УДК 004.7:621.391.1
DOI: https://doi.org/10.17721/AIT.2025.1.05

Микола НЕСТЕРЕНКО, д-р техн. наук, доц.
ORCID ID: 0000-0003-0812-2793
e-mail: mykola.nesterenko@viti.edu.ua
Військовий інститут телекомунікацій та інформатизації імені Героїв Крут, Київ, Україна

Антон МАРІНОВ, асп.
ORCID ID: 0000-0001-8717-2734
e-mail: marinov.anton@lll.kpi.ua
Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського", Київ, Україна

УПРАВЛІННЯ SDN-МЕРЕЖЕЮ ДАТА-ЦЕНТРІВ НА БАЗІ МОДИФІКОВАНОГО ПІДХОДУ DEVOFLOW

Вступ. *Досліджено особливості реалізації рівня управління SDN-мереж під час обслуговування різних за обсягом потоків дата-центрів на основі технології Hedera, протоколу OpenFlow та підходу DevoFlow. Розподіл функцій оброблення, збору статистичних даних та періодичності оновлення статистичних даних впливають як на обсяг службової інформації, так і на ефективність та оперативність управління щодо розподілу вхідного трафіка для балансування навантаження. Метою роботи є модифікація підходу Hedera за рахунок перерозподілу функцій класифікації потоків та адаптивного управління залежно від обсягу потока на рівні управління, а також розроблення алгоритму визначення допустимого часу надсилання статистичних даних від комутаторів на SDN-контролер.*

Методи. *Використано метод системного аналізу та декомпозиції для дослідження складних систем, методи збору даних про стан мережі, а також евристичні правила для визначення обсягу службового трафіка.*

Результати. *У результаті проведеного аналізу особливостей побудови і роботи таких технологій, як Hedera та DevoFlow, запропоновано модифіковану архітектуру побудови рівня управління та визначено основні функції програмних модулів, які дають змогу підвищити ефективність роботи SDN-контролерів. А саме, завдяки зміні порядку збору й оброблення службової інформації на обладнанні SDN-мережі дата-центрів і поверненню до централізованого типу управління.*

Визначено підходи щодо класифікації вхідних потоків та їх розбиття на малі "mice-flows", середні "medium-flows" і великі потоки "elephant-flows", що дозволяє в подальшому використовувати багатошляхову маршрутизацію для ефективнішого використання ресурсів мережі. Також у цій статті запропоновано алгоритм визначення допустимого часу надсилання статистики від комутаторів (метод push-based), який враховує динаміку зміни навантаження в лініях у мережі, кількість активних сесій і базується на евристичному правилі обмеження обсягу службового трафіка.

Висновки. *Розроблено модифіковану архітектуру рівня управління на базі підходу DevoFlow та визначено основні функції програмних модулів SDN-контролера мережі дата-центрів. Розроблено аналітичні залежності для визначення допустимого часу надсилання статистичних даних від комутаторів на SDN-контролер, які враховують поточний стан завантаженості мережі. А також проведено ряд експериментів для обґрунтування обраного методу отримання статистичних даних і підтвердження достовірності запропонованого алгоритму допустимого часу надсилання статистичних даних від комутаторів на SDN-контролер.*

Ключові слова: *Software-defined Networking, дата-центр, OpenFlow, Hedera, DevoFlow, elephant-flows, mice-flows, SDN-контролер, OpenFlow Switch, таблиця потоків.*

Вступ

Сучасний розвиток мереж SDN вказує на широке застосування цієї технології для розгортання мережевої інфраструктури розподілених центрів обробки даних. Причому динамічна природа додатків і сервісів, а також різнорідність технологій транспортних мереж вимагають упровадження додаткових механізмів, що дозволяють ефективно використовувати наявні мережеві ресурси та забезпечувати необхідні показники якості обслуговування QoS.

Зазвичай основною перевагою SDN-мережі є централізований підхід щодо управління мережевими ресурсами у процесі розподілу різнорідних потоків трафіка за рахунок агрегації даних про поточний стан обладнання з урахуванням усієї топології мережі. Тобто, при розрахунку шляхів передачі (маршрутів) SDN-контролер має повнішу статистику про стан усіх елементів мережі (Globa et al., 2019, с. 76–100; Romanov, & Mankivskyi, 2019, с. 683–688).

Однак під час обслуговування потоків різних обсягів і часу тривалості сесії, виникають складності щодо їхнього розподілу між доступними шляхами передачі (оптимальності формування таблиць потоків), оскільки це напряму вливає на ефективність використання мережевого ресурсу та завантаження ліній передачі.

В роботах (Costa et al, 2021, vol. 147; Yu et al., 2018, с. 251–258), розглянуто такі рішення, як Hedera та Devolved OpenFlow (DevoFlow), які націлені на збільшення пропускної спроможності мережі дата-центру за рахунок класифікації (розбиття) вхідного трафіка на великі "elephant-flows" та малі "mice-flows" потоки з метою використання протоколів багатошляхової маршрутизації для балансування навантаження великих потоків. Але Hedera обмежується лише внутрішньою мережею дата-центру й аналізує обсяг потоків лише на граничних комутаторах (edge-switches). Особливість підходу DevoFlow (Costa et al., 2021, vol. 147) полягає також у розділенні потоків на "elephant-flows" і "mice-flows" та обробці великих потоків із залученням контролера, й обробці малих потоків на комутаторах без додаткового звернення до SDN-контролера.

Однак застосування моделі DevoFlow хоч і дає змогу зменшити час для визначення маршруту передачі та зменшує навантаження на сам контролер, але цей фактор може призвести до максимального завантаження окремих маршрутів (ліній передачі) і як наслідок – до перевантаження окремих сегментів мережі. Це пояснюється тим, що часткове перенесення рівня управління на комутатори (OpenFlow Switch) приводить до втрати повної картини щодо топології та стану функціонування мережі. Тому в цій статті пропонується модифікація підходу DevoFlow з перерозподілом функцій класифікації потоків та адаптивного управління залежності від обсягу потока саме на SDN-контролері. Розглянемо детальніше наведені вище технології та особливості їхньої роботи.

ISSN 2788-6603

Постановка задачі. Почнемо з технології Hedera, яку створено для ефективного використання пропускної спроможності в центрі обробки даних. Реалізація Hedera полягає в періодичному збиранні статистики (кожні 5 секунд) із граничних комутаторів для виявлення "elephant-flows". Спочатку комутатори відправляють новий потік, використовуючи свої стандартні правила зіставлення потоків, по одному зі шляхів з однаковою вартістю. Якщо обсяг потоку досягає порогового значення, тоді він маркується як elephant-flow. Поріг, за замовчуванням, становить 10 % потужності мережевого інтерфейсу (NIC). У цей момент центральний планувальник Hedera використовує глобальну інформацію про стан мережі і розраховує найкращий шлях для потоку. Причому, для обчислення шляху передачі визначається його доступна пропускна спроможність, щоб він міг вмістити потік.

Зазначимо, що DevoFlow являє собою модифікацію моделі OpenFlow, в якій відбувається деякий розрив між централізованим керуванням і централізованою видимістю стану всієї мережі. А саме, DevoFlow використовує агресивне використання існуючих OpenFlow-правил на комутаторах для "mice-flows", щоб зменшити обмін службовими повідомленнями між контролером і комутаторами. Отже, за відповідних умов, комутатори можуть приймати рішення про маршрутизацію малих за обсягом потоків локально, в той час як контролер здійснює загальне управління мережею і маршрутизує великі потоки.

Для розуміння запропонованих удосконалень та логіки роботи SDN-мережі зазвичай потрібно приділити увагу протоколу OpenFlow. Відповідно, OpenFlow є одним із ключових протоколів у концепції програмно-конфігурованих мереж (SDN), який дозволяє розділити площину керування та площину пересилання трафіка (Wang et al., 2023, с. 4377–4393). Основна ідея OpenFlow полягає в тому, що комутатори отримують інструкції від централізованого контролера, який визначає маршрутизацію пакетів на основі глобальної інформації про мережу (рис. 1). Це забезпечує гнучке управління потоками, спрощує налаштування політик і дає змогу централізовано контролювати трафік, що є критично важливим для сучасних дата-центрів і мереж із високими вимогами до якості обслуговування (QoS).

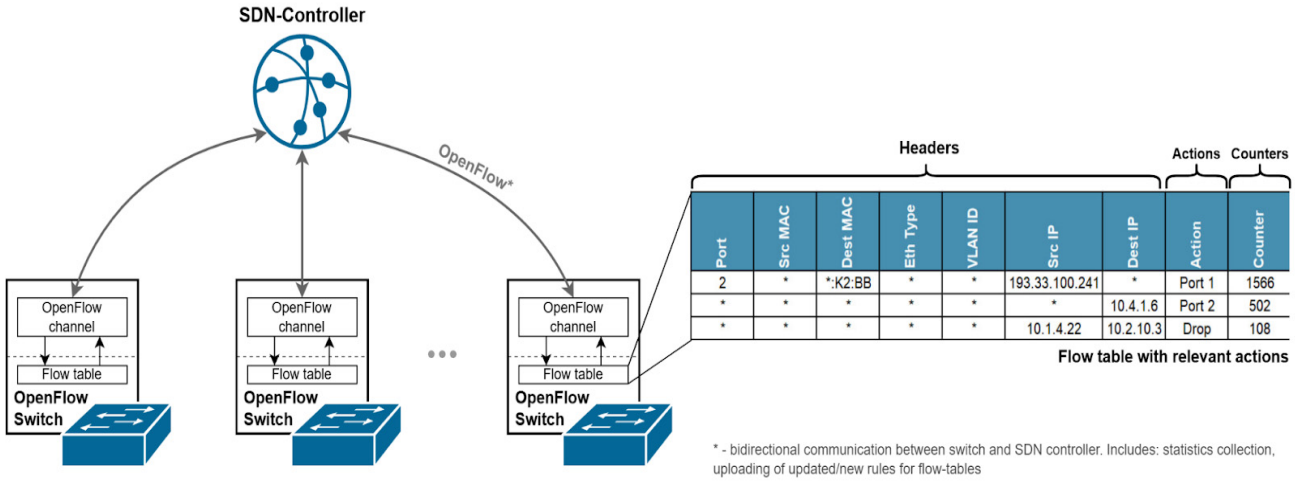

Рис. 1. Модель роботи протоколу OpenFlow і структура таблиці потоків

Перевагами централізованого контролю, базою для якого став підхід OpenFlow, є:

- глобальне управління політиками без складної конфігурації на рівні комутаторів, а точніше, навіть відсутність необхідності налаштовувати комутатор;
- можливість здійснювати оптимальну маршрутизацію потоків, особливо в мережах із гарантіями якості обслуговування (QoS) (Romanov et al., 2021, с. 33–40; Shirmarz, & Ghaffari, 2020, с. 7545–7593);
- гнучке управління потоками та безпека;
- можливість глобального бачення мережі, де контролер має "перед очима" весь процес передачі в мережі, що дозволяє завчасно реагувати на перевантаження, змінювати маршрути в реальному часі та, завдяки цьому, ефективніше використовувати мережеві ресурси. (насправді майже кожен SDN-контролер може задіяти процес динамічного переадресування потоків, на базі використання завжди оновлених даних про навантаження на мережевих з'єднаннях).

За використання традиційного OpenFlow, потоки можуть бути розділені на:

- безпекові (security-sensitive flows), які обробляються централізовано для підтримки безпеки;
- значущі потоки (significant flows), що керуються централізовано для покращення QoS та уникнення перевантажень у мережі;
- звичайні потоки (normal flows), які можуть бути оброблені локально комутаторами на базі заздалегідь прописаних правил, чи історично збереженими політиками передачі (ці дані очищуються досить швидко).

Однак, маючи досить суттєві переваги, що роблять мережу не дуже великих масштабів досить робочою і ефективною у використанні, централізована модель OpenFlow має суттєві недоліки, що серйозно обмежують її масштабованість (Tennakoon et al., 2018, с. 1043–1050).

По-перше, задіяння контролера для кожного нового сеансу передачі серйозно навантажує сам контролер. Наприклад, один NOX-контролер може обробляти до 30 000 нових потоків за секунду, але цього може бути недостатньо для великих за обсягом трафіка дата-центрів, із сотнями, а то й тисячами користувачів (Rout, Patra, & Sahoo, 2017, с. 543–551; Romanov, Nesterenko, & Mankivskyi, 2021, с. 159–182). Якщо взяти кластер, який має в підпорядкуванні 1500 серверів, то в такій мережі може надходити навіть до 100 000 нових потоків щосекунди, що вимагає, як мінімум, використання кількох контролерів для ефективної роботи мережі в цілому.

По-друге, маємо високі витрати на мережеву взаємодію. Використання контролера для налаштування потоків збільшує мережеві затримки в мережі. Наприклад, для *N* комутаторів потрібно створити та передати вдвічі більше (*2N*) записів у таблиці потоків, а також передати додаткові контрольні пакети, що, знову ж таки, створює додаткове службове навантаження.

По-третє, виникають обмеження продуктивності комутаторів, оскільки фізичні комутатори мають суттєві обмеження по ресурсах, які вводять в дію для оброблення потоків. Наприклад, традиційні комутатори можуть обробляти до 146 нових потоків за секунду, що може стати тим самим проблемним місцем (вузьким місцем / bottleneck) у великих за масштабом мережах.

Також варто зазначити, що OpenFlow-правила зберігаються в TCAM-пам'яті (Ternary Content Addressable Memory), яка є хоч дуже швидкою, але дорогою апаратною пам'яттю та використовує багато місця на ASIC (застосування цього підходу дозволяє збільшити продуктивність оброблення на апаратному рівні). Тобто OpenFlow вимагає більше ресурсів на кожен запис на відміну від наприклад, звичайного Ethernet трафіка, де використовується лише хеш-пам'ять і працює значно економніше. До порівняння, Ethernet forwarding потребує лише 60 біт на один запис (48-бітова MAC-адреса + 12 біт VLAN ID), дозволяючи зберігати до 64 000 записів (Rout, Patra, & Sahoo, 2017, с. 543–551). В той час як OpenFlow потребує 288 біт на правило, що суттєво зменшує можливу кількість таких записів до ~1500.

Відповідно правила OpenFlow генеруються та зберігаються на рівні кожного потоку, а саме для кожного нового клієнта, який ініціює з'єднання, необхідно створити окреме правило. Наприклад, середній ToR-комутатор (Top-of-Rack switch) може обслуговувати близько 10 000 потоків одночасно, що просто фізично перевищує кількість правил, які можна було б зберігати в TCAM-пам'яті. Тому жоден із наявних і запропонованих підходів OpenFlow не є ідеальним, оскільки у великих мережах дата-центрів цей процес може спричинити появу bottleneck на деяких ділянках мережі, саме тому сучасні рішення все частіше використовують або гібридні моделі, або розподілену аналітику, щоб мати баланс між продуктивністю й ефективністю (Gilliard et al., 2024, vol. 35).

Розроблення підходу DevoFlow націлено на часткове повернення рівня управління над потоками назад на комутатори, однак централізований контроль і видимість важливих потоків лишається на контролерах, що дає змогу зменшити навантаження на SDN-контролер. Зазначена технологія модифікує модель відкритого потоку у такий спосіб, аби перерозподілити якомога більше рішень на комутатори так, щоб їх можна було просто, економічно й ефективно реалізувати на апаратному рівні. Також це може дозволити зменшити кількість записів TCAM-пам'яті та надати нові механізми для ефективного виявлення QoS-значущих потоків.

Згідно з моделлю DevoFlow є кілька методів ідентифікації великих потоків:

- по таймерах на комутаторах (Timeouts). Якщо потік існує довше певного часу (напр., більше ніж 1 секунда), то він починає вважатися великим (elephant flow);
- по перевищенню заданого порогу обсягу переданих даних (Byte Threshold). Якщо потік передав більше зазначеного обсягу трафіка (напр., 10 МБайт), тоді такий потік маркується як elephant flow. Відповідно це й дозволяє фільтрувати маленькі запити HTTP, DNS і сфокусуватись на FTP, стрімінгу;
- вибірка або семплінг (Packet Sampling), де комутатор періодично аналізує пакети (напр., кожен 100-й пакет). Якщо для певного потоку фіксується велика кількість пакетів за короткий час, він маркується як elephant flow;
- тригери на основі затримки або перевантаження каналів зв'язку (Load-Based Triggers). Якщо канал мережі використовується більш ніж на 70 %, комутатор шукає потоки, які безпосередньо спричиняють таке навантаження, та класифікується як elephant flow.

Коли виявляється великий потік (elephant flow) вже контролер обчислює доступні шляхи і перенаправляє трафік по найменш завантаженому альтернативному маршруту. Механізми виявлення потоків, що використовуються в Hedera та Devoflow, мають досить високі накладні витрати як апаратних так і мережевих ресурсів, що особливо критично для менш продуктивного обладнання рівня інфраструктури SDN-мережі.

Методи

З огляду на зазначене вище в цій роботі, на основі методів аналізу складних систем, тобто аналізу порядку взаємодії базових елементів SDN-мережі за різними технологіями та протоколами, була запропонована архітектура модифікованого підходу DevoFlow. Також за рахунок декомпозиції визначено функціональні можливості програмних модулів, які дозволяють обробляти дані про поточний стан мережі та здійснювати класифікацію вхідних потоків за обсягом із метою балансування навантаження й уникнення аварійних ситуацій.

Зауважимо, що існують проблеми із збиранням статистики потоків, оскільки модель OpenFlow підтримує два методи отримання статистики:

- перший із них push-based, де контролер отримує повідомлення про всі зміни (тобто комутатори автоматично надсилають статистику на контролер, щойно в потоці відбувається певна подія), що створює суттєве навантаження в мережі.
- другий метод – pull-based, в якому контролер періодично запитує дані, що знижує навантаження мережі, але разом із цим падає ефективність збору статистики через свою неоперативність і затримки (оскільки дані оновлюються не миттєво).

Реалізація функції збору статистики напряму пов'язана з кількома ключовими обмеженнями − як з боку мережевих пристроїв (комутаторів), так і з боку самого контролера. Якщо розглядати метод push-based, то кожен новий потік, що надходить, або зміна стану потоку (таймери, лічильники) активує надсилання службових повідомлень до контролера. Для додаткового оброблення такого обсягу службового трафіка контролер потребує високої продуктивності CPU та достатньої кількості пам'яті.

Якщо говорити про pull-based метод, то запити контролером охоплюють усі активні потоки (навіть ті, в яких нічого не змінилося). Відповідно, комутатори повинні формувати відповіді по кожному потоку, обробляючи ці додаткові операції на своїх і так досить обмежених обчислювальних ресурсах (ASIC або CPU). А такі відповіді займають значний обсяг пам'яті та пропускної здатності каналів мережі, особливо це критично в мережах дата-центрів із тисячами потоків.

Як результат, збір статистики займає або занадто багато ресурсів, що може знижувати продуктивність мережі, або збільшення часу періоду опитування зменшує ефективність роботи контролера у реальному часі у випадку

ISSN 2788-6603

перенавантажень. Тому жоден із наявних і запропонованих підходів OpenFlow не є ідеальним, оскільки у великих мережах дата-центрів цей процес може спричинити появу bottleneck на деяких ділянках мережі, саме тому сучасні рішення все частіше використовують або гібридні моделі, або розподілену аналітику, щоб мати баланс між продуктивністю й ефективністю.

Відповідно, у розробленні алгоритму визначення допустимого часу надсилання статистики від комутаторів до SDN-контролера, на основі методу push-based, були використані евристичні правила обслуговування службового трафіка з урахуванням поточних показників мережі.

Результати

На відміну від запропонованих рішень в роботах (Costa et. al., 2021, vol. 147; Yu et. al., 2018, с. 251–258), щодо порядку роботи мережевого обладнання на основі DevoFlow, в цій статті пропонується змінити порядок збору й оброблення службової інформації про характеристики потоків. Тобто комутатори (OpenFlow Switch) в загальному працюють так:

- комутатори не приймають рішень, щодо розрахунку шляхів для передачі даних, а здійснюють комутацію пакетів лише на основі таблиць потоків;
- комутатори збирають статистичні дані (напр., кількість переданих пакетів або байтів) для кожного потоку, використовуючи відповідні поля в таблицях потоків (counters);
- комутатори самостійно відправляють дані на контролер (push-based). Поки сесія передачі потоку активна (на період передачі), статистичні дані на лічильниках (counters) постійно оновлюються, а їхні значення не обмежуються.

Також на SDN-контролер покладають такі завдання:

- збір, узагальнення та підтримка актуальних статистичних даних за кожен потік;
- збір, узагальнення та підтримка актуальних даних про поточну топологію мережі (лінії та активне мережеве обладнання);
- збір, узагальнення та підтримка актуальних даних про поточний стан функціонування всієї мережі (напр., розрахунок завантаження ліній зв'язку);
- класифікація потоків. Тобто, обрахунок обсягів потоків і присвоєння відповідним потокам міток: малий " mice-flows", середній "medium-flows" і великий потік "elephant-flows". У свою чергу, потік – це сесія передачі даних між відправником та отримувачем у мережі.
- оперативний перерозподіл навантаження для значущих потоків (напр., після ідентифікації elephant flow перерахунок і визначення альтернативних шляхів передачі);
- передача отриманих у процесі розрахунків маршрутної інформації OpenFlow-правил на комутатори та їх подальше корегування залежно від одержаних поточних статистичних даних.

На рис. 2 представлено архітектуру й основні компоненти SDN-мережі згідно з удосконаленим підходом DevoFlow.

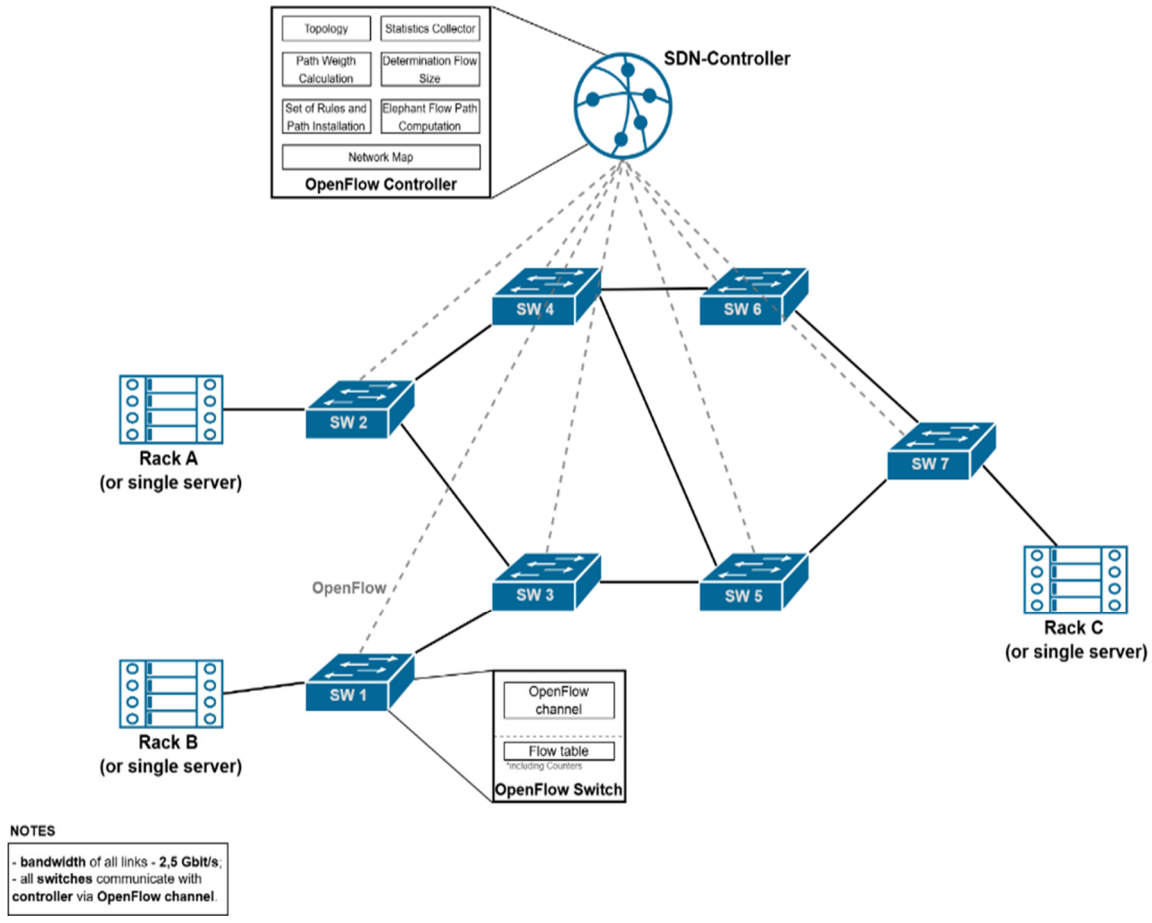

Рис. 2. Варіант архітектури SDN-мережі дата-центру на основі вдосконаленого підходу DevoFlow

Центральним компонентом зазвичай є SDN-контролер, що включає:

- програмно-функціональний блок топології (Topology), який зберігає інформацію про всі наявні в мережі вузли (комутатори) та канали;
- блок збору статистики (Statistics collector), який відстежує завантаження каналів, періодично збираючи (агрегуючи) інформацію про завантаженість портів комутатора;
- блок обчислення ваги шляху (Path weight calculation), який розраховує найкоротші шляхи передачі для вхідних потоків між будь-якою парою джерело-одержувач;
- блок визначення розміру потоку (Determination flow size), який розраховує (класифікує) розмір потоку залежно від порогових значень та обраних параметрів індентифікації;
- блок обчислення шляхів для великих "elephant flow" потоків (Elephant flow path computation), який розраховує шляхи для потоків "elephant flow" у процесі використання алгоритмів багатошляхової маршрутизації наприклад ECMP;
- блок пересилання та контролю доставки таблиць потоків (Set of rules and path installation), який відправляє правила передачі трафіка (flow table) на комутатор;
- блок контролю функціонування мережі (Network map) – узагальнена інформація про поточний стан усієї мережі (напр., поточна матриця циркулюючого трафіка, прокладені (актуальні) маршрути тощо).

Тепер розглянемо в динаміці застосування багатошляхового протоколу ECMP для балансування навантаження у мережі у випадку ідентифікації "elephant flow". На рис. 3 представлено приклад розподілу навантаження з урахуванням запропонованих модифікацій.

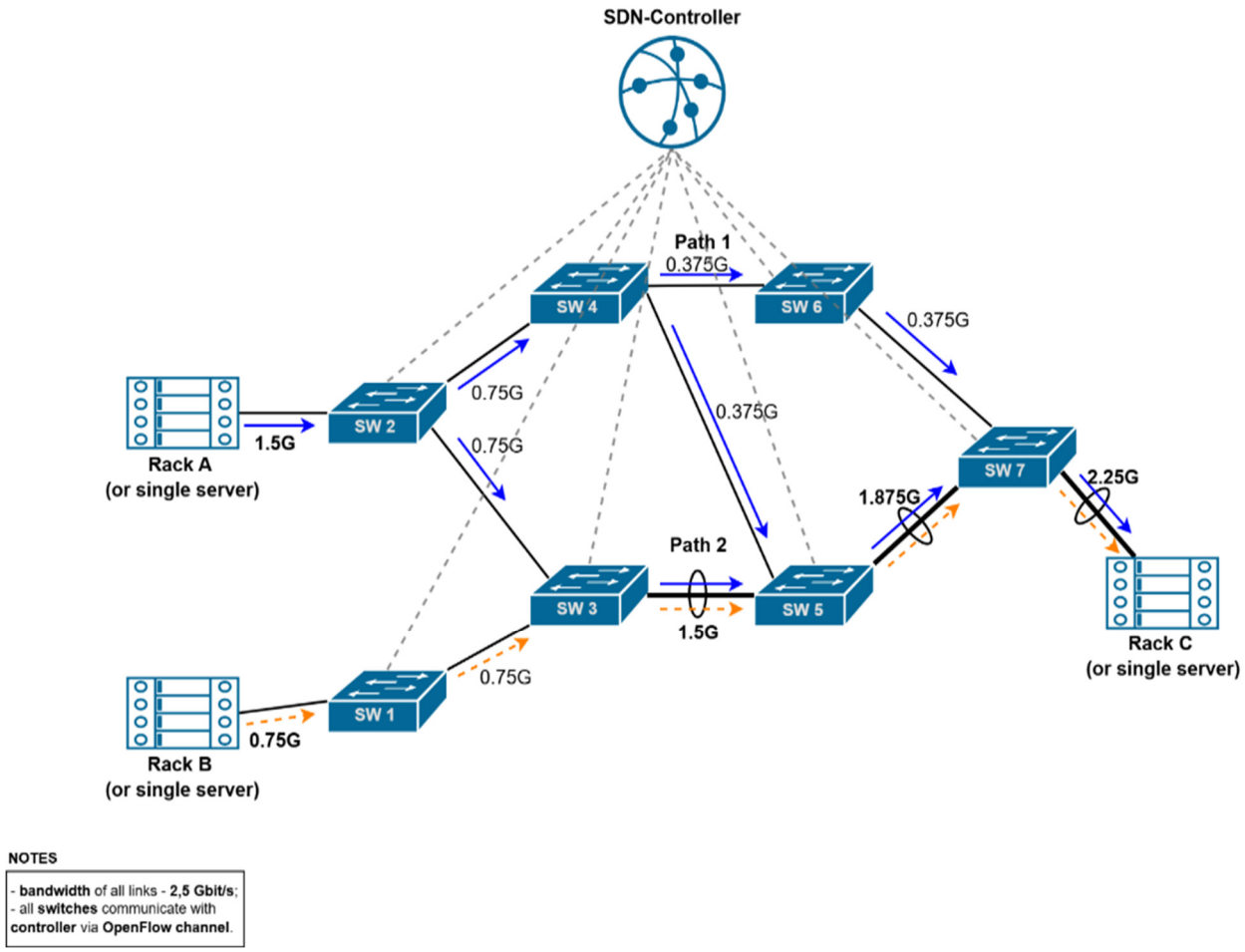

Рис. 3. Розподіл навантаження для модифікованого DevoFlow для великих потоків (elephant flows) на основі протоколу ECMP

Згідно з прикладом і наведеною топологією, відбувається передача трафіка із двох серверів (із сервера А зі швидкістю 1,5 Гбіт/с та із сервера В зі швидкістю 750 Мбіт/с) на сервер С, бітова швидкість усіх каналів зв'язку в мережі – 2,5 Гбіт/с. Особливість ECMP полягає в тому, що на кожному комутаторі відбувається рівноцінний (навантаження ділиться навпіл) розподіл вхідного трафіка між шляхами передачі з однаковою ціною (приймаємо однакову кількість транзитів), якщо ціна одного шляху менша за ціну інших, то розподіл навантаження не відбувається.

Під час передачі трафіка із сервера В на С, його маршрут буде без будь-яких розподілів, оскільки найвигіднішим (мінімальною кількістю транзитів) є маршрут через:

SW1 – SW3 – SW5 – SW7.

ISSN 2788-6603

За передачі трафіка із сервера A до сервера C, існує три рівноцінні маршрути, а саме:

SW2 – SW4 – SW6 – SW7 (Path 1);
SW2 – SW3 – SW5 – SW7 (Path 2);
SW2 – SW4 – SW5 – SW7.

При надходженні потоку на комутатор SW2, контролер має відомості, що є два рівноцінні шляхи передачі (через SW3 та SW4) і тому рівномірно розподіляє трафік між лініями зв'язку цих комутаторів із метою балансування навантаження. Далі розглянемо окремо випадки передачі шляхами 1 та 2:

Path 1. Потік надходить на SW4, контролер має відомості, що має 2 рівноцінні шляхи передачі (через SW5 та SW6) і тому рівномірно розподіляє вхідний трафік між інцидентними лініями цих комутаторів. Частина трафіка після попереднього поділу через SW6 та SW7 доставляється до сервера C.

Path 2. Потік, який проходить по даному шляху об'єднується з потоком від сервера B (в лініях зв'язку між SW3 – SW5, а також SW5 – SW7). Зазвичай потоки об'єднуються на лінії між SW7 і сервером C.

Подальше вдосконалення спрямовано на розроблення аналітичної залежності (алгоритму) для визначення мінімально допустимого часу надсилання статистичних даних (поточне значення лічильників) від комутаторів на контролер. Загальний алгоритм розрахунку допустимого періоду надсилання такий:

- аналізуються поточні значення швидкості на всіх лініях зв'язку $\|V_{link_{ij}}\|$ мережі й обирається найменша швидкість передачі $\min(V_{link})$;
- від визначеної мінімальної швидкості розраховується значення, що становить не більше 10 %, на передачу службової інформації, причому отримане значення ділиться на кількість активних сесій (NAS): $V_{threshold} \leq \left[\frac{(V_{link}) \times 0{,}1}{NAS}\right]$;
- розраховується допустимий час ($T_{allowable}$) надсилання статистики від комутаторів до SDN-контролера згідно з аналітичним виразом:

$$T_{allowable} = \frac{1}{\frac{V_{threshold}}{L_{frame}}}, \quad (1)$$

де L_{frame} – обсяг (розмір) кадру, вміст якого включає службову інформацію, тобто необхідні значення лічильників.

Для проведення експериментів і верифікації запропонованих рішень використано програмний продукт MiniEdit для створення імітаційної моделі сегменту SDN-мережі. По суті, MiniEdit являє собою графічний редактор, в якому користувачу надається можливість власноруч побудувати свою мережу, без прямої роботи з кодом, оскільки більшість параметрів можна налаштувати із самого редактора. MiniEdit є частиною програмного продукту Mininet, що представляє собою віртуальну платформу для емуляції сегментів SDN-мережі, де можна використовувати віртуальні хости, OpenFlow комутатори та контролери (як вбудовані, що завантажуються за замовчуванням із пакетами Mininet, так і з можливістю підключення зовнішніх різних розробників).

Вихідними даними щодо створення імітаційної моделі стала топологія з мережевими параметрами, яка представлена на рис. 3.

Для початку виконуємо побудову нашого сегменту SDN-мережі шляхом наповнення мережевого "полотна" в MiniEdit елементами мережі (контролер, віртуальні OpenFlow комутатори, віртуальні хости). В нашому випадку для емуляції роботи кластерів серверів (Rack A, B та C) використано віртуальні хости (h1, h2, h3). Як базовий, був використаний зовнішній контролер від Hewlett Packard (HPE VAN SDN Controller). Скомпонована схема SDN-мережі для проведення експериментів виглядатиме таким чином (рис. 4).

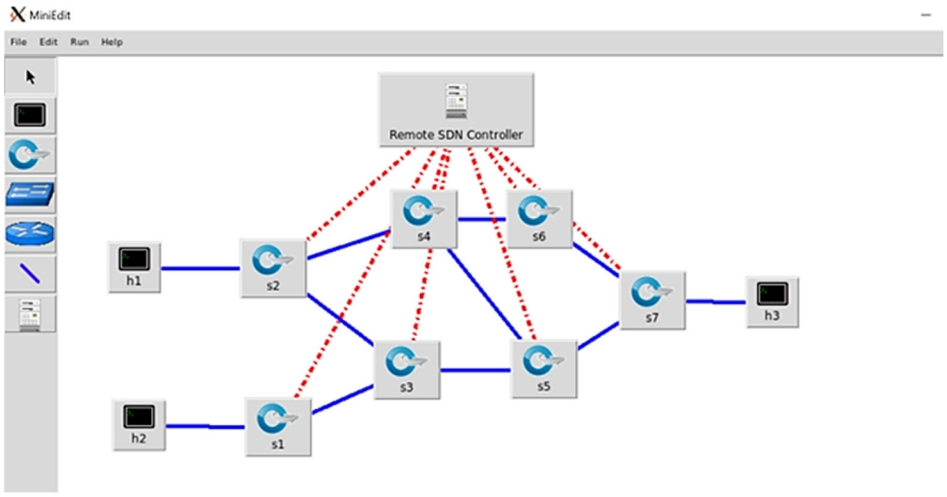

Рис. 4. Візуалізація топології сегменту SDN-мережі дата-центру в MiniEdit

Наступним етапом налаштування є конфігурація контролера для підключення до Mininet (де розгорнута топологія мережі: OpenFlow комутатори, хости, лінії зв'язку), а також налаштування необхідних мережевих параметрів у лініях зв'язку (delay, bandwidth та ін.), що показано на рис. 5.

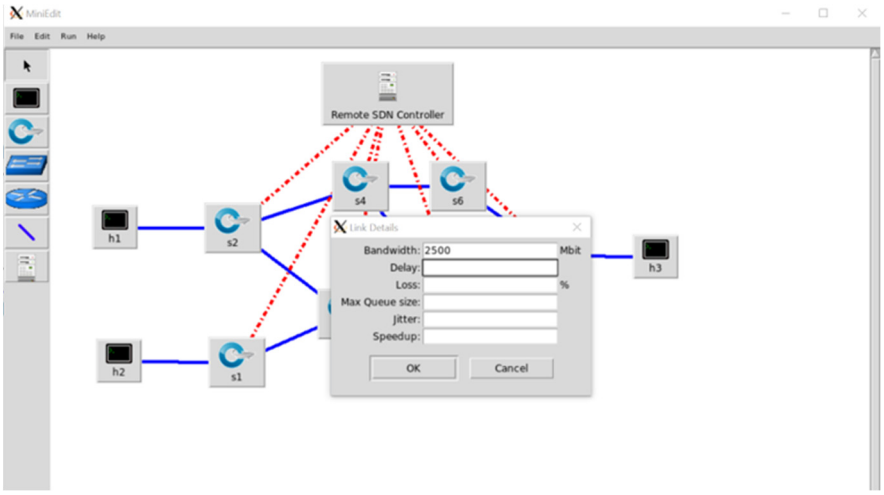

Рис. 5. Порядок налаштування параметрів ліній зв'язку в MiniEdit

В результаті проведених експериментів у процесі зміни величини навантаження в мережі та використанні різних методів збору статистичних даних про характеристики потоків, була зібрана статистика, що представлена на рис. 6. А саме, порівнювали такі методи: метод pull-based (контролер відправляє запит на комутатор із метою отримати статистичні дані) та метод push-based (комутатор самостійно з деякою періодичністю надсилає статистичні дані на контролер).

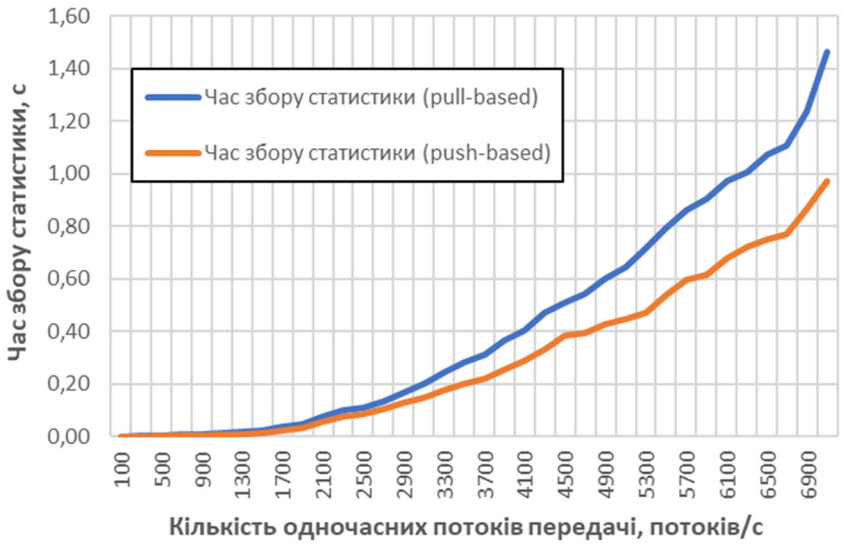

Рис. 6. Залежність часу отримання статистики від розмірності таблиць потоків для методів pull- та push-based

Аналіз отриманих залежностей свідчить про те, що зі збільшенням навантаження в мережі метод push-based демонструє кращі результати, щодо часу отримання статистики, ніж метод pull-based, та відповідно дозволяє отримувати частіше актуальну інформацію про стан мережі. Відповідно push-based метод пропонується використовувати на рівні управління в модифікованій архітектурі на базі підходу DevoFlow.

Подальші експерименти на імітаційній моделі сегменту мережі SDN (див. рис. 4) були направлені на перевірку запропонованої аналітичної моделі допустимого часу надсилання статистики на контролер (аналітичний вираз 1).

На рис. 7 графічно узагальнено отримані експериментальні дані порівняно з аналітичною моделлю для однакових вхідних даних.

Згідно з аналізом результатів проведених експериментів (рис. 6, 7) можемо відмітити таке:

- метод отримання статистичних даних push-based є в рази швидшим за метод pull-based, що підтверджується експериментом на імітаційній моделі SDN-мережі при емуляції різних рівнів навантаження. Посилаючись на залежності на рис. 6, можна побачити, що при передачі 4500 потоків за секунду в мережі, час збору статистики методом pull-based становить 0,51 секунди, в той час як час збору методом push-based – 0,38 секунди, що приблизно в 1,3 раза швидше;
- результати експерименту, зображені на рис. 7, підтверджують достовірність запропонованої аналітичної моделі допустимого часу надсилання статистики на контролер (аналітичний вираз 1), а сама модель враховує залежність зміни інтенсивності надсилання статистичних даних від завантаженості елементів мережі.

ISSN 2788-6603

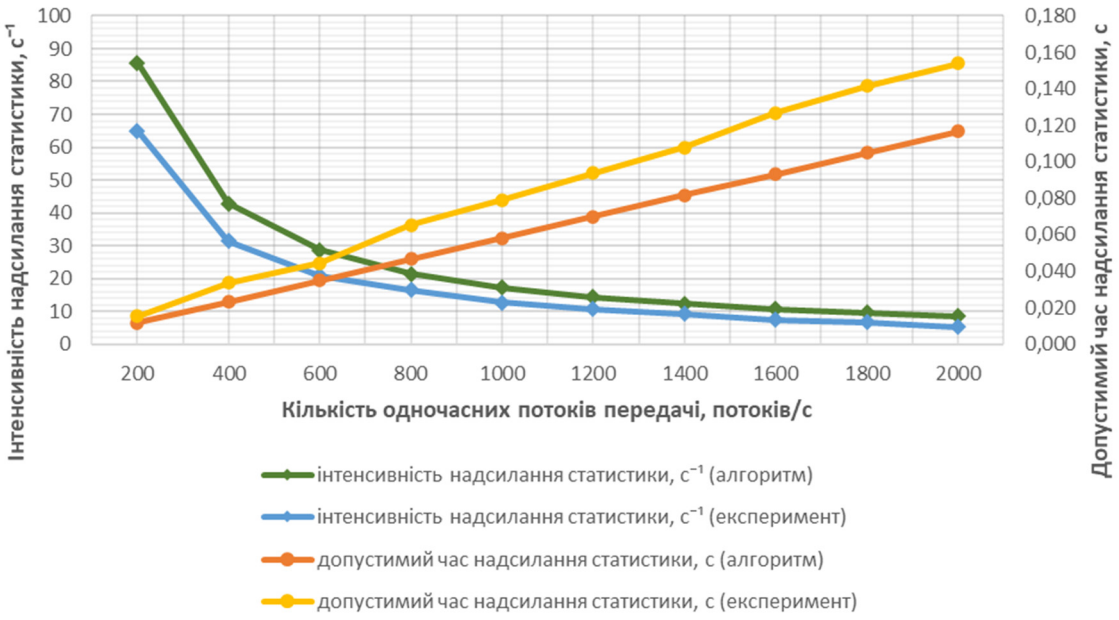

Рис. 7. Залежність часу й інтенсивності надсилання статистики від кількості потоків у мережі

Дискусія і висновки

В результаті дослідження проведено аналіз переваг і недоліків технологій Hedera та DevoFlow. На основі переваг запропоновано модифіковану архітектуру рівня управління з урахуванням підходів DevoFlow. Це дозволило здійснити декомпозицію та визначити функціональні можливості програмних модулів SDN-контролерів дата-центрів для підвищення ефективності оброблення та розподілу вхідних потоків.

Для отримання статистики (службової інформації) з обладнання SDN-мережі дата-центрів запропоновано використовувати push-based метод, який показує кращі результати щодо оперативності збору статистики за коливання навантаження порівняно з методом pull-based.

В подальшому, на основі отриманих статистичних даних від комутаторів пропонується здійснювати класифікацію вхідних потоків та їх розбиття на малі "mice-flows", середні "medium-flows" та великі потоки "elephant-flows". Це дає змогу використовувати різні стратегії оброблення даних потоків для рівномірного завантаження елементів мережі і як наслідок – зменшення ймовірності появи перенавантажень.

У цій статті запропоновано алгоритм визначення допустимого часу надсилання статистики від комутаторів (метод push-based), який враховує динаміку зміни навантаження в лініях в мережі, кількість активних сесій і базується на евристичному правилі обмеженні обсягу службового трафіка.

Для перевірки достовірності запропонованих рішень розроблено імітаційну модель у графічному емуляторі MiniEdit і проведено серію експериментів. Аналіз результатів моделювання підтверджує використання push-based методу як основного для отримання статистики контролером, а також достовірність розроблених аналітичних залежностей для визначення допустимого часу надсилання статистичних даних на контролер.

Подальші дослідження будуть направлені на визначення впливу на мережу різних за обсягом і терміном функціонування потоків ("mice-flows", "medium-flows" "elephant-flows") і розроблення методів (моделей) збалансованого розподілу навантаження.

Внесок авторів. Микола Нестеренко – розроблення архітектури управління на базі модифікованого підходу DevoFlow та аналітичних залежностей для визначення допустимого часу надсилання статистичних даних на контролер, проведення аналізу і порівняння існуючих підходів і технологій, написання висновків; Антон Марінов – огляд літературних джерел, збір теоретичних даних і результатів досліджень, опис балансування навантаженням в мережі модифікованого підходу.

Джерела фінансування. Це дослідження не отримало жодного гранта від фінансової установи в державному, комерційному або некомерційному секторах.

Список використаних джерел

Costa, L. C., Vieira, A. B., de Britto e Silva, E., Macedo, D. F., Vieira, L. F. M., Vieira, M. A. M., da Rocha Miranda, M., Batista, G. F., Polizer, A. H., Gonçalves, A. V. G. S., Gomes, G., & Correia, L. H. A. (2021). OpenFlow data planes performance evaluation. *Performance Evaluation*, 147. https://doi.org/10.1016/j.peva.2021.102194

Gilliard, E., Liu, J., Aliyu, A. A., Juan, D., Jing, H., & Wang, M. (2024). Intelligent load balancing in data center software-defined networks. *Transactions on Emerging Telecommunications Technologies*, 35(4). https://doi.org/10.1002/ett.4967

Globa, L., Skulysh, M., Romanov, O., & Nesterenko, M. (2019). Quality Control for Mobile Communication Management Services in Hybrid Environment. У M. Ilchenko, L. Uryvsky, & L. Globa (Eds.), *Advances in Information and Communication Technologies: Vol. 560. Lecture Notes in Electrical Engineering* (pp. 76–100). Springer. https://doi.org/10.1007/978-3-030-16770-7_4

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

Romanov, O., Nesterenko, M., & Mankivskyi, V. (2021). The method of redistributing traffic in mobile network. In D. Ageyev, T. Radivilova, & N. Kryvinska (Eds.), *Data-Centric Business and Applications* (Lecture Notes on Data Engineering and Communications Technologies, Vol. 69, pp. 159–182). Springer. https://doi.org/10.1007/978-3-030-71892-3_7

Romanov, O., Siemen, E., Nesterenko, M., & Mankivskyi, V. (2021). Mathematical Description of Control Problems in SDN Networks. In *International Conference on Applied Innovations in IT (ICAIIT)* (pp. 33–40). http://dx.doi.org/10.25673/36582.

Rout, S., Patra, S.S., Sahoo, B. (2017). Performance Evaluation of the Controller in Software-Defined Networking. У H. Behera, & D. Mohapatra (Eds.) *Computational Intelligence in Data Mining:* Vol 556. *Advances in Intelligent Systems and Computing* (с. 543–551). Springer. https://doi.org/10.1007/978-981-10-3874-7_51

Shirmarz, A., & Ghaffari, A. (2020). Performance issues and solutions in SDN-based data center: a survey. *The Journal of Supercomputing*, 76(10), 7545–7593. https://doi.org/10.1007/s11227-020-03180-7

Tennakoon, D., Chowdhury, M., & Luan, T. H. (2018). Cloud-based load balancing using double Q-learning for improved Quality of Service. *Wireless Netw*, 29, 1043–1050. https://doi.org/10.1007/s11276-018-1888-8

Wang, C., Cao, W., Hu, Y., & Liu, J. (2023). Data Center Traffic Scheduling Strategy for Minimization Congestion and Quality of Service Guaranteeing. *Computers, Materials & Continua*, 75(2), 4377–4393. https://doi.org/10.32604/cmc.2023.037625

Yu, H., Qi, H., Li, K., Zhang, J., Xiao, P., & Wang, X. (2018). Openflow Based Dynamic Flow Scheduling with Multipath for Data Center Networks. *Computer Systems Science and Engineering*, 33(4), 251–258. https://doi.org/10.32604/csse.2018.33.251

References

Costa, L. C., Vieira, A. B., de Britto e Silva, E., Macedo, D. F., Vieira, L. F. M., Vieira, M. A. M., da Rocha Miranda, M., Batista, G. F., Polizer, A. H., Gonçalves, A. V. G. S., Gomes, G., & Correia, L. H. A. (2021). OpenFlow data planes performance evaluation. *Performance Evaluation*, 147. https://doi.org/10.1016/j.peva.2021.102194

Gilliard, E., Liu, J., Aliyu, A. A., Juan, D., Jing, H., & Wang, M. (2024). Intelligent load balancing in data center software-defined networks. *Transactions on Emerging Telecommunications Technologies*, 35(4). https://doi.org/10.1002/ett.4967

Globa, L., Skulysh, M., Romanov, O., & Nesterenko, M. (2019). Quality Control for Mobile Communication Management Services in Hybrid Environment. In Ilchenko, M., Uryvsky, L., Globa, L. (Eds.), *Advances in Information and Communication Technologies*: Vol. 560. *Lecture Notes in Electrical Engineering* (pp. 76–100). Springer. https://doi.org/10.1007/978-3-030-16770-7_4

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

Romanov, O., Nesterenko, M., & Mankivskyi, V. (2021). The method of redistributing traffic in mobile network. In D. Ageyev, T. Radivilova, & N. Kryvinska (Eds.), *Data-Centric Business and Applications* (Lecture Notes on Data Engineering and Communications Technologies, Vol. 69, pp. 159–182). Springer. https://doi.org/10.1007/978-3-030-71892-3_7

Romanov, O., Siemen, E., Nesterenko, M., & Mankivskyi, V. (2021). Mathematical Description of Control Problems in SDN Networks. In *International Conference on Applied Innovations in IT (ICAIIT)* (pp. 33–40). http://dx.doi.org/10.25673/36582

Rout, S., Patra, S.S., Sahoo, B. (2017). Performance Evaluation of the Controller in Software-Defined Networking. У H. Behera, & D. Mohapatra (Eds.) *Computational Intelligence in Data Mining:* Vol 556. *Advances in Intelligent Systems and Computing* (с. 543–551). Springer. https://doi.org/10.1007/978-981-10-3874-7_51

Shirmarz, A., & Ghaffari, A. (2020). Performance issues and solutions in SDN-based data center: a survey. *The Journal of Supercomputing*, 76(10), 7545–7593. https://doi.org/10.1007/s11227-020-03180-7

Tennakoon, D., Chowdhury, M., & Luan, T. H. (2018). Cloud-based load balancing using double Q-learning for improved Quality of Service. *Wireless Netw*, 29, 1043–1050. https://doi.org/10.1007/s11276-018-1888-8

Wang, C., Cao, W., Hu, Y., & Liu, J. (2023). Data Center Traffic Scheduling Strategy for Minimization Congestion and Quality of Service Guaranteeing. *Computers, Materials & Continua*, 75(2), 4377–4393. https://doi.org/10.32604/cmc.2023.037625

Yu, H., Qi, H., Li, K., Zhang, J., Xiao, P., & Wang, X. (2018). Openflow Based Dynamic Flow Scheduling with Multipath for Data Center Networks. *Computer Systems Science and Engineering*, 33(4), 251–258. https://doi.org/10.32604/csse.2018.33.251

Отримано редакцією журналу / Received: 20.05.25
Прорецензовано / Revised: 02.06.25
Схвалено до друку / Accepted: 18.06.25

Mykola NESTERENKO, DSc (Engin.), Assoc. Prof.
ORCID ID: 0000-0003-0812-2793
e-mail: mykola.nesterenko@viti.edu.ua
Kruty Heroes Military Institute of Telecommunications and Information Technologies, Kyiv, Ukraine

Anton MARINOV, PhD Student
ORCID ID: 0000-0001-8717-2734
e-mail: marinov.anton@lll.kpi.ua
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine

MANAGEMENT OF DATA CENTER SDN BASED ON THE MODIFIED DEVOFLOW APPROACH

Background. *The paper investigates the features of implementing the SDN network management layer when servicing data center flows of different volumes based on Hedera technology, the OpenFlow protocol, and the DevoFlow approach. The distribution of processing functions, collection of statistical data, and the frequency of updating statistical data affect both the amount of service information and the efficiency and effectiveness of management in terms of distributing incoming traffic for load balancing. The aim of the work is to modify the Hedera approach by redistributing the functions of flow classification and adaptive control depending on the flow volume at the control level, as well as to develop an algorithm for determining the allowable time for sending statistics from switches to the SDN controller.*

Methods. *The method of system analysis and decomposition is used to study complex systems, methods of collecting data on the state of the network, as well as heuristic rules for determining the volume of service traffic.*

Results. *The paper analyzes the features of construction and operation of technologies like Hedera and DevoFlow, recommend a modified architecture for building the control plane and identifies the main functions of software modules that can improve the efficiency of SDN controllers. Due to changing the procedure for collecting and processing service information on the equipment of the SDN network of data centers and returning to a centralized type of control.*

Approaches to the classification of incoming flows and their division into small "mice-flows", medium "medium-flows" and large "elephant-flows" are determined, which allows further use of multi-path routing for more efficient use of network resources. Also, this article proposes an algorithm for determining the permissible time for sending statistics from switches (push-based method), which takes into account the dynamics of load changes in the lines in the network, the number of active sessions and is based on a heuristic rule for limiting the amount of service traffic.

Conclusions. *A modified architecture of the control level based on the DevoFlow approach has been developed and the main functions of the software modules of the SDN controller of the data center network have been determined. Also, some analytical dependencies have been developed to determine the approved time for sending statistics from switches to the SDN controller, taking into account the current state of network congestion. Additionally, specific experiments were conducted to confirm both the reliability of the chosen method for obtaining statistics and the proposed algorithm for determining the permissible time for sending statistics from switches to the SDN controller.*

Keywords: *Software-defined Networking, data center, OpenFlow, Hedera, DevoFlow, elephant-flows, mice-flows, SDN controller, OpenFlow Switch, flow table.*

Автори заявляють про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; в рішенні про публікацію результатів.

The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; in the decision to publish the results.

ISSN 2788-6603