УДК 621.3
DOI https://doi.org/10.32782/2663-5941/2025.5.1/12

Романов О.І.
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Нестеренко М.М.
Військовий інститут телекомунікацій та інформатизації імені Героїв Крут

Марінов А.І.
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

МОДИФІКОВАНИЙ БАГАТОШЛЯХОВИЙ МЕТОД БАЛАНСУВАННЯ НАВАНТАЖЕННЯ В SDN МЕРЕЖАХ ДАТА-ЦЕНТРІВ

Стаття присвячена дослідженню та вдосконаленню методу балансування навантаження у програмно-конфігурованих мережах (SDN) дата-центрів. Актуальність проблеми обумовлена необхідністю ефективного використання наявних мережевих ресурсів та забезпечення високих показників якості обслуговування (QoS) в умовах динамічного середовища центрів обробки даних. Сучасний розвиток мереж SDN обумовлює широке застосування даної технології для розгортання мережевої інфраструктури центрів обробки даних. Вирішення цієї проблеми досягається використанням методів інжинірингу трафіку (TE), які дозволяють підвищити продуктивність мережі та якість обслуговування за рахунок багатошляхової маршрутизації в поєднанні з класифікацією та пріоритезацією трафіка. Застосування TE також спрямоване на зниження витрат на експлуатацію та запобіганню появи перевантажених сегментів в мережі. У статті проведено аналіз і порівняння існуючих технологій та інженерних рішень для оптимізації мережевих ресурсів, зокрема B4 (Google), Hedera, DevoFlow, Mahout, MiceTrap та MSDN-TE. Розглянуто підходи (Hedera, DevoFlow, Mahout, MiceTrap), які використовують класифікацію потоків на великі "elephant-flows" та малі "mice-flows" з метою розвантаження централізованого SDN-контролера та підвищення ефективності обслуговування, а також MSDN-TE, який використовує багатошляхову маршрутизацію у поєднанні з алгоритмом Еппштейна, який обраховує найменш завантажений шлях, мінімізуючи затримки та втрати пакетів шляхом обчислення ваги кожного шляху передачі на основі певних мережевих метрик. У даній роботі запропоновано модифікований метод балансування навантаження на основі алгоритму Еппштейна. Вдосконалення полягає у можливості одночасного використання декількох рівнозначних шляхів передачі для трафіку реального часу між джерелом і отримувачем. Цей підхід застосовується, коли пропускної спроможності основного шляху недостатньо для забезпечення необхідних показників QoS (Delay, Packet Loss, Jitter). З метою підтвердження ефективності, було розгорнуто віртуальний сегмент SDN мережі дата-центру (за допомогою ONOS та Mininet). Результати дослідження підтверджують, що динамічне балансування навантаження, яке спирається на відсоток завантаження шляхів передачі у реальний час, є ефективним рішенням для підвищення показників QoS, нівелювання утворення перевантажених сегментів (bottlenecks) та забезпечення надійності передачі трафіку в SDN мережах дата-центрів.

Ключові слова: *програмно-конфігуровані мережі (SDN), дата-центри, якість обслуговування (QoS), інжиніринг трафіку (TE), передача трафіку, балансування навантаження в мережі, OpenFlow, багатошляхова маршрутизація.*

Постановка проблеми. Сучасний розвиток мереж SDN вказує на широке застосування даної технології для розгортання мережевої інфраструктури розподілених центрів обробки даних (Data Center). При чому динамічна природа додатків та сервісів, а також різнорідність технологій транспортних мереж вимагають впровадження додаткових механізмів, що дозволяють ефективно використовувати наявні мережеві ресурси та забезпечувати необхідні показники якості обслуговування QoS.

Одним із способів вирішення даної проблеми є використання методів traffic engineering (TE), які

дозволяють здійснювати балансування навантаження в рамках всієї мережі за рахунок використання багатошляхової маршрутизації в поєднанні з класифікацією та пріоритезацією трафіка, що стало можливим при централізації управління в мережах SDN. Іншими словами, TE дозволяє підвищити продуктивність мережі та якість обслуговування (QoS) [1, с. 4377–4393] користувачів за рахунок ефективного та раціонального використання наявних ресурсів [2, с. 33–40]. Застосування методів "інжинірингу трафіку" також може знизити витрати на експлуатацію та запобігти утворенню перевантажених сегментів (bottlenecks) в мережі.

На даний час існує велика кількість підходів, методів та реалізацій різних технологій для оптимізації мережевих ресурсів з метою досягти належного значення показників якості обслуговування. Тому варто спочатку провести аналіз та порівняння існуючих технологій та архітектурних рішень таких як: B4, Hedera, DevoFlow [3, с. 254–265], Mahout [4, с. 1629-1637], MiceTrap для визначення принципів роботи методів балансування в дата-центрах та мережах SDN. В подальшому в даній статті запропонований модифікований метод балансування на основі протоколу багатошляхової маршрутизації за рахунок одночасного використання декількох рівнозначних шляхів передачі між джерелом і отримувачем повідомлень (Source-Destination).

Аналіз останніх досліджень і публікацій. Розглянемо основні технології, моделі, та протоколи, які використовуються для балансування трафіка в Software Defined WAN (SD WAN) та мережах дата-центрів:

B4 (ECMP на основі хешування) являє собою практичне рішення від компанії Google, що дозволяє з'єднувати дата-центри компанії по всьому світу між собою. B4 була створена для вирішення проблем у глобальній мережі (WAN), таких як: надійність, системні відмови та продуктивність мережі. Основна ідея полягає в розподілі пропускної спроможності між сервісами, що між собою конкурують за рахунок динамічної зміни структури трафіку з метою запобігання збоїв/відмов у інфраструктурі (мережі) дата-центрів. Також B4 розроблявся задля підтримки та швидкого розгортання (впровадження) нових або стандартних протоколів, функцій керування мережею, трафіком. Однією із імплементованих функцій є механізм "інжинірингу трафіку", який дозволяє додаткам гнучко і швидко адаптуватися відповідно до поточного стану мережі, або у разі виникнення аварійних ситуацій.

Важливо, що така архітектура використовує маршрутизацію та "інжиніринг трафіку" як окремі сервіси, а методи TE розміщуються над протоколами маршрутизації. Це дозволяє системі управління мережею використовувати стратегію „повернення до базових протоколів маршрутизації", адже довгострокові відмови в глобальних мережах можуть призвести до величезних збитків. Тобто, якщо служба TE зазнає збоїв в роботі, то в подальшому пакети в мережі пересилаються за допомогою класичних протоколів із визначенням найкоротших шляхів.

На рис. 1. Представлено трирівневу архітектуру відповідно до підходу B4.

Зображена вище архітектура B4 з інтеграцією методів TE складається з наступних логічних рівнів:

– глобальний рівень, дозволяє централізовано керувати всією глобальною мережею за допомо-

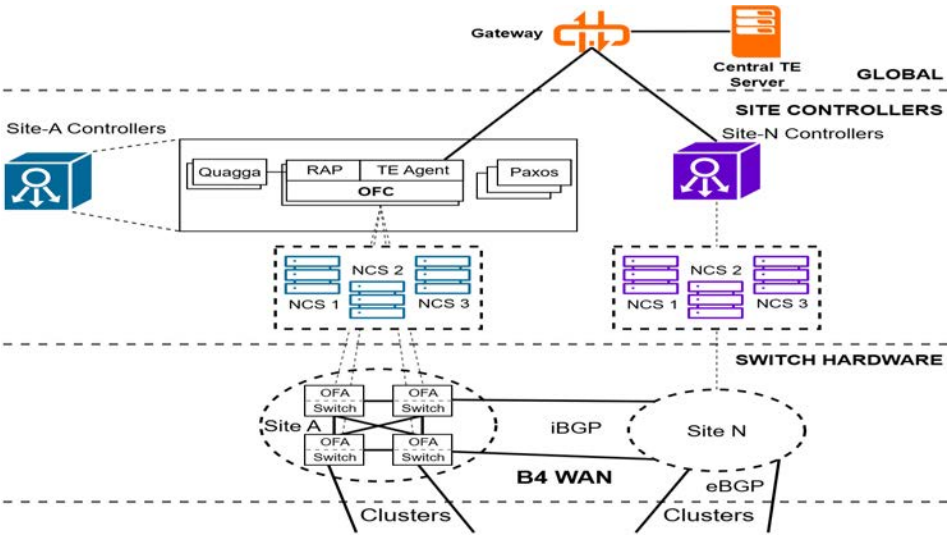

Рис. 1. Архітектура B4 (Google)

гою централізованих додатків, таких як центральний сервер керування трафіком Central TE Server (CTE) та Gateway (шлюз SDN), який дозволяє централізовано керувати мережею (міст між CTE та рівнем нижче);

– рівень контролера області відповідальності (site controllers), який включає контролер OpenFlow і додатки для управління мережею, такі як сервіси маршрутизації;

– комутаційний апаратний рівень (switch hardware), що включає комутатори та на якому виконується переадресація трафіку.

TE Server виконує центральну і головну роль у керуванні трафіком. Він є логічно централізованим додатком у глобальному шарі архітектури B4. Його метою є розподіл пропускної здатності між додатками у межах компанії, при можливості застосовуючи кілька шляхів передачі, аби забезпечити належне виконання функції справедливого розподілу max-min (Min-Max fairness). TE Server оперує інформацією про мережеву топологію, Flow Groups (групами потоків), Tunnels (тунелями) та Tunnel Groups (групами тунелів). Також TE Server генерує Tunnel Groups, Tunnels та Flow Groups, які потім передаються до SDN Gateway, а звідти до контролеру для встановлення політик/правил на комутаторах.

Описана вище архітектура працює за наступним принципом. На найвищому рівні знаходиться CTE, який відповідає за: визначення незадіяної пропускної спроможності у мережі для багатошляхової маршрутизації; розподіл і коригування потреб у необхідних ресурсах між сервісами; динамічне перенаправлення трафіку з несправних каналів зв'язку або комутаторів. Gateway SDN надає загальну картину топології мережі для CTE. CTE використовує це загальне представлення мережі для обчислення агрегованого трафіку на межі між зонами відповідальності (site controllers). Потім абстрактний результат обчислень подається алгоритму оптимізації TE задля рівномірного розподілу ресурсів між групами додатків/сервісів.

Для досягнення рівномірного розподілу мережевих та апаратних ресурсів, використовується алгоритм Min-Max „справедливого розподілу" (Min-Max Fairness). На основі наданих пріоритетів сервісам, алгоритм оптимізації TE і розподіляє відповідно пропускну спроможність між ними.

Основними принципами Min-Max Fairness є:

– рівномірний розподіл: усі потоки отримують рівну частку ресурсу, доки хоча б один із них не досягає своєї максимально допустимої потреби.

– пріоритет найменш забезпечених: якщо деякі потоки вже досягли своїх меж (отримали свою мінімально необхідну пропускну спроможність), доступний ресурс перерозподіляється серед тих потоків, кому його ще не вистачає.

– гарантований мінімум: потоки отримують рівний розподіл ресурсів (наприклад: доступну пропускну спроможність) до тих пір, поки конкуруючі потоки не отримують обмеження, після чого інші зможуть отримати більше ресурсів.

Така взаємодія відбувається на відповідних рівнях в архітектурі B4. Адже Max-Min Fairness являється свого роду стратегією централізованої системи TE, яка визначає ідеальний або оптимальний розподіл трафіку між різними шляхами передачі. В свою чергу протокол ECMP – це механізм на рівні апаратного забезпечення комутаторів, який дозволяє реалізує цей розподіл, фізично хешуючи пакети та надсилаючи їх через обрані тунелі або лінки відповідно до встановлених TE коефіцієнтів.

Досягнення справедливого розподілу смуги пропускання між додатками/службами означає максимізацію використання смуги пропускання, не знижуючи при цьому справедливої частки смуги пропускання для всіх програм у системі.

В даній стратегії задіється централізований алгоритм оптимізації TE, який, в свою чергу, визначає як розподілити смугу пропускання між групами потоків (Flow Groups, FG) за допомогою функцій смуги пропускання, пріоритезуючи їх на „вузьких місцях" мережі. Результатом цього алгоритму є створення тунельних груп (Tunnel Groups, TG), які зіставляють FG з набором тунелів та відповідними вагами. Вага визначає частку трафіку FG, яка має бути перенаправлена по кожному тунелю. Це дозволяє врахувати обмеження обладнання при реалізації справедливого розподілу Max-Min.

Вже на етапі пересилання пакетів комутатором починає застосовуватись ECMP. Тобто, коли сервер TE визначив бажані тунельні групи та їхні коефіцієнти розподілу, він надсилає ці інструкції до OpenFlow контролерів (OFCs), які потім надають відповідні інструкції на підпорядковані комутатори за допомогою OpenFlow.

А саме, комутатори в B4 використовують хешування ECMP для розподілу потоків між тунелями в межах одного міжсайтового (всередині одного Site) з'єднання. Тобто на третьому рівні архітектури B4, для балансування навантаження між декількома шляхами використовується протокол Equal-Cost Multi-Path (ECMP) на основі хешування, це дозволяє мережі адаптувати пропускну спроможність на користь ефективного використ-

тання доступних шляхів (з однаковою вартістю) між вузлами без перевантаження одного з них, що в рази зменшує ймовірність перевантаження каналів зв'язку і призводить до зменшення затримок при передачі трафіку.

В свою чергу, хеш-функція у ECMP відповідає за передачу усіх пакетів одного потоку через однаковий маршрут, що запобігає проблемі зі змішуванням пакетів. Порядок хешування являє собою дописування на комутаторах відповідних міток до заголовків вхідних пакетів, та присвоєння однакового шляху для передачі усіх пакетів потоку (розуміються усі пакети, які мають спільні поля/заголовки, наприклад однакову адресу отримувача чи відправника).

Таким чином рішення B4 від Google має на меті підтримувати на належному рівні роботу усіх сервісів компанії. За рахунок наявності повної інформації про всю топологію мережі між дата-центрами, системи балансування можуть миттєво коригувати виділення пропускної спроможності на різних ділянках усієї WAN мережі для активних сервісів, враховуючи поточний стан мережевих сегментів/каналів зв'язку та функціонуючого навантаження [5, с. 1050-1064]. Це дозволяє забезпечити високий рівень QoS як для кінцевих користувачів, так і для відповідних службових сервісів між дата-центрами всередині компанії.

Необхідно відмітити, що є ряд технологій та рішень (Hedera, DevoFlow, Mahout, MiceTrap), які полягають в розбитті (класифікації) потоків на великі „elephant-flows" та малі „mice-flows" та використання методів TE в процесі обслуговування більш пріоритетних потоків. Це було зроблено з метою розвантажити централізований SDN-контролер (OpenFlow Controler) та перекласти деякі функції обслуговування потоків на мережеве обладнання.

Розглянемо особливості роботи вище приведених технологій. Відповідно до концепції Hedera (рис. 2) розпізнавання великих та малих потоків (elephant/mice flows) відбувається на граничних комутаторах (edge-switches) до яких безпосередньо підключені клієнти (або сервери) та здійснюється розподіл потоків для подальшого обслуговування. А саме, або передати обслуговування потоку на контролер, або самостійно виконати передачу потоку (пакету). Тобто здійснюється певна модифікація моделі OpenFlow за рахунок повернення часткової відповідальності за прийняття рішення передачі на комутатори.

Принцип розпізнавання полягає у встановленні так званого граничного (порогового) значення (threshold), яке по замовчуванню становить 10% від максимальної бітової швидкості каналу зв'язку з сервером (наприклад, якщо канал підтримує 1 Гбіт/с, то threshold буде 100 Мбіт/с). Якщо величина потоку буде більше порогового значення, то потік маркуватиметься як великий „elephant-flows" та буде передано комутатором на централізований SDN-контролер. В свою чергу контролер, прорахує оптимальні (найменш навантажені та найкоротші) шляхи передачі для даного потоку, після чого контролер також надішле правила передачі для даного потоку на всі підпорядковані комутатори, які входять до визначеного шляху передачі. Потоки, які не перевищили threshold, не маркуються, але вважаються малими „mice-flows" і передаються без участі контролера. Це означає, що комутатори самостійно приймають

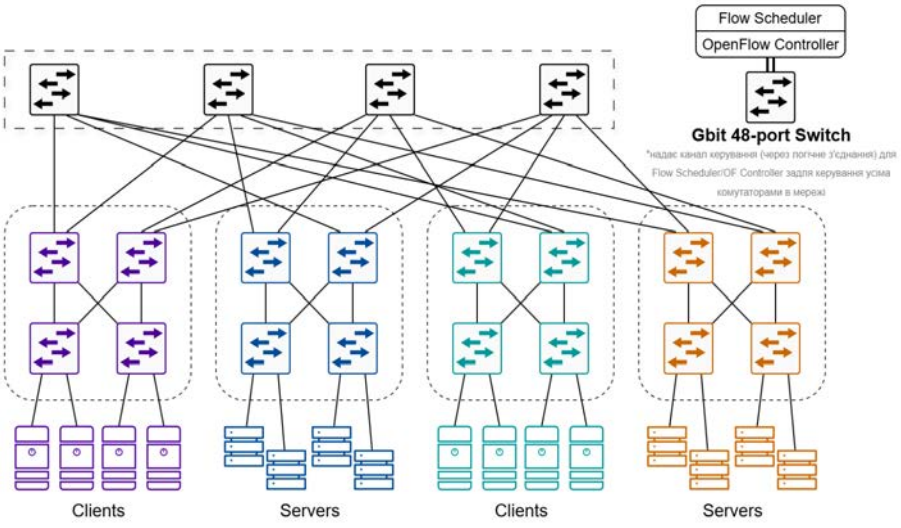

Рис. 2. Архітектура Hedera мережі дата-центру

рішення щодо маршрутизації трафіку, орієнтуючись на "вартість" шляхів передачі. Якщо існують шляхи з однаковою вартістю (equal-cost paths), трафік рівномірно розподіляється між ними.

Однак, в підходах з методами балансування навантаження, таких як DevoFlow та Hedera з'ясували, що виявлення та подальше маркування потоків досягають високих накладних витрат ресурсів, які споживають суттєву частину апаратних та мережевих ресурсів [6].

Тому в подальшому було розроблено підхід Mahout, який став своєрідний подальшим напрямком вирішення проблем розвантаження SDN-контролеру та елементів інфраструктури щодо класифікації потоків за їх об'ємом.

У Mahout виявлення великих потоків „elephant-flows" відбувається за рахунок модифікації кінцевих хостів (end-hosts). А саме, в роботі end-hosts впроваджується проміжний рівень (shim layer), який інтегрований в операційну систему для класифікації потоків. Цей проміжний рівень стежить за буфером TCP-сокета та позначає потоки великими, коли відбувається перевищення порогу швидкості в зазначеному буфері. Таким чином, підхід Mahout значно зменшує навантаження на комутатори та SDN контролер, тому що: не потрібно зберігати інформацію про всі потоки у TCAM; немає необхідності запитувати контролер про статус потоків. Це призводить до того, що в цілому зменшується навантаження на мережу, оскільки виявлення потоків виконується на рівні кінцевих пристроїв (хостів). Mahout використовує механізм внутрішньо-смугової сигналізації для позначення потоків як „elephant-flows" та інформування контролера про такі потоки. Для обчислення оптимальних шляхів контролер збирає статистику про „elephant-flows" та інформацію про завантаження каналів з комутаторів, аби вибрати та призначити найменш перевантажений шлях. Mahout використовує ECMP для маршрутизації звичайного трафіку. Цей метод може виявляти elephant-flows швидше, з меншими витратами на апаратні ресурси, ніж інші методи. Але він вимагає модифікації кінцевих хостів.

Також досить цікавим підходом являється MiceTrap, який орієнтований на оптимізацію передачі та балансування „mice-flows" у мережах дата-центрів. Це пов'язано з тим що, „mice-flows" іноді можуть становити до 90% усіх потоків дата-центру, що значно впливатиме на продуктивність всієї мережі дата-центру. В такому випадку застосування рішень ТЕ лише для „elephant-flows" 10% від усіх потоків дата-центру просто не є доцільним [7, с. 683-688].

MiceTrap забезпечує масштабованість шляхом агрегації „mice-flows" та використовує програмно-конфігурований алгоритм зваженої маршрутизації, що дозволяє динамічно і під поточне навантаження в мережі балансувати навантаження для „mice-flows".

Основні компоненти мережі на базі MiceTrap зображено на рис. 3.

На рисунку переставному вище видно, що MiceTrap активно взаємодіє з усіма компонентами SDN мережі, та налаштовує на них механізми для більш збалансованої роботи, а саме:

– на кінцевих серверах (хостах) розгорнуто модулі виявлення „elephant-flows";

– на OpenFlow-комутаторах розгорнуто модуль агрегації малих потоків на базі стандартних технологій OpenFlow (таблиці потоків, групові таблиці потоків);

– на MiceTrap SDN контролері розгорнуто модулі управління які використовуються для: побудови абстрактної структуру (топології) всієї мережі; збору статистики по мережі; розрахунку "вартості" усіх шляхів передачі в мережі; розрахунку шляхів для потоків „mice-flows".

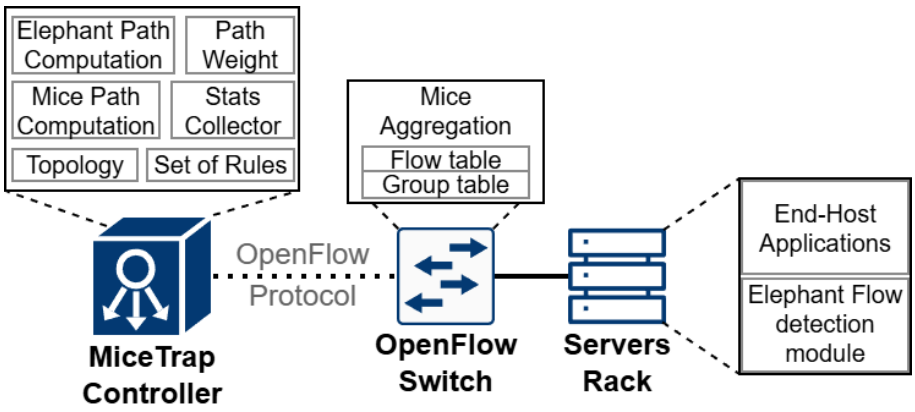

Рис. 3. Архітектура MiceTrap у межах мережі дата-центру

Хоча MiceTrap займається по більшій частині передачею „mice-flows", йому також необхідний алгоритм розпізнавання потоків та окремий механізм для обслуговування „elephant-flows". Тому, для виявлення та маркування великих потоків, MiceTrap (також як у Mahout) на кінцевому хості (сервері) використовує інтегрований shim layer на рівні ядра операційної системи. Shim layer контролює буфери TCP-сокетів, ідентифікує та позначає потік як "elephant-flow", коли кількість байт у буфері перевищує заздалегідь визначений поріг швидкості за певний проміжок часу.

Після виявлення "elephant-flow" визначається по замовчуванню режим передачі наступних ідентичних пакетів потоку (наприклад, з використанням OpenFlow записів в таблицях потоків на комутаторах зі спеціальними символами та багатошляхової маршрутизації ECMP).

Оскільки MiceTrap наголошує на тому, що більша частина трафіку мережі дата-центру може складатися з малих потоків, тому й було застосовано модуль Mice Aggregation (на комутаторі), котрий об'єднує вхідні малі потоки (усі потоки, які не були промарковані як великі) за якоюсь спільною ознакою (наприклад, конкретною IP-адресою отримувача або стійкою призначення).

Таким чином кількість правил, необхідних для передачі малих потоків зі спільною ознакою, суттєво зменшується. Це все фактично досягається за рахунок використання лише функцій, що надаються стандартом OF (OpenFlow). Під цим розуміється: використання групових таблиць (group tables) у поєднанні із таблицями потоків (flow tables) для підтримки багатошляхової маршрутизації (multi-path routing).

Багатошляхова маршрутизація реалізується за рахунок призначення потоку до певної групи (за певною ознакою). Кожна група складається з "бакетів" (group action buckets), причому кожен бакет містить набір дій, які застосовуються до відповідних потоків, тобто кожен бакет відповідає конкретному маршруту, яким потік може дістатися до місця призначення. А також, кожен бакет має власний параметр ваги (weight field), що визначає його частку в загальному трафіку, що обробляється групою.

Необхідно відмітити, що багатошляхова маршрутизації на базі ECMP розподіляє трафік рівно навпіл і шукає лише можливість виконати такий розподіл (шукає на кожному комутаторі декілька рівноцінних шляхів доставки до отримувача). В ECMP враховується також вага кожного маршруту передачі, така вага зазвичай розраховується з урахуванням кількості стрибків (хопів) між комутаторами та заданої пропускної здатності каналів. Тобто поточний стан мережі (навантаження каналів, затримки) не враховано в даному концепті маршрутизації, через це можуть з'являтися сегменти мережі, де буде перевантаження каналів.

Відповідно у підході MiceTrap на базі ECMP впровадили власний алгоритм зваженої маршрутизації, який дозволяє у режимі реального часу розподіляти навантаження між шляхами передачі, враховуючи поточне завантаження каналів зв'язку. Це дає змогу при передачі mice flows більш оптимально використовувати ресурси мережі дата-центру, за рахунок визначення функціонуючого навантаження та стану мережі.

Також MiceTrap, пропонує встановлювати правила що відповідають спільній адресі призначення, оскільки модель трафіку "багато до одного" досить поширена в дата-центрах. Наприклад, якщо 1000 малих потоків надходять на комутатор Top of Rack (ToR), які потім будуть доставлятися на одну й ту саму адресу призначення, тоді контролеру варто встановити лише одне правило, що відповідає адресі призначенню, замість створення окремого правила для кожного потоку.

Ще одним оригінальним рішенням для балансування навантаження є Multipath Software-Defined Networking Traffic Engineering (MSDN-TE) [8, с. 630–639], який також використовує багатошляхову маршрутизацію для оптимального використання мережевих ресурсів та підвищення продуктивності мережі.

Якщо розглядати принцип роботи MSDN-TE, то основна архітектура підходу зображена на рис. 4.

Спочатку відбувається збір статистики та передача її на контролер для цього призначений модуль моніторингу MONFUN (рис. 4). Даний модуль отримує необхідні статистичні дані такі як топологія мережі, статистика потоків, використання каналів, а також коефіцієнт втрати пакетів (Packet Loss Ratio) та час затримки пакетів (Packet Delay). MONFUN надає інформацію про потоки та їх шляхи передачі. Також збирається інформація скільки трафіку генерує кожен хост у мережі. При чому, частота оновлення метрик шляхів передачі та оновлення топології мережі знаходиться в діапазоні 10-15 секунд і залежить від здатності контролера збирати цю інформацію з комутаторів.

В результаті обробки статистичних даних отримуються наступні синтетичні параметри мережі: Link Load – середнє значення навантаження кожного каналу за певний період часу; Average Network Load – середнє значення навантаження

всіх каналів мережі; Total Network Dropped – кількість всіх втрачених пакетів в мережі.

Оброблені дані надходять на наступний модуль TE Algorithm, який прораховує навантаження в мережі, перебудовує маршрути в мережі при використанні алгоритму Еппштейна, створює нові правила про призначення потоків на визначені шляхи передачі. Усі оновлені (або нові) правила обробки потоків передаються на модуль керування ACTFUN.

ACTFUN залучається задля внесення змін (політик передачі) на площині передачі (data plane, оновлення таблиць потоків на комутаторах). Контролер може динамічно налаштовувати діючі параметри (path_ID та flow_ID) для визначення потокам оптимальних шляхів передачі.

Якщо більш детально, то MSDN-TE використовує підхід SDN із залученням алгоритму Еппштейну. Підхід базується на використанні декількох шляхів для пересилання потоків з урахуванням поточного значення навантаження шляхах в момент передачі. При прийнятті рішення про пересилання потоку, здійснюється обчислення k-шляхів найкоротших для пересилання потоків між будь-якою парою Source-Destination (S-D), а потім здійснюється вибір найменш завантаженого шляху для передачі потоку. Тобто, алгоритм Еппштейна собою являє метод оптимізації розподілу мережевих ресурсів, за яким трафік розподілятиметься між доступними шляхами таким чином, аби мінімізувати затримки та втрати пакетів.

Працює алгоритм Еппштейна наступним чином:

– спочатку відбувається збір мережевих метрик, де визначаються усі можливі шляхи передачі між джерелом і призначенням (S-D), а також збираються дані про мережеві параметри: затримки, поточне завантаження каналів зв'язку;

– після чого відбувається обчислення ваги кожного шляху з урахуванням багатьох мережевих параметрів за допомогою наступного виразу:

$$W(P)=\alpha\times delay+\beta\times loss+\gamma\times utilization, \quad (1)$$

де: delay – затримка; loss – втрата пакетів; utilization – завантаження каналу; коефіцієнти α, β, γ і визначають вагу (weigh) кожного параметру з аналітичного виразу (1);

– далі йде вибір найбільш оптимального шляху передачі потоку за найменшим значенням W(P).

– останнім кроком є процес динамічного коригування, де при зміні мережевого стану (збільшенні завантаженості, відмові каналів зв'язку) алгоритм перераховує параметр ваги шляху (W(P)) та адаптує маршрутизацію під поточний стан мережі.

Використання MSDN-TE в поєднані з алгоритмом Еппштейна має суттєві переваги в порівнянні за протоколом багатошляхової маршрутизації ECMP, однак у випадку рівнозначних шляхів передачі не дозволяє одночасно їх використовувати при перенавантаженні.

Формулювання цілей статті. Тому в дані статті пропонується модифікація підходу MSDN-TE, а саме при передачі трафіку реального часу між Source-Destination, що матимуть рівнозначні шляхи одночасно використовувати декілька шляхів передачі з урахуванням показників якості обслуговування QoS (Delay, Packet Loss, Jitter).

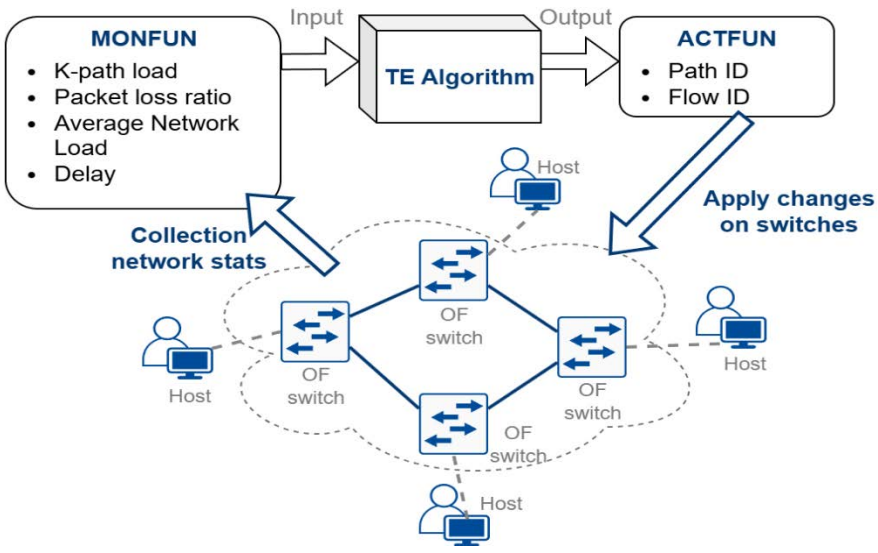

Рис. 4. Архітектура MSDN-TE

Тобто, на етапі динамічного розподілу (балансування) навантаження, коли пропускної основного шляху спроможності не вистачає щоб забезпечити задані показники QoS, то надлишкові потоки трафіку реального часу передавати по альтернативному рівноцінному шляху передачі.

Виклад основного матеріалу. З метою підтвердження запропонованих рішень та проведення експериментів був розгорнути віртуальний сегмент SDN мережі дата-центру, при використанні розподіленого контролеру ONOS та Mininet для емуляції необхідної топології мережі.

Візуалізація топології SDN мережі на контролері ONOS, яка використовувалась у експериментах, зображено на рис 5.

Для експерименту було використано кластер з трьох ONOS контролерів, які в подальшому (при запуску мережі) взаємодіють через Mininet з віртуальними комутаторами OpenFlow virtual Switch (OvS). Відповідно до комутаторів під'єднані хости (сервери), які генерують UDP-потоки за допомогою утиліти iPerf. Топологія мережі з її елементами, бітовою швидкістю каналів, прописувалася у вигляді програми на мові програмування Python, яка запускається через платформу-емулятор Mininet.

Комутатори (OvS) на схемі віртуального макету підсвічуються різними кольорами, бо вони підпорядковуються різним контролерам з кластеру, що дає змогу рівномірно розподілити навантаження між контролерами без втрати продуктивності. Оскільки контролери в кластері синхронізуються між собою за допомогою Atomix Map, це дає змогу на кожному контролері мати однаково оновлені метрики про стан мережі рис. 6.

Вхідними даними для побудови топології (рис. 5) перед проведенням експериментів є:

10 OpenFlow virtual switches (віртуальні комутатори з підтримкою протоколу OpenFlow);

8 віртуальних хостів (які являють собою сервери у різних дата-центрах, кожен сервер має унікальну IP-адресу на кшталт 192.168.0.*);

3 ONOS-контролера (які об'єднані в кластер) рис. 6;

18 каналів зв'язку (links) в мережі між комутаторами з бітовою швидкістю 10 Мбіт/с кожен;

8 каналів зв'язку (links) між комутаторами та серверами (хостами) з бітовою швидкістю 10 Мбіт/с.

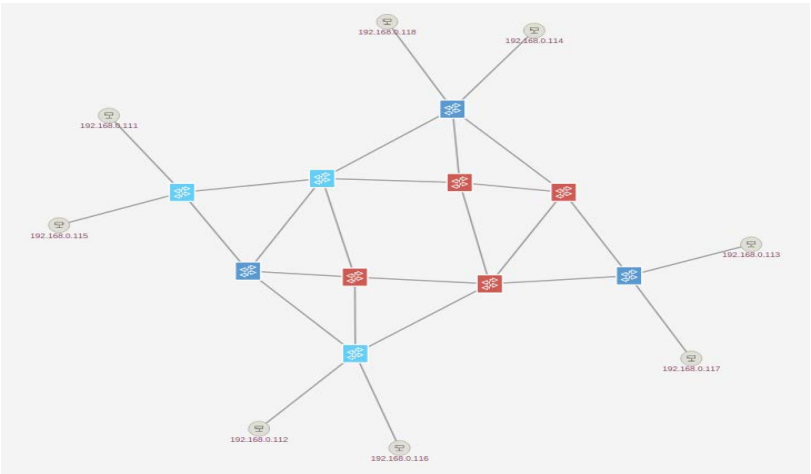

Рис. 5. Варіант топології SDN мережі дата-центрів

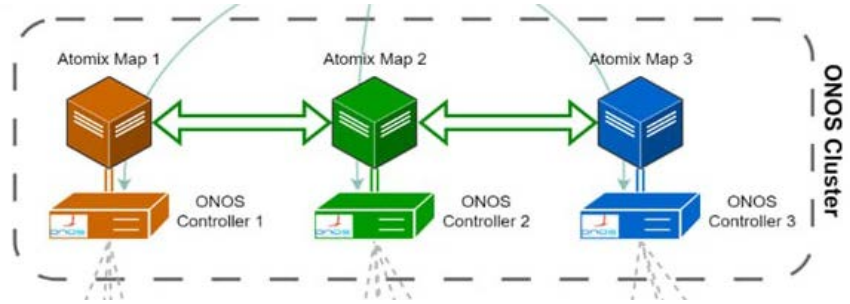

Рис. 6. Варіант архітектури ONOS кластеру

Для проведення експерименту були обрані найбільш віддалені сервери (згідно нижче приведеної топології SDN мережі на рис. 7), причому генерація UDP-потоків відбувалась у двох напрямках для емуляції роботи трафіка реального часу.

Результати моделювання для різних значень навантаження в кожному інформаційному напрямку (Information Direction, *ID*) та отримані значення мережевих параметрів зведені в таблицю 1. Під номером експерименту (табл. 1.) розуміється одночасна передача чотирьох UDP-сесій з однаковим значенням навантаження.

З таблиці 1 можемо побачити, що при сумарному навантаженні 8 Мбіт/с на два інформаційних напрямках, з'являються перші мінімальні втрати у мережі (2 втрачених на 142,861 доставлених пакетів). При подальшому збільшенні вхідного навантаження ситуація на мережі лише погіршується і відсоток втрат зростає. Так при навантаженні 10 Мбіт/с середній відсоток втрат пакетів в середньому складає 3,1%, а при навантаженні 12 Мбіт/с, що перевищує пропускну спроможність каналів у визначеному шляху мережі, середній відсоток втрат досягає 18%, що являється неприпустимо великим відсотком для трафіка реального часу. Також варто помітити, що коли навантаження досягло граничного значення нашої пропускної спроможності шляху передачі (10 Мбіт/с), тоді середній показник RTT набуває значення 1,9188 мілісекунд та затримки передачі пакетів також збільшується, що в подальшому також призведе до недотримання вимог щодо QoS. При навантаженнях, які перевищують пропускну здатність каналів передачі, а саме: 11 та 12 Мбіт/с параметри затримки та RTT починають показувати занижені дані, бо мережа перевантажена і працює некоректно.

Більш того, при подальшому збільшенні сумарного вхідного навантаження (11, 12 Мбіт/с) відповідно до протоколів маршрутизації по замовчуванню спостерігались великі втрати пакетів (відмова лінії через перенавантаження, див. рис. 8) та автоматично здійснювалось переналаштування резервного маршруту. Однак, перенаправлення трафіку на резервний шлях передачі не дало суттєвого ефекту та покращення показників якості обслуговування (втрати пакетів мали практично такі ж значення).

Тому наступним кроком стало використання модифікованого методу балансування навантаження на основі алгоритму Епштейна, а саме при додаванні кожної наступної сесії передачі контролери враховували всі допустимі рівнозначні шляхи передачі та рівномірно розподіляли потоки трафіка між ними та одночасно використовували декілька шляхів передачі. Принцип роботи удосконаленого методу балансування показано на рис. 9.

Після необхідних налаштувань при використанні модифікованого методу балансування була проведена серія експериментів, результати зведені в таблицю 2.

Аналіз результатів тестування (таблиця 2), згідно запропонованих удосконалень, вказують

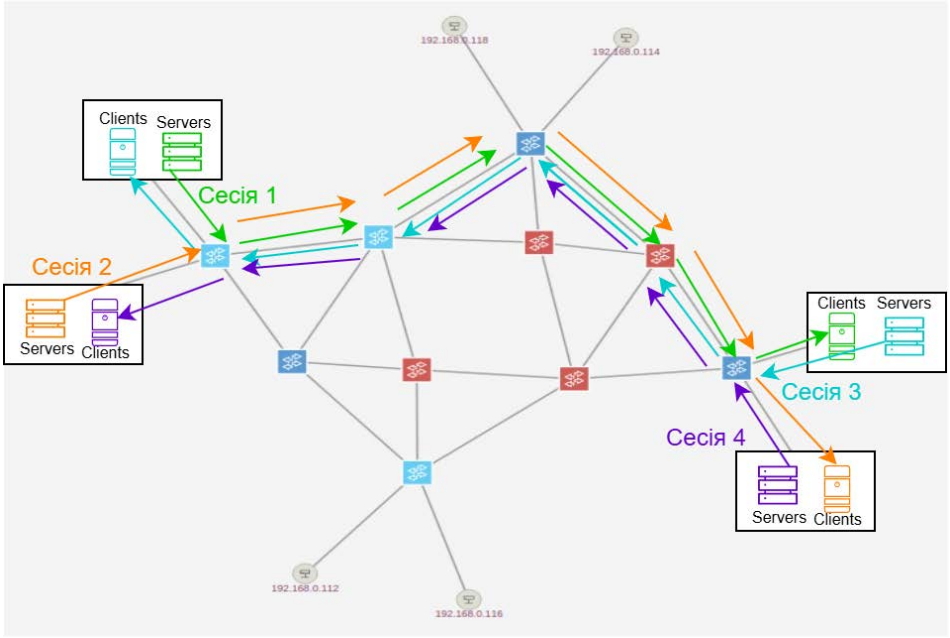

Рис. 7. Схема передачі трафіку до використання балансування навантаження

Таблиця 1

Статистичні дані після виконання першого експерименту

№ експер.	вхідне навантаження одного потоку, [Мбіт/с]	обслужене навантажен. за одну сесію, [Мбіт/с]	сумарне навантажен. в *ID*, [Мбіт/с]	середн. об'єм переданих даних за всі сесії, [Мбайт]	середн. затримка пакетів *Delay*, [мс]	середн. показник RTT за сесію, мс	середн. відсот. втрат пакетів Packet Loss, [%]
1	0,25	0,25	1	6,26	0,0341	0,0683	0,000
2	0,5	0,5	2	12,5	0,0310	0,0620	0,000
3	0,75	0,75	3	18,8	0,0116	0,0233	0,000
4	1	1	4	25	0,0185	0,0370	0,000
5	1,25	1,25	5	31,3	0,0198	0,0395	0,000
6	1,5	1,5	6	37,6	0,0128	0,0255	0,000
7	2	2	8	50,1	0,0816	0,1633	0,001
8	2,25	2,25	9	56,3	0,1525	0,3050	0,004
9	2,5	2,425	10	60,625	0,9594	1,9188	3,108
10	2,75	2,5075	11	63,125	0,8808	1,7615	8,275
11	3	2,45	12	61,6	0,7795	1,5590	18,075

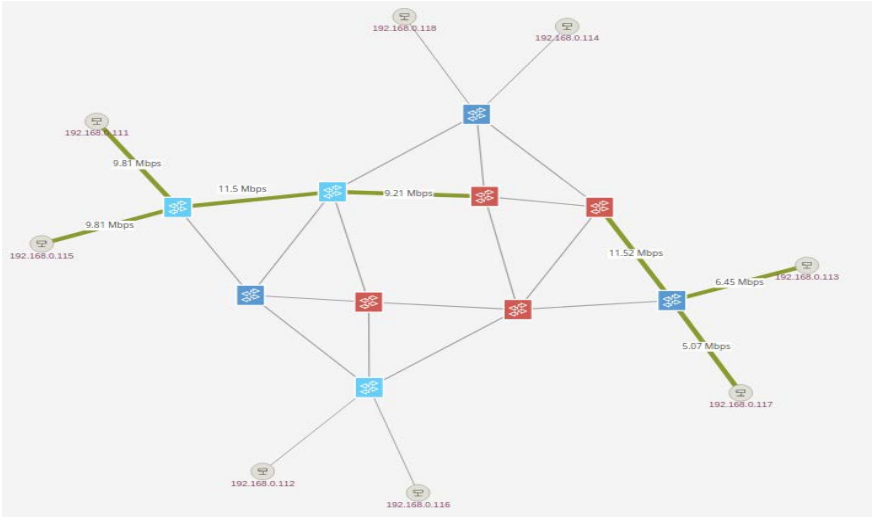

Рис. 8. Аварійна ситуація відмова лінії в SDN-мережі

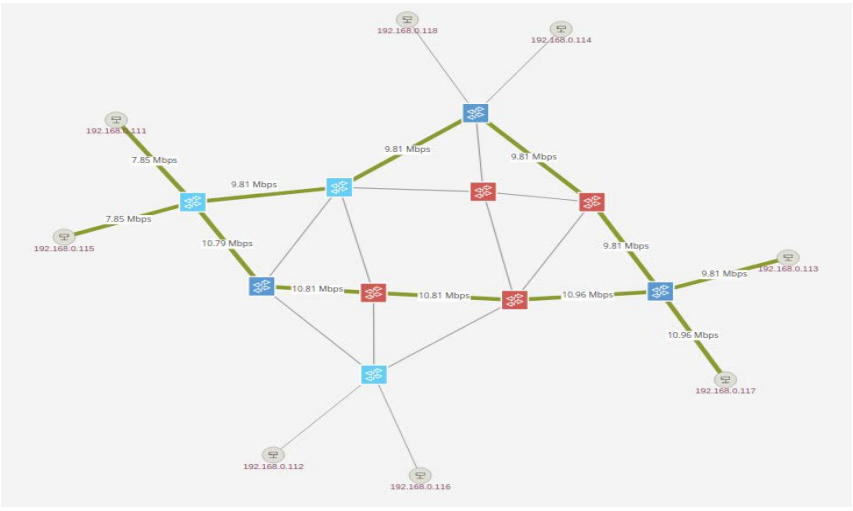

Рис. 9. Схема передачі трафіку при використанні балансування навантаження

Таблиця 2

Статистичні дані після виконання другого експерименту

№ експер.	вхідне навантаження одного потоку, [Мбіт/с]	обслужене навантажен. за одну сесію, [Мбіт/с]	сумарне навантажен. в *ID*, [Мбіт/с]	середн. об'єм переданих даних за всі сесії, [Мбайт]	середн. затримка пакетів *Delay,* [мс]	середн. показник RTT за сесію, мс	середн. відсот. втрат пакетів Packet Loss, [%]
1	2,25	2,25	9	56,3	0,0124	0,0248	0,001%
2	2,5	2,5	10	62,6	0,0118	0,0235	0,001%
3	2,75	2,75	11	68,9	0,0226	0,0453	0,000%
4	3	3	12	75,1	0,0080	0,0160	0,004%

на те, що за рахунок розподілу навантаження між рівнозначними шляхами передачі можна отримати допустимі значення показників QoS без зміни топології та збільшення бітової швидкості ліній передачі. Тобто при загальному навантаженні 10 Мбіт/с середній відсоток втрат пакетів складає лише 0,001% (в порівнянні з попередніми експериментами).

Графічне порівняння, на базі отриманих даних, об'єму переданих даних при першому (передача даних через 1 шлях) та другому експерименті (передача даних через 2 шляхи) зображено на рисунку 10.

З рисунку 10 видно, що під час першого експерименту, при досягненні пропускної здатності (10 Мбіт/с) кількість сумарно переданих даних починає знижуватися. В той час як під час другого експерименту об'єм переданих даних лишається на належному рівні, що свідчить про стабільність роботи мережі і відсутність втрат в каналах зв'язку.

Графічне порівняння, обслуженого навантаження за одну сесію при кожному експерименті зображено на рисунку 11.

З рисунку 11 видно, що під час першого експерименту, при досягненні пропускної здатності (10 Мбіт/с) реальне, або ж обслужене, навантаження, яке генерується при кожній сесії передачі (під час кожного експерименту виконувалася одночасна передача 4 сесій) почало занижуватися, через перевантаження в мережі. В той час як під час другого експерименту навантаження генерується в потрібному обсязі, без заниження, що вчергове свідчить про стабільність роботи мережі і відсутність втрат в каналах зв'язку. Таким чином ми вважаємо, що динамічне балансування одразу між декількома шляхами передачі являється ще одним ефективним рішенням для мереж дата центрів на рівні усіх розглянутих підходів та методів. А те, що балансування відбувається і обраховується з кожним новим надходячим потоком даних, тим самим спираючись на відсоток завантаження шляхів передачі у даний момент часу, дозволяє підвищити показників QoS у системі, та нівелювати утворення ситуацій "bottleneck" у мережі.

Висновки. Аналіз різних технологій та архітектур дата-центрів вказує на широке застосу-

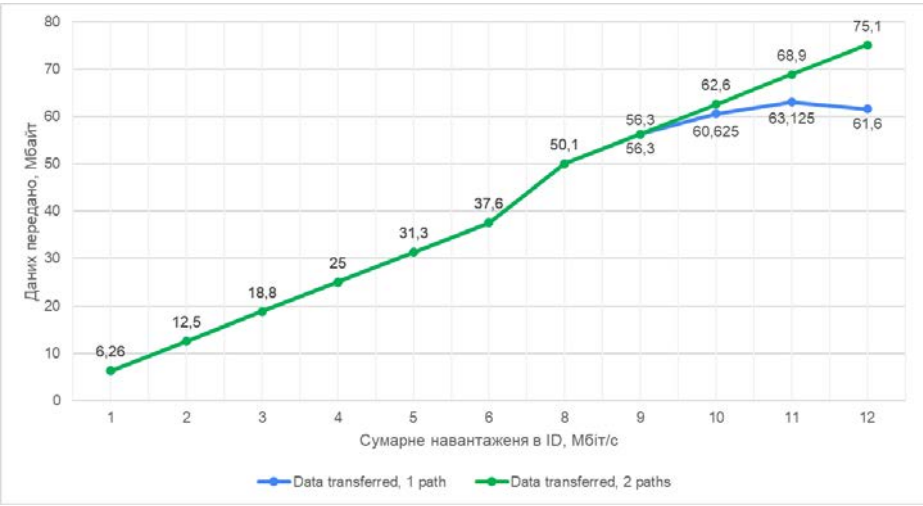

Рис. 10. Залежність сумарного навантаження від переданих даних двох експериментів

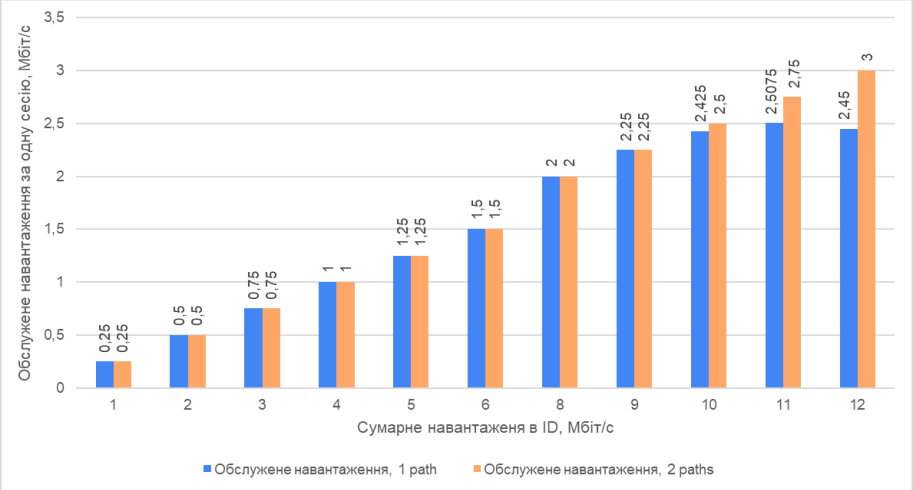

Рис. 11. Залежність сумарного навантаження від обслуженого навантаження двох експериментів

вання підходів SDN-мереж та методів „інжирінгу трафіка" для ефективного використання наявних ресурсів та забезпечення показників QoS в залежності від типу сервісів.

Розбиття трафіка на великі „elephant-flows" та малі „mice-flows" потоки (Hedera, DevoFlow, Mahout, MiceTrap) дозволяє визначити різні сценарії обслуговування та протоколи балансування навантаження з багатошляховою маршрутизацією.

В свою, чергу, метод Multipath Software-Defined Networking Traffic Engineering з використанням алгоритму Епштейна, дозволяє визначити більш оптимальний шлях передачі для кожного вхідного потоку з урахуванням більшої кількості мережевих параметрів ніж існуючий протокол багатошляхової передачі (ECMP).

В роботі представлено удосконалений метод балансування на основі алгоритму Епштейна, що дозволяє одночасно використовувати декілька шляхів передачі з дотриманням показників якості обслуговування QoS при передачі трафіка реального часу.

Список літератури:

1. Data Center Traffic Scheduling Strategy for Minimization Congestion and Quality of Service Guaranteeing / C. Wang et al. Computers, Materials & Continua. 2023. Vol. 75, no. 2. P. 4377–4393. URL: https://doi.org/10.32604/cmc.2023.037625.

2. Mathematical Description of Control Problems in SDN Networks / Romanov O. et al. International Conference on Applied Innovations in IT (ICAIIT). P. 33–40. URL: http://dx.doi.org/10.25673/36582.

3. Curtis A. R. et al. DevoFlow. ACM SIGCOMM Computer Communication Review. 2011. Vol. 41, no. 4. P. 254–265. URL: https://doi.org/10.1145/2043164.2018466.

4. Curtis A. R., Kim W., Yalagandula P. Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection. IEEE INFOCOM 2011 – IEEE Conference on Computer Communications, Shanghai, China. 2011. P. 1629-1637 URL: https://doi.org/10.1109/infcom.2011.5934956.

5. Dynamic Parallel Flow Algorithms with Centralized Scheduling for Load Balancing Improvement in Cloud Data Center Networks / W.-K. Chung et al. IEEE Transactions on Cloud Computing. 2021. P. 1050-1064. URL: https://doi.org/10.1109/tcc.2021.3129768.

6. Peterson L., Cascone C., Davie B. Software-Defined Networks: A Systems Approach. Systems Approach LLC, 2021.

7. Romanov O., Mankivskyi V. Optimal Traffic Distribution Based on the Sectoral Model of Loading Network Elements. 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine. 2019. P. 683-688 URL: https://doi.org/10.1109/picst47496.2019.9061296.

8. MSDN-TE: Multipath Based Traffic Engineering for SDN / K. T. Dinh et al. Intelligent Information and Database Systems. Berlin, Heidelberg, 2016. P. 630–639. URL: https://doi.org/10.1007/978-3-662-49390-8_61.

Romanov O.I., Nesterenko M.M., Marinov A.I. MODIFIED MULTIPATH LOAD BALANCING METHOD IN SDN OF DATA CENTERS

The article is dedicated to the study and enhancement of load balancing methods in software-defined networks (SDN) of data centers. The relevance of this problem is driven by the need for efficient utilization of available network resources and the assurance of high Quality of Service (QoS) levels in the dynamic environment of cutting-edge data centers. The ongoing evolution of SDN architectures underlines the widespread adoption of this paradigm for the deployment of data center network infrastructures. The solution to this challenge lies in the application of traffic engineering (TE) techniques, which considerably can improve network performance and service quality through multipath routing combined with traffic classification and prioritization. The use of TE is also aimed at reducing operational costs and preventing the emergence of congested segments within the network. The paper provides an analysis and comparison of existing technologies and engineering solutions for network resource optimization, including B4 (Google), Hedera, DevoFlow, Mahout, MiceTrap, and MSDN-TE. Particular attention is given to approaches such as Hedera, DevoFlow, Mahout, and MiceTrap, what rely on classifying flows into large "elephant flows" and small "mice flows" to offload the centralized SDN controller and increase service efficiency. In addition, MSDN-TE is considered, which leverages multipath routing in combination with the Eppstein algorithm. The latter computes the least-loaded path by assigning weights to each transmission path based on specific network metrics, thereby minimizing delay and packet loss. In this work propesed a modified load balancing method based on the Eppstein algorithm. The improvement enables simultaneous utilization of multiple equivalent transmission paths for real-time traffic between the source and the destination. This approach is particularly relevant when the bandwidth of the primary path is insufficient to meet required QoS parameters such as Delay, Packet Loss, and Jitter. To validate the proposed solution was deployed a virtual SDN data center segment with the help of ONOS and Mininet solutions. The results demonstrate that dynamic load balancing, based on the real-time utilization of transmission paths, represents an effective approach to improving QoS, mitigating the formation of bottlenecks, and ensuring reliable traffic delivery in SDN-based data center networks.

Key words: *software-defined networking (SDN), data centers, quality of service (QoS), traffic engineering (TE), traffic transmission, network load balancing, OpenFlow, multi-path routing.*

Дата надходження статті: 21.09.2025
Дата прийняття статті: 08.10.2025
Опубліковано: 16.12.2025