей результат, ймовірно, пов'язаний з тим, що OVX покладається на складні правила потоку, щоб переписати трафік при вході або виході з мережі. Автономний перемикач рівня 2 давав в середньому 0,08 мс часу налаштування потоку. Це представлено контролером “None”. Ці результати відповідали очікуванням: час налаштування потоку набагато вищий при використанні мережевих гіпервізорів між мережею та контролерами орендаря, а програмно-визначені мережі демонструють значні накладні витрати на час налаштування потоку в порівнянні з традиційними мережами.

Рисунок 3.13 показує, що середня RTT пінгів, ініційованих після налаштування потоку, взагалі не залежить від контролера. Це очікується після встановлення потоків, оскільки контролер більше не впливає на затримку між хостами. Крім того, саме за цим сюжетом можна спостерігати різницю між програмним та апаратним режимами переадресації. Автономний режим переадресації комутатора (Controller = None) використовує перевагу апаратної переадресації швидкості передачі, що призводить до приблизно в 10 разів швидших RTT і набагато менших дисперсіях.

Результати експериментів, проведених у фізичній топології, зведені на рисунку 33. Пропускна здатність з FlowVisor в середньому становить ≈617,9 кбіт / с, з OVX ≈618,6 кбіт / с та VeRTIGO ≈ 620,0 кбіт / с. Однак розбіжність у всіх результатах досить велика, і статистично не спостерігається суттєвої різниці в отриманій смузі пропускання, виміряній між двома хостами при переході на різні мережеві гіпервізори. Цей результат був очікуваним, оскільки вузьким місцем цього експерименту був процесор комутації. Висока завантаженість процесора під час експерименту є найімовірнішою причиною того, що в наборі даних «FlowVisor + Floodlight» є точка відхилення даних. Прості завдання під час експерименту, такі як відкриття сеансу telnet

комутатору, можуть вплинути на пропускну здатність, оскільки вони вимагають від ЦП тимчасової зупинки обробки трафіку

3.5 Пропускна здатність

Тест пропускної здатності базується на iperf в режимі TCP. TCP є найкращим протоколом, і досягнута пропускна здатність залежить від затримки мережі, розміру кадру, втрати пакетів, між кадрового розриву та інших умов мережі. Очікувана пропускна здатність у локальній мережі з низькою затримкою повинна знаходитися в межах 90-100% від діапазону автоматичної узгодженої лінії зв'язку 100 Мбіт / с. Однак на результати тестів вплинула низька продуктивність комутатора HP-3500yl-24G, а виміряна пропускна здатність становила менше 1 Мбіт / с

Результати експериментів, проведених у фізичній топології, показані на рисунку 3.14. Пропускна здатність з FlowVisor в середньому становить ≈617,9 кбіт / с, з OVX ≈618,6 кбіт / с та VeRTIGO ≈ 620,0 кбіт / с. Однак розбіжність у всіх результатах досить велика, і статистично не спостерігається суттєвої різниці в отриманій смузі пропускання, виміряній між двома хостами при переході на різні мережеві гіпервізори. Цей результат був очікуваним, оскільки вузьким місцем цього експерименту був процесор комутації. Висока завантаженість процесора під час експерименту є найімовірнішою причиною того, що в наборі даних «FlowVisor + Floodlight» є точка відхилення даних. Прості завдання під час експерименту, такі як відкриття сеансу telnet комутатору, можуть вплинути на пропускну здатність, оскільки вони вимагають від ЦП тимчасової зупинки обробки трафіку

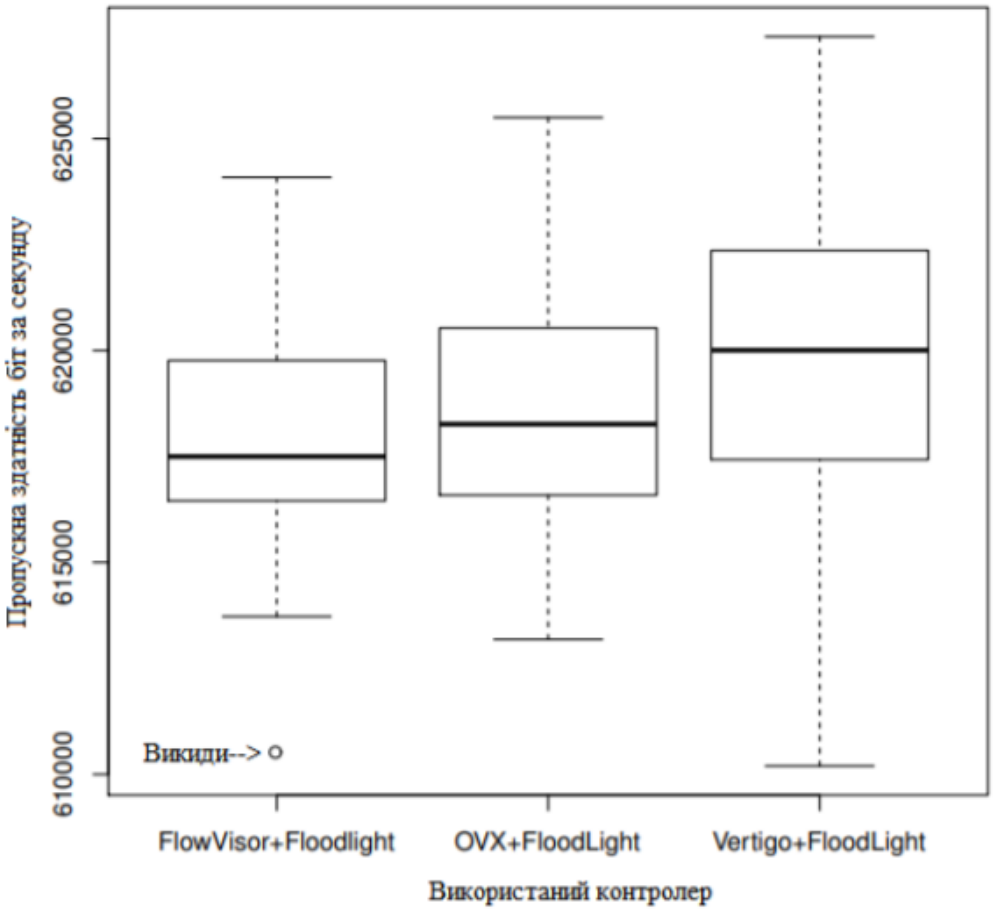

Рис. 3.14 Пропускна здатність, виміряна різними контролерами в бітах за секунду

Висновок

У цьому розділі було порівняно три гіпервізори за рядом показників.

Два із трьох додатків для віртуалізації мережі були успішно оцінені запропонованими тут експериментами. І FlowVisor, і OVX виправдали сподівання і протягом більшості експериментів працювали добре відповідно до своєї документації. Більшість експериментів з VeRTIGO не вдалося завершити через обмежену документацію та проблеми взаємодії з OVS та Floodlight. VeRTIGO мав кілька обмежень і був використаний лише для дуже простих тестів з апаратним комутатором. Спроби використовувати віртуальні посилання, щоб дозволити різні віртуальні топології, спричинили винятки у Floodlight при обміні повідомленнями з VeRTIGO.

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

В результаті виконання роботи було виявлено, що два із трьох додатків для віртуалізації мережі були успішно оцінені запропонованими тут експериментами. І FlowVisor, і OVX виправдали сподівання і протягом більшості експериментів працювали добре відповідно до своєї документації. Більшість експериментів з VeRTIGO не вдалося завершити через обмежену документацію та проблеми взаємодії з OVS та Floodlight. VeRTIGO мав кілька обмежень і був використаний лише для дуже простих тестів з апаратним перемикачем. Спроби використовувати віртуальні посилання, щоб дозволити різні віртуальні топології, спричинили винятки у Floodlight під час обміну повідомленнями з VeRTIGO.

FlowVisor не цікавить, використовує мережевий трафік IP чи ні, доки мережевий контролер здатний обробляти мережевий трафік, що піддається механізму нарізки FlowVisor. OVX, виходячи з поточного механізму роботи, передбачає, що контрольовані мережі працюватимуть з IPv4. Проте обидва інструменти віртуалізації мережі обмежені можливостями OpenFlow, які вони підтримують. Наприклад, в даний час FlowVisor та OVX не підтримують IPv6, оскільки вони можуть обробляти лише OpenFlow 1.0, тоді як більшість реалізацій IPv6 пропонуються лише версіями 1.2 та 1.3.

Як VeRTIGO, так і OVX стверджували, що мають підтримку автономного перенаправлення маршрутів у разі відмови каналу. VeRTIGO не можна було перевірити через проблеми сумісності. Функція автономного перенаправлення OVX спрацювала, але все ще не може гарантувати автономне відновлення, якщо хост клієнта не дозволить правилам потоку закінчуватися, перш ніж повторно відправляти трафік. Виміряти час відновлення мережі неможливо, оскільки функція перенаправлення залежить від використання мережі.

Тести пропускної здатності показали, що контролери проксі-сервера віртуалізації мережі можуть значно зменшити пропускну здатність, якщо встановлені потоки не підтримуються обладнанням пристрою. В

експериментах, проведених у місцевій фізичній лабораторії, жодне з мережевих додатків не підтримувалось обладнанням OpenFlow, і всі контролери працювали погано через обмеження перемикача OpenFlow.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Логинов С.С. Об уровнях управления в программно-конфигурируемой сети (SDN) // T-Comm: Телекоммуникации и транспорт. – 2017.

2. POX. (n.d.). The POX Controller Repository, [Online]. Available: https:// github.com/noxrepo/pox (visited on 03/30/2016).

3. Semenovykh A.A. Comparative analysis of SDN controllers. / Semenovykh A.A., Laponina O.R. // International Journal of Open Information Technologies ISSN: 2307-8162 vol. 6, no.7., 2018 – P. 50-56.

4. Stallings W. Software-defined networks and openflow // The internet protocol Journal. – 2013. – Т. 16. – №. 1. – С. 2-14. – Режим доступу: http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/tablecontents-59/161-sdn.html

5. OpenFlow Switch Specification Version 1.4.0 (Wire Protocol 0x05). – 2013. – Режим доступу: https://www.opennetworking.org/images/stor ies/downloads/sdn-resources/onfspecifications/openflow/openflow-specv1.4.0.pdf

6. F. Pakzad, M. Portmann, W. L. Tan, and J. Indulska, “Efficient topology discovery in software defined networks,” in Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on, Dec. 2014, pp. 1–8. doi: 10.1109/ICSPCS.2014.7021050.

7. D. Bombal. (n.d.). Datapath IDs, [Online]. Available: http : / / pakiti . com / datapath-ids/ (visited on 04/01/2016).

8. Ефимушкин В.А. Обзор решений SDN/NFV зарубежных производителей / Ефимушкин В.А., Ледовских Т.В., Корабельников Д.М., Языков Д.Н. // T-Comm: Телекоммуникации и транспорт. – 2015. – Том 9. – №8. – С. 5-13.

9. N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, “NOX: Towards an operating system for networks,” ACM SIGCOMM CCR, vol. 38, no. 3, 2015.

10. M. R. Nascimento, C. E. Rothenberg, M. R. Salvador, C. N. A. Corrˆea, S. C. de Lucena, and M. F. Magalh˜aes, “Virtual routers as a service: The RouteFlow approach leveraging software-defined networks,” in Conference on Future Internet Technologies (CFI)

11. “IEEE Standard for Ethernet,” IEEE Std 802.3-2012 (Revision to IEEE Std 802.3-2008), pp. 1–3747, Dec. 2012. doi: 10.1109/IEEESTD.2012.6419735.

12. J. van der Merwe and I. Leslie, “Switchlets and dynamic virtual ATM networks,” in IFIP/IEEE International Symposium on Integrated Network Management, 2018.

13. R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T.-Y. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K.-K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, R. Johari, N. McKeown, and G. Parulkar, “Carving Research Slices out of Your Production Networks with OpenFlow,” SIGCOMM Comput. Commun. Rev., vol. 40, no. 1, pp. 129–130, Jan. 2010. doi: 10.1145/1672308.1672333. [Online]. Available: http://doi.acm.org/10.1145/1672308.1672333.

14. A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, “In VINI veritas: Realistic and controlled network experimentation,” in ACM SIGCOMM, 2009.

15. NOX. (n.d.). The NOX Controller Repository, [Online]. Available: https://github.com/noxrepo/nox (visited on 03/30/2016).

16. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling innovation in campus networks,” ACM SIGCOMM CCR, vol. 38, no. 2, pp. 69–74, 2015.

17. N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, “NOX: Towards an operating system for networks,” ACM SIGCOMM CCR, vol. 38, no. 3, 2013.

18. ——, (n.d.). Openvirtex architecture, [Online]. Available: http://ovx.onlab.us/documentation/architecture/overview/ (visited on 03/13/2016).

19. Floodlight. (n.d.). Floodlight project, [Online]. Available: http://www.projectfloodlight.org/ (visited on 03/23/2016).

20. B. Pfaff, J. Pettit, T. Koponen, E. J. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon, and M. Casado, “The Design and Implementation of Open vSwitch,” in Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation, ser. NSDI’15, Oakland, CA: USENIX Association, 2015, pp. 117–130. [Online]. Available: http://dl.acm.org/citation.cfm?id=2789770.2789779.

21. Badach, Anatol; Hoffmann, Erwin: Technology of IP networks. Verlag Hanser, 2019, ISBN: 978-3-446-45511-5

22. Schulte, Heinz (ed.): Protocols and Services of the Information's technology. WEKA publishing house, ISBN: 978-3824540662

23. R. Fisli, "Secure Corporate Communications over VPN-Based WANs," Master's Thesis in Computer Science at the School of Computer Science and engineering, Royal Institute of Technology, sweden, 2012.

24. E. C. Rosen, Y. Rekhter, "BGP/MPLS VPNs," IETF RFC 2547, March 2017.

25. Wanger, and Schneier, "Analysis of the SSL 3.0 protocol," The Second USENIX Workshop on Electronic Commerce Proceedings, pp. 29-40, 2014.