МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХТЕХНОЛОГІЙ

Кваліфікаційна наукова
праця на правах рукопису

ДМИТРЕНКО ВОЛОДИМИР ВІТАЛІЙОВИЧ

УДК 004.896

ДИСЕРТАЦІЯ

МОДЕЛЬ ТА МЕТОД ПОБУДОВИ КОНВЕРГЕНТНИХ ПРОГРАМНО-КОНФІГУРОВАНИХ МЕРЕЖ З ГАРАНТОВАНОЮ ДОСТОВІРНІСТЮ УПРАВЛЯЮЧОЇ ІНФОРМАЦІЇ

Спеціальність 123 «Комп’ютерна інженерія»

Галузь знань 12 «Інформаційні технології»

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____В.В.Дмитренко

Науковий керівник: Заїка Віктор Федорович, доктор технічних наук, професор

Київ – 2023

АНОТАЦІЯ

Дмитренко В.В. Модель та метод побудови конвергентних програмно-конфігурованих мереж з гарантованою достовірністю управляючої інформації. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 123 «Комп'ютерна інженерія» — Державний університет інформаційно-комунікаційних технологій Міністерства освіти і науки України, Київ, 2024.

Дисертаційна робота присвячена розробці моделі та методу побудови конвергентних програмно-конфігурованих мереж для дослідження затримки пакетів за різних навантажувальних характеристик процесів обробки трафіку з гарантованою достовірністю управляючої інформації.

Вирішення завдання мінімізації пакетів SDN мережі безпосередньо залежить від показників якості параметрів конвергентних програмно-конгфігурованих мереж. Результуючі моделі, що згенеровані програмно-апаратним комплексом, можуть бути використані для побудови мережі та дослідження параметрів її продуктивності. Розроблені методики створення моделей вхідних та вихідних параметрів конвергентних програмно-конфігурованих мереж дозволяють підвищити показники якості передачі даних, послуг та мереж, а також гарантовану достовірність управляючої інформації.

Сучасні високошвидкісні мережі передачі даних повинні володіти передбачуваною поведінкою при будь-яких поєднаннях параметрів трафіку і налаштувань обладнання. Одним із основних пристроїв високошвидкісних магістралей є комутатори, які поступово ускладнювалися, збільшуючи функціональність. На сьогоднішній день провайдери починають використовувати програмно-конфігуровані мережі, які надають більш гнучкі підходи до управління трафіком. Програмно-конфігуровані мережі, що працюють на базі протоколу OpenFlow, є таким підходом до побудови інфокомунікаційних мереж, при якому функції управління мережею відокремлені від рівня передачі даних і винесені на спеціальний сервер-контролер. Вони суттєво покращили керованість мережі та дозволили об'єднати віртуальну та фізичну інфраструктуру під єдиним гарантованим управлінням.

На сьогоднішній день імітаційне моделювання є основним інструментом дослідження SDN мереж. Розроблено досить велику кількість систем імітаційного

моделювання мережі, що спираються у своїй роботі на такі параметри, як кількість вузлів, зв'язки між ними, протоколи, характеристики каналу передачі даних, а також виконувані функції.

Ймовірнісно-часові характеристики обробки трафіку телекомунікаційної мережі залежать від часу роботи всіх обробних трафік-процесів, включаючи роботу черг, фільтрів, налаштувань, маршрутизації і комутації, а також від завантаження ресурсів. Тривалість обробки одного і того ж пакета на однаковому устаткуванні з різними налаштуваннями може відрізнятися в десять і більше разів. Додають додаткові затримки етапи обробки трафіку, які не враховані в стандартних моделях. Найпоширенішою мережею є конвергентна програмно-конфігурована мережа, яка поєднує реалізацію основних функцій, як у традиційному, так і у програмно-конфігуруємому режимі.

В дисертаційній роботі проведено аналіз підходів до автоматизації перенесення характеристик з мережі до системи моделювання. Це дозволило суттєво спростити процес побудови моделей конкретних пристроїв та мереж, а також автоматизовано одержувати характеристики конкретних моделей обладнання та контролювати достовірність моделей статистичними методами.

Методика побудови конвергентної SDN мережі з гарантованою достовірністю управляючої інформації використана для побудови інфокомунікайційних мереж України.

Ключові слова: Мережа, передача даних, програмно-конфігуровані мережі, показники якості, OpenFlow, математичне та комп'ютерне моделювання, імітаційно-аналітичні моделі, комутатор, дестабілізуючі фактори.

ANNOTATION

Dmytrenko V.V. Model and method of building convergent software-configured networks with guaranteed reliability of control information. – Qualifying scientific work on manuscript rights.

Dissertation for obtaining the scientific degree of Doctor of Philosophy in the specialty

123 "Computer Engineering" – State University of Information and Communication Technologies of the Ministry of Education and Science of Ukraine, Kyiv, 2024.

The dissertation is devoted to the development of a model and method of building a simulation-analytical model of convergent software-configured networks for the study of packet delay under various load characteristics of traffic processing processes with guaranteed reliability of control information.

Solving the task of minimizing SDN network packets directly depends on the quality parameters of converged software-configured networks. The resulting models generated by the software and hardware complex can be used to build a network and study its performance parameters. The developed methods of creating models of input and output parameters of convergent software-configured networks allow to increase the quality indicators of data transfer, services and networks, as well as the guaranteed reliability of control information.

Modern high-speed data transmission networks must have predictable behavior for any combination of traffic parameters and equipment settings. One of the main devices of high-speed highways are switches, which gradually became more complicated, increasing functionality. Today, ISPs are beginning to use software-configured networks that provide more flexible approaches to traffic management. Software-configured networks operating on the basis of the OpenFlow protocol are such an approach to the construction of information communication networks, in which network management functions are separated from the level of data transmission and transferred to a special server-controller. They significantly improved network manageability and allowed to combine virtual and physical infrastructure under a single guaranteed management.

To date, simulation modeling is the main tool for SDN network research. A fairly large number of network simulation systems have been developed, relying in their work on such

parameters as the number of nodes, connections between them, protocols, characteristics of the data transmission channel, as well as performed functions.

The probabilistic time characteristics of telecommunication network traffic processing depend on the time of operation of all processing traffic processes, including the operation of queues, filters, settings, routing and switching, as well as on resource loading. The processing time of the same package on the same equipment with different settings can vary by a factor of ten or more. Traffic processing steps that are not considered in standard models add additional delays. The most common network is a converged software-configurable network, which combines the implementation of basic functions in both traditional and software-configurable mode.

The dissertation analyzes approaches to automating the transfer of characteristics from the network to the modeling system. This made it possible to significantly simplify the process of building models of specific devices and networks, as well as automatically obtain the characteristics of specific models of equipment and control the reliability of models using statistical methods.

The method of building a convergent simulation-analytical model of a telecommunications network with guaranteed reliability of control information is used for the construction of information communication networks of Ukraine.

Keywords: Network, data transmission, software-configured networks, quality indicators, OpenFlow, mathematical and computer modeling, simulation and analytical models, switch, destabilizing factors.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. Дмитренко В.В. Аналіз технологій та стандартів зв'язку для мережі IOT / К. О. Домрачева, Н. М. Довженко, В. В. Дмитренко. // Наукові записки Українського науково-дослідного інституту зв’язку. – 2019. – №3. – С. 54–62.

2. Дмитренко В.В Аналіз переваг мережі ip/mpls / Александров М.В., Кращенко Д.Д., Дмитренко В.В. // // Зв’язок. – 2019. – №6. – С.50 – 59.

3. Дмитренко В.В. Моделювання цифрового коректору міжсимвольних спотворень сигналів у каналі передачі інформації / Заїка В.Ф., Ярцев В. П., Дмитренко В.В., Фокін В.І., Котомчак О.Ю// Телекомунікаційні та інформаційні технології. – 2020. – № 3 (68)– С. 94-106.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Дмитренко В.В. Імітаційно-аналітичні моделі гібридних програмно-конфігурованих пристроїв./ Дмитренко В. В., Марчук О. М. // Зв’язок. , №6(166), 2023, С. 37-43.

2. Дмитренко В.В. Проблеми синтезу моделей мережного обладнання та методи їх вирішення/ Дмитренко В.В., Брезіцький С.М. // Телекомунікаційні та інформаційні технології, № 4 (81) , 2023, С. 95-100.

Наукові праці, які засвідчують апробацію матеріалів дисертації

1. Liubov Berkman, Olexandr Turovsky, Liudmyla Kyrpach, Oksana Varfolomeeva, Volodymyr Dmytrenko, Oleksii Pokotylo. Analyzing the code structures of multidimensional signals for a continuous information transmission channel. Eastern-European Journal of Enterprise Technologies, 5/9 (113), pp. 70–81. 2021. DOI: 10.15587/1729-4061.2021.24235

Зміст

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 9
ВСТУП 10
РОЗДІЛ 1. ОСНОВНІ ПРИНЦИПИ ПОБУДОВИ ПРОГРАМНО – КОНФІГОВАНИЙ МЕРЕЖ 18
1.1 Архітектура програмно-конфігурованих мереж 18
1.2 Стандартизація програмно-конфігурованих мереж 30
1.3 Характеристики надійності програмно – конфігурованих мереж 39
1.4 Організації управління програмно-конфігурованими мережами SDN 44
Висновки до розділу 1 45
РОЗДІЛ 2. ІМІТАЦІЙНО-АНАЛІТИЧНІ МОДЕЛІ ГІБРИДНИХ ПРОГРАМНО-КОНФІГУРОВАНИХ ПРИСТРОЇВ 46
2.1 Розробка мережевого обладнання 46
2.1.1 Дослідження процесу проходження пакета крізь гібридний програмно-конфігурований пристрій на прикладі моделі комутатора 46
2.1.2 Розробка додаткових модулів моделі гібридного програмно-конфігурованого обладнання для реалізації внутрішніх процесів обробки трафіку мережевого пристрою 50
2.2 Розробка моделі комутатора із підтримкою OpenFlow 53
2.3 Розробка моделі мережевого пристрою NFV 56
2.4 Алгоритм автоматизованого синтезу гібридного програмно-конфігурованого обладнання 59
Висновки до розділу 2 62
РОЗДІЛ 3. МЕТОДИ ТА АЛГОРИТМИ АВТОМАТИЗОВАНОГО ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ ГІБРИДНОГО ПРОГРАМНО-КОНФІГУРОВАНОГО ОБЛАДНАННЯ 63
3.1 Отримання параметрів гібридного програмно-конфігурованого обладнання 63
3.2 Розроблення структурної моделі гібридного програмно-конфігурованого обладнання для проведення експериментального дослідження 70
3.2.1 Структурна модель вхідних параметрів гібридного програмно-конфігурованого обладнання 70
3.2.2 Структурна модель вихідних параметрів гібридного програмно-конфігурованого обладнання 74
3.3 Методики автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання 76
3.4 Алгоритм автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання 79
3.5 Багатопараметрична симплексна інтерполяція емпіричних залежностей параметрів пристрою 84
Висновоки до розділу 3 86
РОЗДІЛ 4. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМНО-КОНФІГУРОВАНИХ ПРИСТРОЇВ 87
4.1 Автоматизована система експериментальних досліджень 87
4.1.1 Архітектура випробувального стенду 87
4.1.2 Опис обладнання та ПЗ 89
4.2 Екпериментальне дослідження комутатора 90

4.2.1 Планування експериментального дослідження 90
4.2.2 Експериментальне дослідження комутатора на стенді 92
4.2.3 Сформована модель OMNeT++ для досліджуваного комутатора 94
4.3 Експериментальне дослідження комутаторів за різних параметрів трафіку 98
4.4 Експериментальне дослідження комутаторів за різних конфігураціях 104
4.5 Дослідження параметрів мережі з прикладу експериментального сегмента центру обробки даних 108
4.6 Розрахунок завадостійкості сигнальних сузір'їв 114
4.7 Ефективність каналу передачі управляючої інформації 121
Висновки до розділу 4 133
ВИСНОВКИ 134
Список використаних джерел 136

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ASIC (Application-Specific Integrated Circuit) – інтегральна схема спеціального призначення;

CBFQ (Credit-based fair queueing) – справедлива організація черги на основі кредиту;

DPDK (Data Plane Development Kit) – комплект бібліотек та драйверів розробника для швидкої обробки даних;

DSCP (Differentiated Services Code Point) – точка коду диференційованих послуг;

eMBB (enhanced mobile broadband) - мобільний широкосмуговий зв'язок

FPGA (Field-Programmable Gate Array) – програмована користувачем вентильна матриця;

IDE (Integrated Development Environment) – інтегроване середовище розробки;

IoT (Internet of Things) – Інтернет речей;

FIFO (first in, first out) - черга типу "першим прийшов - першим пішов";

mIoT (massive Internet of Things) – масовий Інтернет речей

NFV (Network Function Virtualization) – віртуалізація мережевих функцій;

OVS (Open vSwitch) – програмний комутатор з відкритим вихідним кодом;

OVSDB (Open vSwitch Database) – база даних станів Open vSwitch;

PQ (priority queuing) – черга з пріоритетами;

RED (Random Early Detection) - алгоритм управління переповненням черг «довільне раннє виявлення»;

TCAM (Ternary Content Addressable Memory) – трійкова асоціативна пам'ять;

TMPFS (Temporary File System) - тимчасове файлове сховище у багатьох Unix-подібних ОС;

TTL (Time to live) – час життя пакету даних у протоколі IP;

VM (Virtual Machine) – віртуальна машина;

WRED (Weighted random early detection) – алгоритм для управління переповненням черг з можливостями запобігання перевантаженням;

WSN (Wireless Sensor Network) – бездротова сенсорна мережа;

ACEI – автоматизована система експериментального дослідження;

ВСТУП

Актуальність теми. Сучасні високошвидкісні мережі передачі даних повинні володіти передбачуваною поведінкою при будь-яких поєднаннях параметрів трафіку, що проходить, і налаштувань обладнання. Основною одиницею побудови високошвидкісних магістралей до кінцевого обладнання є комутатори, які поступово ускладнювалися, збільшуючи функціональність. На сьогоднішній день провайдери починають використовувати програмно-конфігурованих мережі, яке надає більш гнучкі підходи до управління трафіком. Програмно-конфігуруємі мережі (ПКМ), що працюють на базі протоколу OpenFlow, є таким підходом до побудови SDN мереж, при якому функції управління мережею відокремлені від рівня передачі даних і винесені на спеціальний сервер - контролер [14]. ПКМ суттєво покращили керованість мережі, уможливили створення накладених мереж, об'єднання віртуальної та фізичної інфраструктури під єдиним управлінням [24].

Повсюдне використання віртуалізації підвищує вимоги до устаткування, оскільки комутатори повинні поєднувати різнорідний трафік віртуальних сервісів, особливо підходу з допомогою віртуалізації мережевих функцій NFV [80]. Основною проблемою використання всіх цих інструментів є прогнозування затримок, втрати пакетів, розмірів черг задля забезпечення необхідної якості обслуговування (QoS) [10]. Дослідження таких мереж часто виконують на моделях, яких пред'являються підвищені вимоги до точності відтворення умов реального устаткування.

Імітаційне моделювання надає широкі перспективи для досліджень і дозволяє отримати результати, порівняні з результатами реального обладнання, але лише у разі точного опису всіх механізмів роботи та їх часових та навантажувальних характеристик. Цей підхід дає можливість проведення експериментів без побудови реальної мережі, що суттєво знижує тимчасові та фінансові витрати [18].

На сьогоднішній день імітаційне моделювання є основним інструментом дослідження SDN мереж. Розроблено досить велику кількість систем імітаційного

моделювання мережі, що спираються у своїй роботі на такі параметри, як кількість вузлів, зв'язки між ними, протоколи, що використовуються, характеристики каналу передачі даних, а також виконувані мережеві програми [1].

Більшість таких систем дозволяє не будувати модель з нуля, а використовувати готові моделі основних елементів мережі, таких як, наприклад, деякі типи маршрутизаторів, канали зв'язку, протоколи і т.д. Тестування пристроїв з подальшим аналізом принципів роботи дозволяє згенерувати бібліотеку стандартних елементів мережі. Параметри отриманих стандартних елементів можна змінювати за допомогою передбачених у моделях налаштувань. Використання бібліотек готових рішень значно спрощує розробку моделі мережі. Проте, високі темпи розвитку мережевих технологій призводять до того, що існуючі моделі не задовольняють потреб розробників, а багато систем імітаційного моделювання не мають блоків, що описують вплив процесів внутрішньої обробки трафіку в обладнанні на параметри продуктивності, а також не враховують вплив зовнішніх параметрів та конфігурації обладнання на характеристики передачі трафіку.

Моделювання мережевої інфраструктури на основі ПКП має низку особливостей [66]. По-перше, час прийняття рішень щодо комутації та маршрутизації у такому устаткуванні практично не відрізняється і залежить тільки від реалізації пошуку в таблицях OpenFlow у реактивному режимі (із заздалегідь завантаженими даними про топологію). По-друге, у таких мережах інший профіль навантаження на процесор та пам'ять під час проходження трафіку. По-третє, таке устаткування може працювати у гібридних режимах, коли частина портів працює у режимі ПКП, інші в режимі традиційної комутації чи маршрутизації.

Робота черг у таких системах можна порівняти з традиційними комутаторами та маршрутизаторами, крім того, що налаштування параметрів цих черг може відбуватися динамічно. Створення моделей, в яких використовується також проактивний режим роботи ПКС, вимагає введення в процес комутації та маршрутизації роботи стороннього додатка на зовнішньому сервері, доступного через мережу передачі даних з заздалегідь невідомими параметрами по реакції на навантаження [22].

Ймовірнісно-часової характеристики обробки трафіку телекомунікаційного обладнання залежить від часу роботи всіх обробних трафік процесів, включаючи роботу черг, фільтрів, налаштувань, що модифікують, маршрутизації і комутації, а також від завантаження ресурсів. Тривалість обробки одного і того ж пакета на однаковому устаткуванні з різними налаштуваннями може відрізнятися в десять і більше разів. ПКС додають додаткові затримки обробки трафіку, які не враховані в стандартних моделях. Найпоширенішим устаткуванням із підтримкою ПКС є гібридне програмно-конфігуруємих мереж, яке поєднує реалізацію основних функцій, як і традиційному, і програмно- конфигурируємому режимі.

Отримання характеристик обладнання та програмних модулів можливе за допомогою проведення стрес-тестування, причому при використанні ПКС можна генерувати трафік і приймати його лише за допомогою механізмів протоколу OpenFlow у проактивному режимі [28]. Для мінімізації впливу часу запиту та відповіді до сервера решта процесів, крім генерації прийому трафіку, повинні проходити в реактивному режимі з мінімально необхідними розмірами таблиць. Стрес-тестування на реально працюючих мережах пов'язане з труднощами співіснування синтетичного та реального трафіку. З точки зору отримання всіх можливих варіантів та поєднань характеристик обладнання такий режим дозволить створити моделі тільки для навантаження рівного або вище існуючого. Оскільки при використанні ПКС можна знімати дуже докладну статистику з кожної одиниці обладнання, можна проводити такі стрес-тести по дереву пов'язаних комутаторів і маршрутизаторів, одночасно знімаючи характеристики з усіх пристроїв на тому самому патерні трафіку.

Автоматизація перенесення характеристик з обладнання до системи моделювання дозволить суттєво спростити процес синтезу моделі конкретних пристроїв та їх мереж. Автоматизоване одержання характеристик конкретних моделей обладнання дозволить контролювати достовірність моделей статистичними методами.

Існуючі системи імітаційного моделювання комп'ютерних мереж з підтримкою OpenFlow, такі як NS-3, EstiNet, Mininet, OPNET Modeler та ін., не

доцільно використовувати як готове рішення поставленого завдання через такі причини:

- відсутність блоків, що описують вплив процесів внутрішньої обробки трафіку в устаткуванні на параметри продуктивності;

- використання найпростіших алгоритмів проходження пакета крізь обладнання, що не враховують особливості конкретних моделей пристроїв.

Реалізація програмно-апаратного комплексу автоматизованого синтезу гібридного програмно-конфігурованих мереж призведе до необхідності відновлення ймовірно-часових характеристик трафіку на основі експериментальних даних, а також до розробки методів автоматизованого отримання таких даних при довільних вхідних параметрах трафіку, налаштуваннях та поточному навантаженні .

Ступінь розробленості теми дослідження. Актуальною проблемою в рамках дисертаційного дослідження є синтез, що адекватно описує характеристики гібридного програмно-конфігурованого пристроїв лише на рівні перших моментів розподілу тимчасових і навантажувальних значень процесів обробки пакету всередині устаткування.

Питаннями розвитку технології ПКП займалися вчені Беркман Л.Н., Климаш М.М., Вишнівський В.В., Заїка В.Ф., Жураковський Б.Ю., Хлапонін Ю.І., Туровський О.Л., Дробик О.В., Коршун Н.В., Tootoonchian A., Sherwood R., Salman O. У Zhao Y. розглянуто питання управління потоками даних із забезпеченням якості обслуговування у ПКП.

Наукове завдання полягає у розробці моделі, методів, алгоритмів та методик отримання вхідних та вихідних параметрів програмно-конфігуруємих мереж з гарантованою достовірністю управляючої інформації.

Мета та задачі дослідження. Метою роботи є підвищення ефективності дослідження затримки пакетів у SDN мережі за різних навантажувальних характеристик, процесів обробки трафіку з визначеною достовірністю управляючою інформацією. Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести аналіз та обґрунтувати актуальність завдання побудови конвергентної SDN мережі з гарантованою достовірністю управляючої інформації.

2. Розробити модель конвергентної мережі.

3. Розробити метод автоматизованого експериментального дослідження конвергентної програмно-конфігурованої мережі.

4. Розробити методику автоматизованої побудови конвергентної програмно-конфігурованої мережі.

5. Розробити методику оптимального прийому багатопозиційних сигналів з фазорізницевою модуляцією та гарантованою достовірністю управляючої інформації.

6. Виконати експериментальне дослідження параметрів та оцінювання ефективності гібридного програмно-конфігурованого обладнання.

Об'єктом дослідження є процес обробки трафіку в гібридному програмно-конфігурованому пристрої.

Предметом дослідження є імовірнісно-часові та навантажувальні характеристики процесів обробки трафіку гібридним програмно-конфігурованим обладнанням.

Методи дослідження. Дисертацію виконано з використанням апарату теорії ймовірності та математичної статистики, методів проектування інформаційних систем, математичного моделювання, системного аналізу принципів об'єктно-орієнтованого програмування, теорії обчислювальних систем у частині мереж масового обслуговування, методів контейнеризації, теорії позиційної завадозахищеності.

Наукова новизна результатів дисертаційного досліження полягає в наступному:

1. Вперше розроблено модель конвергентної мережі, наукова новизна якої полягає в тому, що вона ґрунтується на теорії інформації та методах імітаційного моделювання, що дозволяє враховувати особливості роботи пристрою та оцінити основні характеристики трафіку, що передається через нього.

2. Вперше розроблено метод автоматизованого експериментального дослідження конвергентного програмно-конфігурованого обладнання, наукова

новизна якого полягає в тому, що він ґрунтується на теорії систем та методах системного аналазі та дозволяє отримати імовірнісні та навантажувальні характеристики обладнання на довільних біфоркаціях трафіку та наборах конфігурації.

3. Вперше розроблено методику автоматизованої побудови гібридного програмно-конфігурованого обладнання, наукова новизна якої полягає в тому, що вона ґрунтується на методах об’єктно орієнтовного підходу та дозволяє автоматичному режимі виконувати побудову гібридного програмно-конфігурованого обладнання.

4. Вперше розроблено алгоритм оптимального прийому багатопозиційних сигналів з фазорізницевою модуляцією, наукова новизна якого полягає в тому, що він ґрунтується на максимально правдоподібній оцінці сигналу та дозволяє отримати максимальну достовірність передаваємої інформації при заданому співвідношенні сигнал/завада.

Практична значення отриманих результатів роботи полягає в реалізації на базі запропонованих алгоритмів програмно-апаратного комплексу, що складається з автоматизованої системи експериментального дослідження гібридного програмно-конфігурованого обладнання та модулів синтезу імітаційно-аналітичних моделей на основі відновлених ймовірно-часових характеристик процесу проходження трафіку через пристрій. Програмно-апаратний комплекс дозволяє спростити процес синтезу моделі гібридного програмно-конфігурованого пристрою, а також проконтролювати достовірність моделей статистичними методами. Результуючі моделі, що згенеровані програмно-апаратним комплексом, можуть бути використані для синтезу моделі мережі та дослідження параметрів її продуктивності. Метод оптимального прийому багатопозиційних сигналів з фазорізницевою інформацією дозволяє на 27% підвищити енергетичний виграш системи.

Методика побудови ефективного цифрового каналу управляючої інформації використана для побудови системи управління інфокомунікайційними мережами України.

Особистий внесок здобувача.

Наукові положення та результати дисертації отримані автором самостійно. Особисто автором здійснена розробка загальної концепції дисертації, вибір об’єктів, визначено мету та задачі роботи, обрано та обґрунтовано методи досліджень. В дисертаційній роботі узагальнено результати досліджень, виконаних автором самостійно і опублікованих в співавторстві в роботах [1-6]. У опублікованих у співавторстві роботах автором дисертації зроблено такий особистий внесок. В [1] Розглянуто стандарти,розроблені Internet Engineering Task Force (IETF), Інститутом інженерів з електротехніки та електроніки (IEEE), Міжнароодним союзом електрозв'язку (МСЕ) та іншими організаціями зі стандартизації.. В [2] розглянуто перспективи впровадження мережі зв’язку на базі технології IP/MPLS, виконано огляд технічних особливостей та проаналізовано переваги даної мережі, оскільки саме ця технологія в перспективах глобального розвитку операторів мобільного зв’язку визначена як основа транспортної мережі. В [3] запропоновано методику моделювання цифрового коректора міжсимвольних спотворень сигналів в каналі передачі інформації, визначення оптимальних значень вагових коефіцієнтів коректора при проходженні послідовності випадкових двійкових сигналів, перевірку ефективності роботи шляхом визначення числа помилок демодуляції при спотворенні сигналу шумами. В [4 Проведено аналіз алгоритму, на основі якого автоматично синтезується імітаційно-аналітична модель гібридного програмно-конфігурованого пристрою.. В [5] дослідженні розглянуті проблеми розвитку програмно-конфігурованих мереж (ПКМ) та акцентовано на гібридному програмно-конфігурованому обладнанні (ПКО). Актуальність дослідження підкреслена в контексті сучасних вимог до мережевих технологій та визначених труднощів у моделюванні таких пристроїв. В [6] подано один із напрямків підвищення ефективності сучасних телекомунікаційних систем є перехід до використання багатовимірних сигналів для безперервних каналів передачі інформації.

Апробація результатів дисертації

Основні положення і результати дисертації, практичні висновки і рекомендації, які одержані в ході роботи, апробовані та оприлюднені в ході: XIII

наукової конференції студентства та молоді «Світ інформації та телекомунікації», (Київ, ДУТ, 21.10. 2021), регіонально міжнародна науково-технічна конференція МСЕ для країн Європи і СНД «Цифрова трансформація на основі інновацій у сфері ІКТ для розвитку цифрової економіки», (Київ, ДУТ, 15.12.2021)

Крім того, основні положення і результати дисертації, практичні висновки і рекомендації також апробовано на міжкафедральному семінарі Навчально-наукового інституту інформаційних технологій Державного університету інформаційно комунікаційних технологій.

Публікації

Основні наукові положення та результати дисертаційного дослідження опубліковано в 6 наукових працях. У томи числі: в 1 науковій статті у періодичному науковому виданні іноземної держави, яке індексується наукометричною базою «Scopus» в 5 наукових статтях у періодичних виданнях України включених до “Переліку наукових фахових видань України”.

Структура дисертації та її обсяг

Дисертаційна робота складається з анотації, змісту, переліку умовних скорочень вступу, чотирьох розділів, загальних висновків, списку використаних джерел має 125 сторінки основного тексту, 46 рисунки та 13 таблиць,. Список використаних джерел містить 104 найменувань і займає 12 сторінок. Загальний обсяг дисертаційної роботи – 147 сторінки.

РОЗДІЛ 1. ОСНОВНІ ПРИНЦИПИ ПОБУДОВИ ПРОГРАМНО – КОНФІГОВАНИЙ МЕРЕЖ

Метою даного розділу є визначення стратегії розвитку сучасних мереж, у т.ч. програмно-конфігурованих мереж з новим рівнем мережного управління, що дозволить забезпечити управління складною різнорідною мережею з максимальним рівнем автоматизації для забезпечення різних категорій споживачів сучасним набором послуг телекомунікацій від базових до високошвидкісних послуг мультимедіа.
Саме тому, у подальшому, досліджено принципи побудови системи управління програмно-конфігурованих мереж; визначено основні показники якості системи управління, які обумовлюють характеристику програмно- конфігурованих мереж в цілому та на базі яких доцільно реалізувати метод багатокритеріальної оптимізації.

1.1Архітектура програмно-конфігурованих мереж

Сучасний етап розвитку комп’ютерних систем можна охарактеризувати зростанням ролі розподілених мережних структур унаслідок об’єктивного зниження темпів зростання продуктивності окремих вузлів та необхідністю організації їх спільної роботи для досягнення потрібних характеристик. До прикладів цього процесу можна віднести розповсюдження багатоядерних процесорів, багатоканальну організацію оперативної пам’яті, ускладнення обчислювальних кластерів, центрів обробки даних та інше.

Організація потоків даних в таких структурах вимагає нових підходів до управління у зв’язку із лавиноподібним збільшенням комутаційних правил та трудомісткості управління мережною інфраструктурою [1]. Це пов’язано, з одного боку, з існуванням великої кількості різноманітних комунікаційних технологій, часто закритих та несумісних, з іншого боку існуюче мережне обладнання, як правило, спеціалізоване для типових локальних мереж, що не дозволяє реалізувати багатокритеріальне та багаторівневе управління в складних гетерогенних структурах.

Традиційний підхід до вирішення проблеми мережних взаємодій припускає

послідовну обробку одиниць передачі, зокрема пакетів, на кожному рівні еталонної моделі мережної взаємодії ISO/OSI: комутатори другого рівня покликані виконувати фільтрацію та пересилку кадрів між портами на основі MAC-адресів, VLAN та QoS, класичні маршрутизатори використовуються для об’єднання сегментів мережі на основі IP-адресів, шлюзи забезпечують трансляцію мережних адресів та контроль інформаційних потоків. На кожному рівні використовуються свої правила обробки (комутаційні матриці, таблиці маршрутизації, списки доступу та ін.) і різні формати даних. Сучасне мережне устаткування дозволяє організувати управління трафіком на основі інформації декількох рівнів, що дозволяє ефективніше обробляти трафік, але істотно ускладнює апаратуру, оскільки фактично призводить до поєднання декількох пристроїв в одному, при цьому алгоритми та методи комутації третього і четвертого рівнів є в переважній більшості закритими розробками виробників і мають обмежені виробником можливості налаштування.

Одним із напрямів ―модернізації‖ класичного підходу до організації мережної архітектури є створення програмно-конфігурованих мереж, на основі використання протоколу OpenFlow [2].

Динамічне розширення мереж та відповідної номенклатури послуг, що надаються ними, у операторів викликає гостру необхідність оптимізувати витрати на управління та адміністрування мереж. Одним з ефективних шляхів, що дозволяють значно зменшити експлуатаційні витрати, а також забезпечити безвідмовність обладнання, є застосування автоматизованих мережних систем управління (СУ).

Рівень існуючих СУ не відповідає повною мірою сучасним вимогам до них, не дає змогу отримувати інформацію потрібної якості для оперативного прийняття обґрунтованих рішень з управління об’єктами, обміну інформацією між суб’єктами СУ, а також не дає можливості оперативно управляти ситуаціями у мережах в автоматизованому режимі.

Суть концепції ПКМ зводиться до розділення процесів пересилки даних (forwarding) та управління потоками (control) [7]. Безпосередньо пересилку виконують спеціальні комутатори OpenFlow на основі універсальних таблиць потоків (ТП), які формуються та підтримуються зовнішнім управляючим

компонентом – контролером мережі [8]. Ключова особливість управління ПКМ в тому, що як ТП в комутаторах, так і правила функціонування самих контролерів доступні для управління зовнішніми програмними засобами через спеціалізовані API [104], як показано на рис. 1.1.

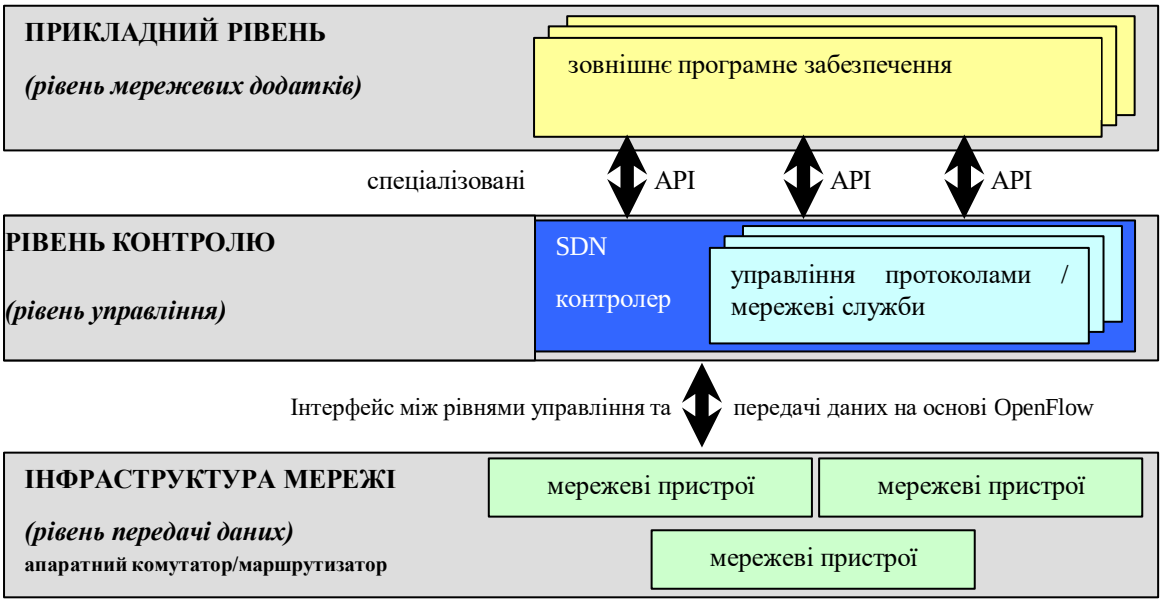


Рис. 1.1. Загальна архітектура програмно-конфігурованих мереж

В архітектурі SDN можна виділити три рівні (рис. 1.1):

- інфраструктурний рівень, що надає набір мережних пристроїв (комутаторів і каналів передачі даних);
- рівень управління, що включає мережну операційну систему, яка забезпечує мережні сервіси та програмний інтерфейс для управління мережними пристроями та мережею;
- прикладний рівень – для гнучкого та ефективного управління мережею – прикладні вирішення високорівневого управління мережею (засоби забезпечення безпеки, балансування навантаження програмного міжмережного екрану, адміністрування, виявлення вторгнень, а також функції інжинірингу трафіку, управління мобільністю, управління доступом, організації енергоефективної роботи мережі та ін.).

•

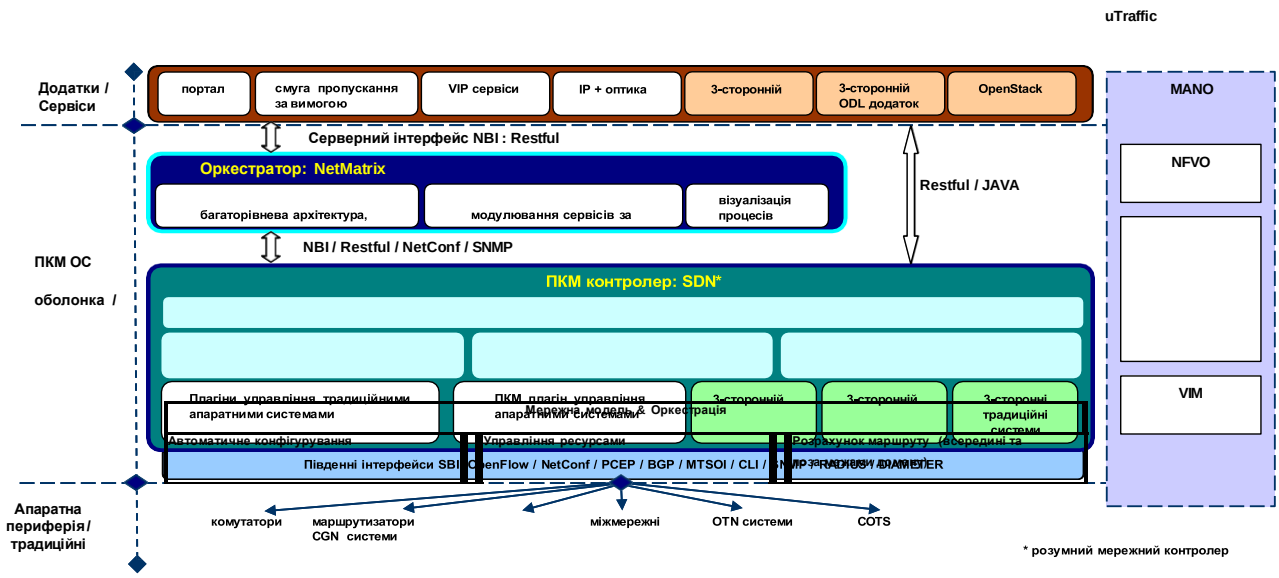


Рис. 1.2. Архітектура програмно-конфігурованих мереж

Виділення функції управління в окремий рівень та її делегування центральному компоненту (контролеру мережі, рис. 1.2) націлені на оптимізацію мережних конфігурацій для вирішення прикладних завдань і, зокрема, дозволяють отримати наступні переваги:

- формати даних, правила обробки та технології передачі в комутаторі не обмежені певним рівнем взаємодії, типом устаткування або виробником;
- одиниця передачі може бути задана багатовимірним вектором, що включає поля з різних рівнів моделі мережної взаємодії;
- при побудові правил пересилки пакетів можуть застосовуватися методи автоматизованої корекції потоків залежно від навантаження компонентів та інших критеріїв;
- контролери мережі можуть об’єднуватися в мережні домени, які дозволяють оптимізувати та резервувати канали передачі.

Інтелектуальні функції мережі зосереджені в централізованому мережному контролері, який відстежує загальний стан мережної інфраструктури і потоків, що протікають по ній. У такій концепції управління всією мережею відбувається в єдиній логічній точці, що значно спрощує завдання конфігурації та управління. Крім того, простіше виглядає і функціонування мережних пристроїв, тому що на відміну від традиційної моделі не потрібно більше підтримувати та обробляти безліч різних

протоколів, а достатньо тільки отримувати та обробляти інструкції від контролерів ПКМ. Для налаштування мережі досить набудувати програмний контролер мережі, замість того, щоб змінювати сотні рядків кодів в безлічі мережних пристроїв мережі. Також поведінку мережі можна змінювати в реальному часі, а нові рішення впроваджувати за набагато коротший час, ніж в традиційній архітектурі. Централізуючи стан мережі в єдиному рівні контролю, ПКМ можуть конфігуруватися за допомогою програмних засобів [9]. Мережні контролери також володіють набором прикладних інтерфейсів, які дозволяють реалізувати типові завдання по маршрутизації, зокрема багатоадресність, безпека, контроль доступу, управління смугою пропускання, якість обслуговування, які дуже вузьконаправлено налаштовані під завдання конкретного споживача.

У комутаторі реалізований тільки рівень передачі даних. Замість контролера використовується набагато простіший пристрій, завдання якого полягає в отриманні даних, які надходять, визначення їх адресів і, якщо адресат є в таблиці комутації, негайної передачі даних комутаційній матриці [10]. Інакше комутатор по захищеному каналу відправляє запит на центральний контролер мережі, і на підставі отриманої від нього інформації, вносить необхідні зміни в таблицю комутації, після чого здійснюється обробка отриманих даних (устаткування не переналаштовується вручну, а налаштовується за допомогою спеціального ПЗ).

Ідея ПКМ у створенні уніфікованого, незалежного від виробника мережного устаткування, програмно-керованого інтерфейсу між контролером та транспортним середовищем мережі знайшла віддзеркалення в протоколі OpenFlow, що дозволяє користувачам самим визначати і контролювати, хто з ким, за яких умов і з якою якістю може взаємодіяти в мережі [104]. До OpenFlow мережну архітектуру можна представити як телефонний зв’язок на самому початку XX століття, коли комутація здійснювалася вручну. Сьогодні адміністратор також вручну налаштовує устаткування за заданими параметрами, і будь-які подальші зміни здійснюються переважно на апаратному рівні. OpenFlow дозволяє відійти від такого управління мережею, що позитивно позначається на її масштабованості [11].

Розширена функціональність ПКМ визначається не тільки розподіленим способом управління, але і структурою таблиць потоків, що містять правила

пересилки трафіку. Як показано на рис. 1.3, кожен запис в ТП містить шаблон пакету (поля заголовків), статистичні лічильники та набір інструкцій (дій).

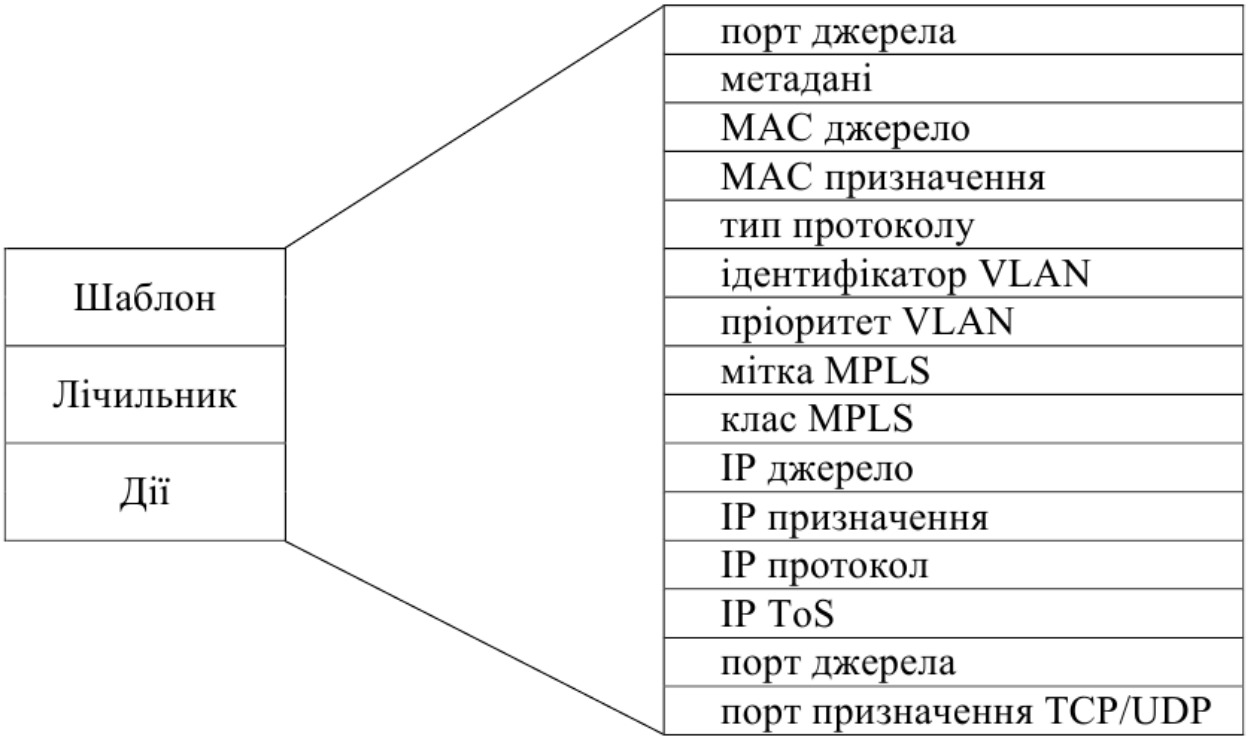


Рис. 1.3. Структура записи таблиці протоколів

Комутатор OpenFlow послідовно порівнює вміст передаваємого кадру із записами ТП і при збігу виконує вказані в записі дії. Якщо збіг не знайдений, то залежно від налаштувань комутатору пакет може бути відкинутий або відправлений OpenFlow-запрос контролеру для ухвалення рішення. Контролер може додавати, модифікувати, видаляти записи з таблиць як на основі аналізу пакетів, що отримуються від мережного устаткування, так і виходячи з власних алгоритмів роботи.

Специфікація OpenFlow визначає різноманітний набір інструкцій для обробки одиниць передачі, серед яких пересилка до порту(ів) або контролеру (Output), зміна полів (Set-Field, Change-TTL), робота з тегами IEEE 802.1Q та MPLS (Push-Tag/Pop-Tag), групові операції (Group) та ін. Також існує специфікація гібридного комутатора з підтримкою як OpenFlow-операцій, так і класичної комутації [8].

За допомогою протоколу OpenFlow фахівцям потрібно буде витрачати менше часу на початкові налаштування мережі та передачі її в експлуатацію в практично автоматичному режимі для підтримки заданого QoS. Всі маршрутизатори та комутатори об’єднуються під управлінням Мережної Операційної Системи (МОС),

яка забезпечує програмним забезпеченням доступ до управління мережею, і яка постійно відстежує конфігурацію засобів мережі. Потік даних flow таблиць OpenFlow (OpenFlow pipeline) містить безліч flow- таблиць, які можна об'єднати в деревовидні структури, що дозволяє ефективніше використовувати асоціативну пам'ять TCAM та реалізувати практично будь-які сценарії обробки і просування даних. Процес просування таблиць завжди починається з першої таблиці з номером "0", а закінчується, якщо немає вказівки на наступну flow-таблицю з вищим номером. Деревовидна структура (1.1) будується за допомогою вказівки виконання номерів наступних таблиць та виконання функцій цих таблиць:

$$f_{Tn+1}=(f_{Tn-1}, \ldots) + f_{Tn}; \quad (1.1)$$

Протокол OpenFlow вирішує також проблему залежності від мережного устаткування якого-небудь конкретного постачальника, оскільки ПКМ використовує загальні абстракції для пересилки пакетів, які МОС використовує для управління мережними комутаторами. Можна сказати, що OpenFlow – це протокол нижнього рівня для програмування комутаторів.

Теоретично SDN дає можливість абсолютної гнучкості в управлінні трафіком, теоретично – легке балансування трафіку без залучення окремого приладу (устаткування). На практиці, у SDN є три великі проблеми, які вирішують всі розробники:

- транспорт від контролера до комутаторів;
- шляхи стикування з традиційною мережею;
- безпека мережі.

Основні вимоги до SDN, запропоновані консорціумом ONF [13]:

1. Централізоване управління для мультівендорного устаткування.
2. Зниження складності налаштувань та конфігурації мережі за рахунок автоматизації настройки та конфігурації.
3. Високий рівень змінності в реальному часі для підтримки нових комерційних вимог.
4. Посилення безпеки та стабільності роботи мережі.
5. Детальний контроль мережі для служб підтримки сервісів.
6. Збір, обробка статистики мережі та управління.

За допомогою сучасних маршрутизаторів зазвичай вирішуються два основних завдання: передача даних (forwarding) – просування пакету від вхідного порту на певний вихідний порт та управління даними – обробка пакету та ухвалення рішення про тих, куди його передавати далі, на основі потокового стану маршрутизатору. Це відповідає рівню передачі даних, на якому зібрані засоби передачі (лінії зв'язку, канало-утворюючі устаткування, маршрутизатори, комутатори) і рівню управління станами засобів передачі даних (рис. 1.4).

Розвиток маршрутизаторів до цих пір йшов по шляху зближення цих рівнів, проте з нахилом на передачу (апаратне прискорення, вдосконалення програмного забезпечення та впровадження нових функціональних можливостей для збільшення швидкості прийняття рішення з маршрутизації кожного пакету), тоді як рівень управління залишався достатньо примітивним та спирався на складні розподілені алгоритми маршрутизації та хитромудрі інструкції з конфігурації та налаштування мережі.

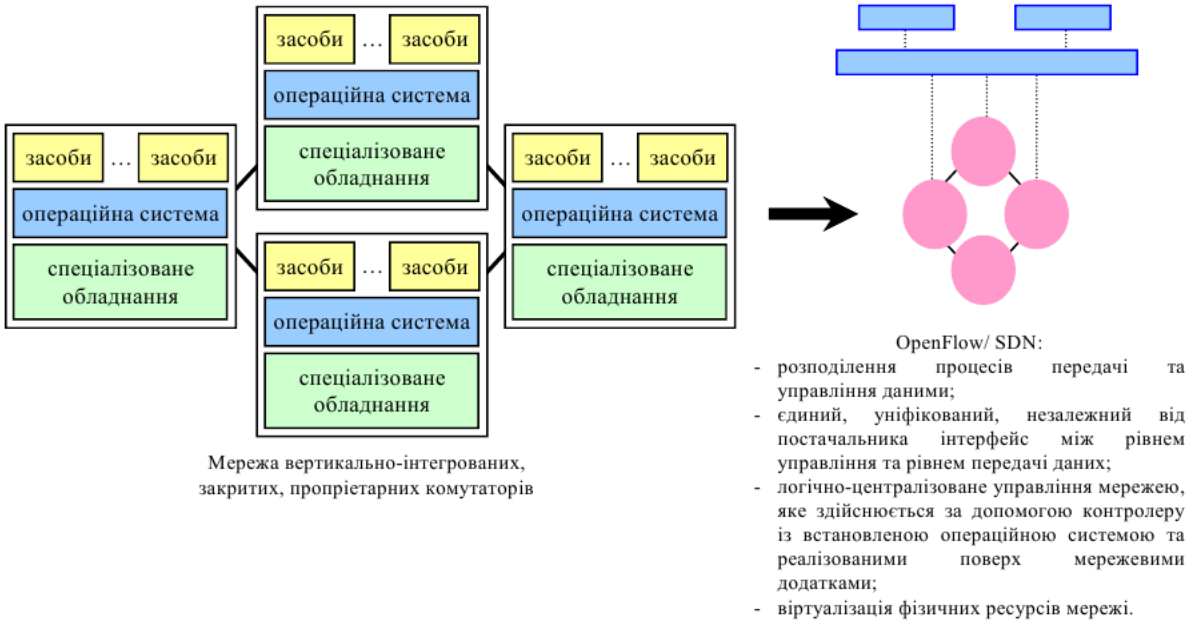


Рис. 1.4. Традиційні мережі та SDN

Логічно-централізоване управління даними в мережі припускає винесення всіх функцій управління мережею на окремий фізичний сервер, контролер, який знаходиться у веденні адміністратора мережі. Контролер може управляти як одним,

так і декількома OpenFlow-комутаторами та містить мережну операційну систему, що надає мережні сервіси по низькорівневому управлінню мережею, сегментами мережі та станом мережних елементів, а також додатки, що здійснюють високорівневе управління мережею та потоками даних.

В основі концепції SDN лежить відкритий стандарт OpenFlow – стандарт, визначений громадською організацією Open Networking Foundation (ONF, в 2011 році [14]. Даний фонд утворений альянсом компаній Deutsche Telekom, Facebook, Google, Microsoft, Verizon та Yahoo, націлений на просування і стандартизацію мереж, що програмно-конфігуруються. Про значущість нового руху говорить швидкорослий склад ONF, до якої увійшли вже більше 40 найбільших компаній миру, в числі яких Brocade, Cisco, Citrix, Oracle, Dell, Ericsson, HP, IBM, Juniper, Marvell, NEC, Netgear, NTT, Riverbed і ряд інших.

Інтерфейс OpenFlow надає доступ та зв’язок між рівнями управління та інфраструктури архітектури SDN як фізичною, так і віртуальною. OpenFlow використовує термін контролер, який ґрунтуючись на загальній конфігурації та інформації у всій мережі, передає по захищеному каналу команди для кожного маршрутизатора (ПКМ комутатора) окремо для інтерпретації їх в індивідуальних таблиць рішень.

Центральний контролер має точну інформацію про структуру та топологію мережі. Це дозволяє оптимізувати просування пакетів даних, і, зокрема, прокладати зв’язки “кожен з кожним” на рівні L2, не удаючись до IP- маршрутизації (рис. 1.5).

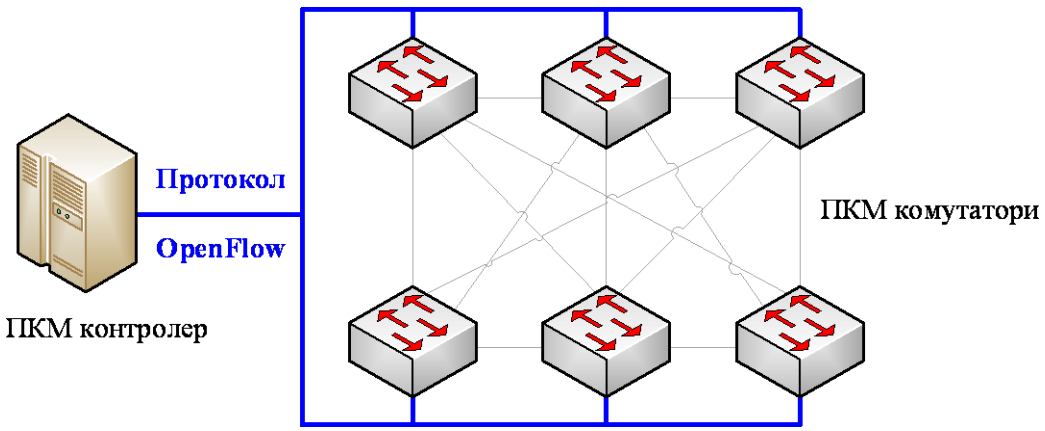


Рис. 1.5. Топологія SDN-мереж на основі OpenFlow

У ПКМ комутаторі з підтримкою OpenFlow реалізований тільки рівень передачі даних. У кожного комутатора своя унікальна таблиця, яку він заповнює тільки на підставі інформації, отриманої від центрального контролера. Така таблиця комутації отримала назву FlowTable (таблиця потоків), оскільки по ПКМ передаються потоки даних, а не окремі пакети (правило в комутаторі встановлюється тільки для першого пакету, а потім решта всіх пакетів потоку його використовує). За допомогою таких таблиць вхідні пакети класифікуються, ґрунтуючись на порту, MAC-адресі, IP-адресі та інших засобах [15].

У кожного отриманого пакету “вирізується” заголовок (бітовий рядок певної довжини). Для цього бітового рядка в таблицях потоків, починаючи з першого, визначається правило, до якого поля ознак щонайближче відповідає (співпадає) заголовок пакету. За наявності збігу, над пакетом та його заголовком виконуються перетворення, визначені набором інструкцій, вказаних в знайденому правилі. Запис про потік може вказати переслати пакет до певного порту (звичайний фізичний порт або віртуальний, призначений комутатором, або зарезервований віртуальний порт, встановлений специфікацією протоколу). Зарезервовані віртуальні порти визначають загальні дії пересилки: відправка контролеру, широкомовна (лавинна) розсилка, пересилка без OpenFlow. Віртуальні порти можуть точно визначати групи агрегації каналів, тунелі або інтерфейси із зворотним зв’язком.

Якщо потрібного правила в першій таблиці не виявлено, то пакет інкапсулюється та відправляється контролеру, який формує відповідне правило для пакетів даного типу та встановлює його на комутаторі (або на наборі керованих ним комутаторів), або пакет може бути змінений або скинутий. Інструкції конвеєра обробки дозволяють пересилати пакети в подальші таблиці для подальшої обробки і у вигляді метаданих передавати інформацію між таблицями. Інструкції також визначають правила модифікації лічильників, які можуть бути використані для збору різноманітної статистики (рис. 1.6).

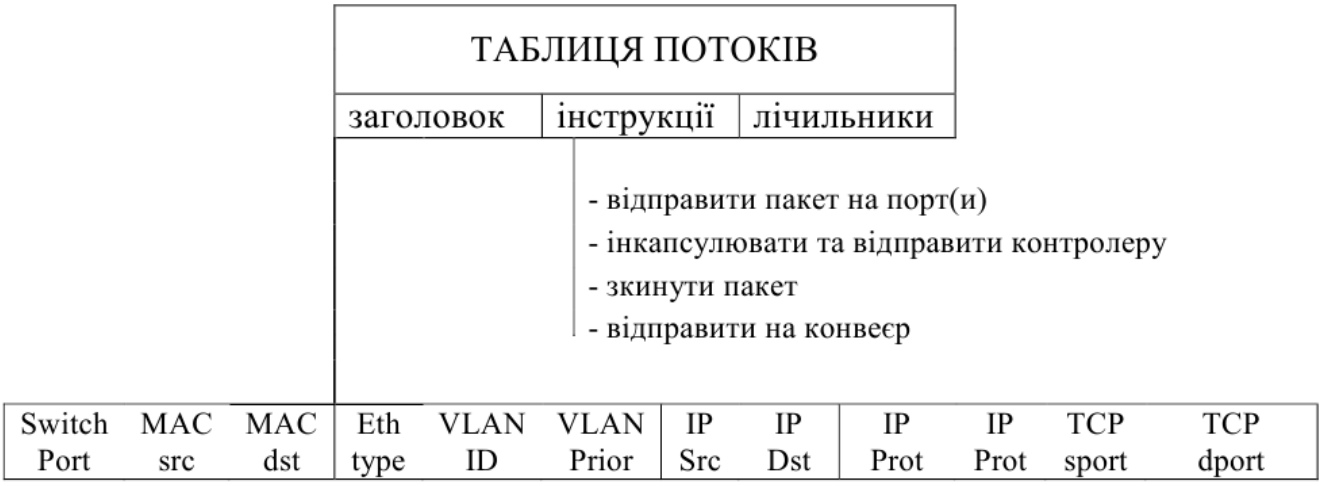


Рис. 1.6. Таблиця потоків в OpenFlow-комутаторі

Традиційні мережі можуть вибирати маршрути, ґрунтуючись на швидкості каналів, латентності або резервних потужностях, але при цьому їм важко ефективно справлятися з будь-якою реальною деталізацією (навіть використовуючи MPLS TE – Multiprotocol Label Switching Traffic Engineering). OpenFlow, навпаки, дозволяє запрограмувати мережу для принципово різних оптимізацій на основі кожного потоку. Це означає, що високочутливий до затримок трафік може використовувати найшвидший шлях, тоді як об’ємні потоки можуть йти по “найдешевших” маршрутах. Замість того щоб ґрунтуватися на конкретних кінцевих пристроях, OpenFlow дає можливість використовувати для класифікації широкі межі, аж до типу трафіку, що поступає від кожної кінцевої точки.

Протокол OpenFlow не зупиняється тільки на управлінні трафіком. Оскільки сумісний з OpenFlow пристрій здатний також переписувати пакети, він може діяти як NAT або AFTR (Address Family Transition Router). Оскільки він може відкидати пакети, пристрій здатний діяти як брандмауер. Він також може реалізувати ECMP (equal-cost multi-path) або інші алгоритми балансування навантаження. Маршрутизатори можуть мати різні правила для різних потоків, що проходять через них. Вони можуть розподіляти навантаження для одного потоку, використовувати брандмауер для іншого та змінювати заголовки пакетів для третього потоку. Таким чином, окремі елементи мережі стають значно гнучкішими в умовах OpenFlow у порівнянні з традиційною мережею.

OpenFlow призначений для використання в межах однієї організації. Всі маршрутизатори в домені OpenFlow діють як єдине ціле. Контролер має владу над

всіма маршрутизаторами, які він контролює. Інтернет-провайдер може використовувати OpenFlow, щоб контролювати свої маршрутизатори, але не може контролювати мережі клієнтів. Підприємство може використовувати один OpenFlow домен для управління мережею усередині великого центру обробки даних, а так само інший домен OpenFlow для управління доступу до глобальної комп'ютерної мережі.

Одна з ідей, що активно розвивається в рамках SDN, – це віртуалізація мереж з метою ефективнішого використання мережних ресурсів (рис. 1.7).

Під віртуалізацією мережі розуміється ізоляція мережного трафіку – групування (мультиплексування) декількох потоків даних з різними характеристиками в рамках однієї логічної мережі, яка може розділяти єдину фізичну мережу з іншими логічними мережами або мережними зрізами (network slices). Кожен такий зріз може використовувати свою адресацію, свої алгоритми маршрутизації, управління якістю сервісів, тощо.

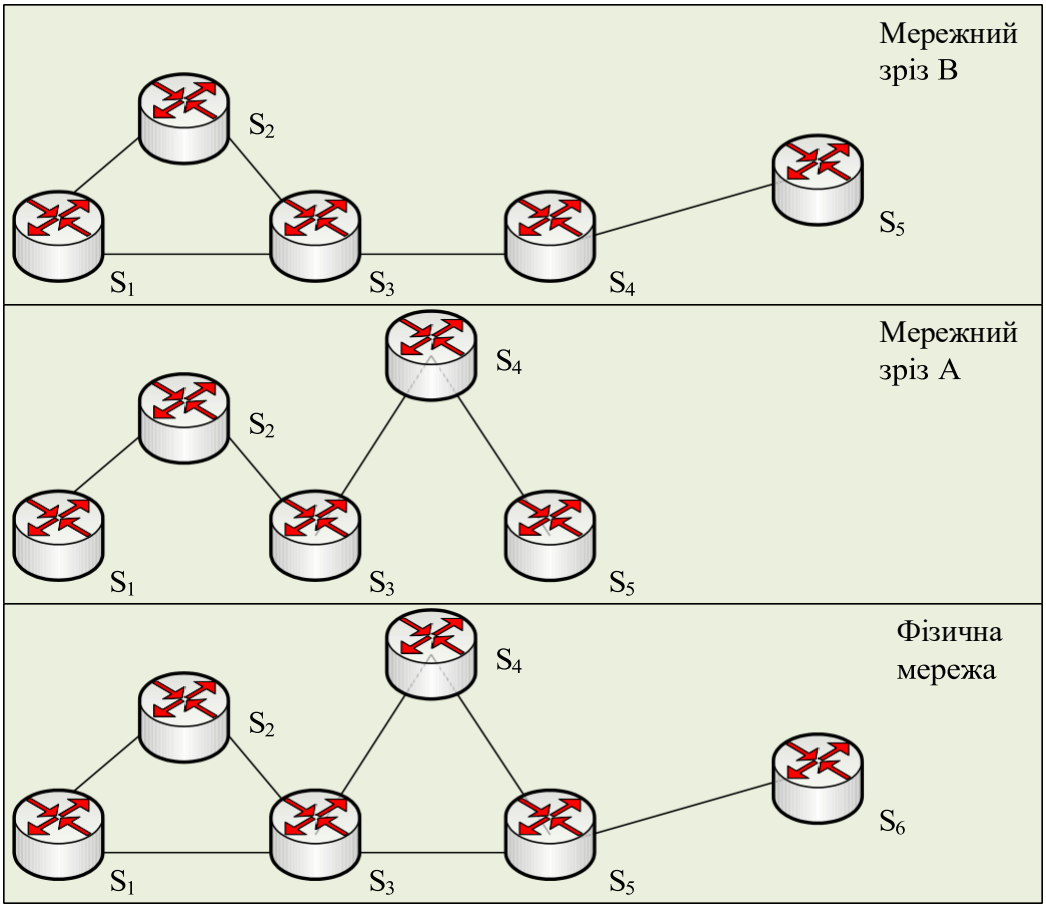


Рис. 1.7. Віртуалізація в SDN

Віртуалізація мережі дозволяє:

- підвищити ефективність розподілу мережних ресурсів та збалансувати навантаження на них;
- ізолювати потоки різних користувачів і додатків в рамках однієї фізичної мережі;
- адміністраторам різних зрізів використовувати свої політики маршрутизації і правила управління потоками даних;
- проводити експерименти в мережі, використовуючи реальну фізичну мережну інфраструктуру;
- використовувати в кожному зрізі тільки ті сервіси, які необхідні конкретним додаткам.

Одним з прикладів віртуалізації ресурсів SDN, розділення мережі на зрізи та управління ними є FlowVisor – програма-посередник (proxy), що діє на рівні між OpenFlow-комутаторами та різними контролерами SDN. За допомогою FlowVisor можна створювати логічні сегменти мережі, що використовують різні алгоритми управління потоками даних, забезпечуючи ізоляцію даних мереж один від одного. Це означає, що кожен контролер управляє тільки своєю логічною мережею і не може робити вплив на функціонування інших. Для контролера, що взаємодіє з устаткуванням OpenFlow через FlowVisor, весь обмін повідомленнями виглядає так само, якби контролер взаємодіяв із звичайною мережею SDN. Всю необхідну модифікацію повідомлень, потрібну для підтримки різних ізольованих сегментів мережі, виконує FlowVisor. Тобто для контролера логічної мережі не вимагається модифікації – це може бути будь-який контролер SDN, наприклад мережева операційна система NOX з довільним набором програм.

1.2 Стандартизація програмно-конфігурованих мереж

Розробкою стандартів SDN і дослідженнями в цьому напрямі в даний час займається велике число галузевих об’єднань та міжнародних організацій, в яких беруть участь зацікавлені комерційні компанії – виробники мережного устаткування, оператори мереж зв’язку, розробники ПЗ.

Концепція SDN активно просувається, проте її розуміння ще не устоялося, а ключові стандарти знаходяться на різних стадіях розробки та апробації, тому сьогодні загальноприйняте визначення поняття SDN відсутнє. Так, галузеві стандартизуючи організації: Фонд відкритих мережних технологій (англ. Open Networking Foundation, ONF), Робоча група по інженерних завданнях Інтернет (англ. Internet Engineering Task Force, IETF) та MCE-T (Сектор стандартизації Міжнародного союзу електрозв'язку) – пропонують наступні визначення:

- ONF: SDN – динамічна, керована та здатна до адаптації мережна архітектура, в якій розділені рівні управління мережею та передачі даних, що забезпечує програмне управління мережею та абстрагування/ізоляцію (рівня) мережної інфраструктури від (рівня) додатків та мережних послуг/сервісів [16].
- IETF: SDN – підхід до побудови мереж, що забезпечує пряме управління ресурсами та мережами, а також їх розподіл за рахунок додавання власних засобів обробки, адміністрування та програмного управління за допомогою відкритих мережних інтерфейсів і абстракції (абстрагування, ізоляції) рівня мережі [17].
- MCE-T: SDN – технологія побудови мереж, яка дозволяє реалізувати централізований, програмований рівень управління та ізоляцію (абстракцію) рівня даних; при цьому рівні управління та даних розділені, завдяки чому оператори мереж зв'язку можуть безпосередньо управляти своїми віртуальними ресурсами і мережами. Програмований рівень управління – рівень управління, який повинен бути програмованим і керованим централізованим чином; ізоляція (абстракція) рівня даних – моделі рівня даних повинні бути абстрактними і спрощеними, а не спеціалізованим апаратним забезпеченням [18].

Дослідженням загальних питань і стандартизацією SDN займаються ONF, IETF, Дослідницька група інтернет-технологій (англ. Internet Research Task Force, IRTF), Європейський інститут по стандартизації в області телекомунікацій (англ. European Telecommunications Standards Institute, ETSI) та MCE-T. Представники Форуму широкосмугових мереж (англ. Broadband Forum, BBF) вивчають деякі питання побудови та експлуатації мереж SDN. В рамках робочої групи "Інноваційні послуги

та ринкові вимоги” (англ. Service Innovation & Market Requirements) виділений проект SD-313 – комерційні вимоги та структура SDN в широкосмугових телекомунікаційних мережах (англ. Business Requirements and Framework for SDN in Telecommunication Broadband Networks). Проект припускає проведення досліджень в області сценаріїв переходу до мереж SDN, включаючи варіанти підтримки SDN частиною устаткування, а також впровадження функціональності SDN в устаткування при оновленні ПЗ.

Приватні питання стосовно SDN розглядаються і учасниками Форуму оптичної міжмережної взаємодії (англ. Optical Internetworking Forum, OIF). Ця некомерційна організація, яка розробляє угоди по реалізації (англ. Implementation Agreement, IA) для устаткування оптичних мереж, оцінює концепцію SDN як перспективну і займається розробкою вимог до SDN в частині транспортних мереж з боку операторів (оптичних) мереж і постачальників послуг, структури SDN та її співвідношення з архітектурою оптичних мереж з автоматичною комутацією (англ. Automatically Switched Optical Network, ASON), а також демонстрацією і тестуванням SDN.

Вимоги OIF до SDN з погляду операторів транспортних мереж включають:
функції рівня управління транспортної мережі SDN;
функціональні вимоги до оркестратора, що координує функції та служби на різних рівнях транспортної мережі і з кінця в кінець в суміжних доменах SDN;
співвідношення рівнів SDN в транспортній мережі – рівня передачі даних, управління, адміністрування та оркестровки.

Апробацією стандартів, що розробляються, та розробкою відкритих рішень SDN займаються декілька організацій:

- OpenDaylight – об’єднання галузевих виробників, включаючи IBM, Juniper Networks, Cisco, Red Hat, VMware, Citrix, Ericsson, Microsoft, NEC, Big Switch Networks, Brocade Communications Systems. Метою проекту є створення єдиної відкритої платформи SDN, розробка відкритої структури класів (фреймворка) в якості основа для створення готових продуктів та сервісів різними учасниками ринку;
- Open vSwitch – проект з розробки програмного комутатору SDN з відкритим

початковим кодом для застосування у віртуалізованому мережному середовищі. У проекті беруть участь компанії Citrix, Red Hat, Canonical, Oracle, FREEBSD Foundation, Nicira;

- OpenStack – проект з розробки комплексу безкоштовних відкритих програмних засобів для створення хмарної обчислювальної інфраструктури та сховищ даних, що включає підтримку SDN. У проекті представлені компанії AT&T, AMD, Brocade Communications Systems, Canonical, Cisco, Dell, EMC, Ericsson, Groupe Bull, HP, IBM, Inktank, Intel, NEC, Rackspace Hosting, Red Hat, SUSE Linux, VMware, Yahoo!.

Узагальнена схема основних напрямів стандартизації та розробки представлена на рис. 1.8 [21], де сірими овалами показані стандартизуючі організації, а білими – об’єднання галузевих виробників та проекти з розробки відкритого ПЗ.

ONF, як родоначальник SDN, ставить за мету формування нової архітектури мережі та устаткування, що припускає відділення площини управління від площини передачі, і докладає значні зусилля до подолання виникаючих проблем, пов’язаних із складнощами міграції від традиційних мереж до SDN і з труднощами стандартизації площини передачі із-за перешкод з боку провідних світових виробників устаткування.

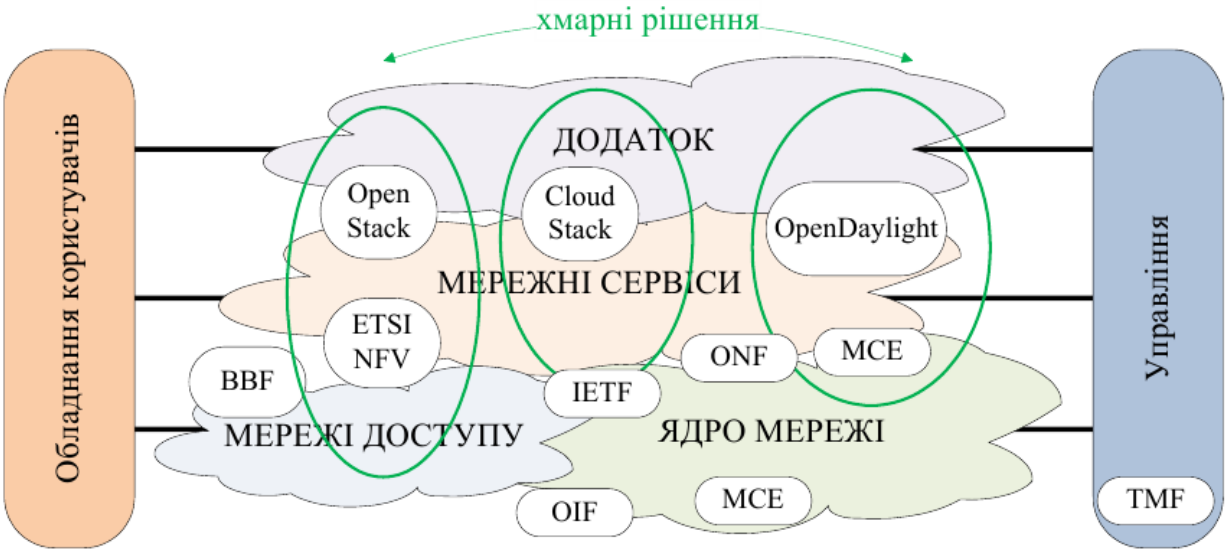


Рис. 1.8. Розподіл організацій, які займаються стандартизацією, за напрямками розробки архітектури SDN

IETF слідує убік SDN іншим шляхом, додаючи властивості відкритих інтерфейсів в існуючу архітектуру та захищаючи, таким чином, вкладені інвестиції. Недоліками при цьому є обмежена відкритість та значне ускладнення мережі, оскільки устаткування повинне підтримувати старі та нові технічно/технологічні рішення.

ETSI, маючи спеціалізовану групу ISG для стандартизації NFV, зосереджує значні зусилля на доказовій базі спроможності концепції. При цьому відома проблема продуктивності функціонування визначає реалізацію мережних функцій на промислових серверах, хоча створення спеціалізованого устаткування для NFV також актуально.

Метою OpenStack є створення хмарної операційної системи для оркестровки обчислень, зберігання, реалізації мережних функцій. У цьому напрямі досягнутий значний прогрес, проте потрібний час для того, щоб перетворити цю систему на готове, зокрема в комерційному плані, рішення [102].

Таким чином, створення ПКМ дозволить відокремити рівень управління мережним устаткуванням від рівня управління передачі даних, створити програмно-керований інтерфейс між мережними додатками та транспортним середовищем та перейти від управління окремим устаткуванням до управління мережею в цілому.

Для тестування додатків контролера можна використовувати: фізичне устаткування, віртуальне устаткування (емуляція), моделювання. Перший підхід має високий ступень довіри, але при цьому є украй не зручним при експериментуванні (зміни топології, кодів, налаштувань). Емуляція покриває цей недолік і при цьому дозволяє заощадити на покупці устаткування. Але через те, що кожен пристрій представляє віртуальну машину, при великих мережних топологіях потрібні значні обчислювальні потужності. Найоптимальнішим інструментом для тестування є моделювання, яке використовує низьку вимогу до ресурсів.

Віртуальні мережі повинні бути:

- гнучкими, здатними підтримувати різні топології,
- архітектури маршрутизації та переадресації, а також незалежні конфігурації;
- керованими, щоб відокремлювати політику, яку намагається нав’язати мережний оператор, від тих механізмів, як ця політика реалізується;

- програмованими;
- масштабованими, для збільшення кількості співіснуючих віртуальних мереж;
- захищеними, шляхом виділення різних логічних мереж один від одного; гетерогенними, для підтримки різних технологій.

Віртуальні мережі складаються з двох компонентів: вузлів та ребер. Самі фізичні вузли повинні бути віртуалізованими. Один з можливих способів віртуалізації вузла представляє собою віртуальну машину. Цей спосіб віртуалізації вузла використовує віртуальне середовище, таке як VServer або Jail. Гіпервізор або інша технологія дозволяє віртуальному середовищу ефективно надавати фізичне устаткування, щоб забезпечити ілюзію декількох гостьових вузлів. Приклади віртуалізації вузла: середовище віртуальної машини, таке як Xen або VMware; віртуалізація на рівні ОС або віртуального середовища, такого як Linux vServer. [20] У віртуальній мережі необхідно з'єднати ці віртуальні машини. Кожна віртуальна машина або віртуальне середовище має свій власний вид мережного стека. Потрібно забезпечити видимість того, що ці вузли сполучені один з одним через 2-й рівень OSI, навіть якщо вони насправді розділені декількома IP-мережами. Один з можливих способів зробити це полягає в інкапсуляції Ethernet-кадру коли він покидає межі VM в IP-пакеті. У пункті призначення IP-пакет піддається де-інкапсуляції та передає початковий Ethernet-кадр до ВМ або віртуального середовища. Кожен з цих фізичних хостів, може насправді, розмістити декілька віртуальних машин або віртуальних середовищ, що створює необхідність у віртуальному комутаторі, який знаходиться на фізичному вузлі. Цей віртуальний комутатор об'єднує у віртуальну мережу 2-го рівня OSI все ВМ. [10] Отже, Linux-мост, Open vSwitch є прикладами програмних комутаторів, які можуть виконувати цей тип функції.

Починаючи з версії 2.6.24 ядром Linux підтримуються механізми віртуалізації та ізоляції – Cgroups, які дозволяють забезпечити мережними інтерфейсами, таблицями маршрутизації та ARP-таблицями процеси в рамках однієї операційної системи. Це один з видів віртуалізації на рівні ОС, що дозволяє запустити безліч однотипних процесів в ізольованому і обмеженому за ресурсами оточенні. Дану технологію використовує середовище для моделювання MiniNet, яка дозволяє

створити віртуальну мережу на малопотужному устаткуванні. [22] При запуску MiniNet за допомогою сценарію “mn”, кожен хост у віртуальній мережі є bash-процес з власною мережею простору імен, фактично це віртуалізація на рівні ОС (рис. 1.9).

Так, кожен з цих віртуальних вузлів, має свій власний мережний стек. Кореневий простір імен управляє зв’язком між різними віртуальними вузлами, а також виконує роль комутатора, який сполучає ці вузли в створену розробником топологію.

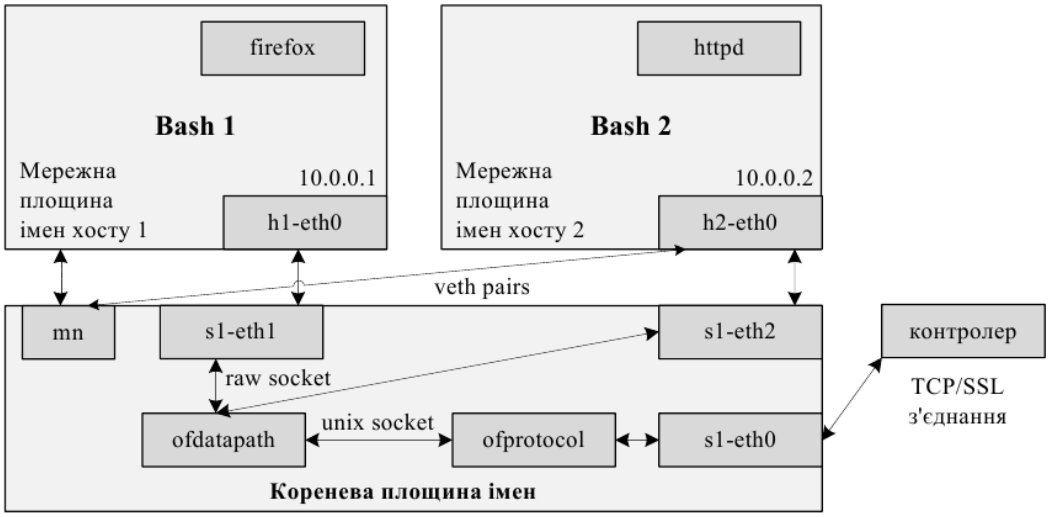


Рис. 1.9. Архітектура MiniNet

Віртуальним парам Ethernet призначено два простори імен. Наприклад, s1 ethl призначається інтерфейс в мережному просторі h2, s1 eth2 привласнюється мережному простору імен h3. Комутатор OpenFlow ефективно виконує переадресацію між інтерфейсами в кореневому просторі імен. Але оскільки інтерфейси є парними, ми отримуємо ілюзію відправки трафіку між h2 та h3. [23] Коли ми вносимо зміни в комутаторі OpenFlow через контролер, ми фактично це робимо в кореневому просторі імен. Подібна техніка дозволяє MiniNet створювати в просторі ядра або користувача комутатори, OpenFlow-контролери та хости, що дозваляє взаємодіяти в рамках модельованої мережі. Як віртуальні комутатори використовується адаптована реалізація Open VSwitch. [20]

Відзначимо потенційні достоїнства архітектури SDN-мереж щодо традиційних сучасних мереж.

Глибока інтеграція. Кожен веб-сервіс може направити вимоги до пропускної

здатності до контролера, який відповідає за задоволення запиту.

Зменшення вартості розгортання мереж. Проектування та виготовлення пристрою з фіксованою конфігурацією, яка не вимагає оновлень, менш коштовне.
Використання простіших алгоритмів. Замість того щоб ухвалювати рішення на основі виводу і спиратися на алгоритми взаємодії між пристроями, можуть бути використані більш безпосередні алгоритми.

Можливість розробки та розвитку мережних програмних модулів. ПКМ-контролер має інтерфейси API, які можуть використовуватися додатками.
Зручність адміністрування та налаштування.

Глобальна оптимізація та планування. Контролер посилається на глобальне уявлення про мережу, тим самим, використання мережних ресурсів може стати раціональнішим, а також масштабування мережі простіше.
Відкритість протоколу OpenFlow. Це дозволяє не залежати від виробників мережних пристроїв.

Досліджувана інфокомунікаційна мережа має N джерел, M комунікаційних пунктів і L адресатів. Всі пункти мережі зв'язані каналами зв'язку. Передбачається, що кожен канал зв'язку характеризується надійністю проведення повідомлення і часом проведення повідомлення. Вважається, що ці два параметри зв'язано функціональною залежністю, вигляд якої не деталізований. Всі джерела можуть служити комунікаційними пунктами.

Основними завданнями в даному випадку є: управління мережею, що включають оптимальну комутацію каналів між даним джерелом і даним адресатом, яка забезпечує прийнятні ймовірнісні і часові характеристики проведення повідомлення; вибір характеристик мережі, котрі оптимізують проведення повідомлень від джерел до адресатів.

Дослідимо уточнені постановки завдань і можливі підходи до їх.

Ототожнимо дану інформаційну мережу з орієнтованим мультиграфом(G, M^*), поклавши для простоти $G = (0,1,\cdots,n)$ – сукупність вершин графа (пунктів мережі) і $M^* \subset G \times G$ – сукупність дуг мультиграфа (каналів зв'язку). У множині M^* присутні однотипні елементи, що визначають канали зв'язку між сусідніми пунктами.

Спочатку розглянемо простий випадок, коли в мережі є одне джерело (O) і один адресат (n). Як прийнято в теорії мереж, зручно замість множини M^* розглянути бінарні змінні $\left(x_{ij}^{(K_{ij})}\right)_{i,j=0}$, де K_{ij} - кількість каналів зв'язку, що сполучають i-тий і j-тий пункти мережі $x_{ij}^{(K_{ij})} = \{0 \text{ або } 1\}$.

Аналогічно, часи передачі повідомлень по каналах зв'язку можна охарактеризувати матрицею $\mathrm{T}^* = \left(t_{ij}\right)_{i,j=0}^{n}$ в якій елементи $t_{ij}^{(K_{ij})}$ вважаються рівними ∞, якщо i-тий і j-тий пункти є не сусідніми, тобто не мають між собою каналів зв'язку. Надійність передачі повідомлення каналами зв'язку може бути визначена випадковою матрицею $X^* = \left(X_{ij}^{(K_{ij})}\right)_{i,j=0}^{n}$, де елементи є незалежними біноміально розподіленими випадковими величинами з рядами розподілу

$X_{ij}^{(K_{ij})}$	0	1
P	$1-P_{ij}^{(K_{ij})}$	$P_{ij}^{(K_{ij})}$

де $P_{ij}^{(K_{ij})}$ – вірогідність правильної передачі повідомлення по K_{ij}-тому каналу зв'язку між сусідніми пунктами i і j.

Дослідимо тільки характеристики надійності передачі повідомлення і часу передачі повідомлення в мережі. Дещо спростимо модель, замінивши пучок каналів зв'язку між i-тим і j-тим сусідніми пунктами одним каналом з характеристиками: надійністю передачі повідомлення $X_{ij} = \min\limits_{K_{ij}} X_{ij}^{(K_{ij})}$ і часом передачі повідомлення $t_{ij} = \min\limits_{K_{ij}} t_{ij}^{(K_{ij})}\left(X_{ij}^{(K_{ij})}\right)^{-1}$. Простий випадок ми отримуємо, коли $t_{ij}^{(K_{ij})} = \tau_{ij}$ при фіксованих i і j. Неважко встановити закони розподілу випадкових величин X_{ij} і t_{ij}. Саме, $P_{ij} = P\{X_{ij} = 1\} = 1 - \prod\limits_{K_{ij}}\left(1 - P_{ij}^{(K_{ij})}\right)$, отже, випадкова величина X_{ij} має ряд розподілу

X_{ij}	0	1
P	$1-P_{ij}$	P_{ij}

Випадкова величина в простому випадку, має ряд розподілу

t_{ij}	τ_{ij}	∞
P	P_{ij}	$1-P_{ij}$

Загальний випадок відрізняється громіздкістю, тому не будемо розглядати. Дослідимо наступну модель як зразок мережі. Розглянемо орієнтований граф (G, M), $G=(0,1,\cdots,n); M \subset G \times G$, на якому визначено дві випадкові матриці $X=(X_{ij})_{i,j=0}^{n}$ і $T=(t_{ij})_{i,j=0}^{n}$ з елементами, визначеними вище.

1.3 Характеристики надійності програмно – конфігурованих мереж

Позначимо $\Phi_1(x), x=(x_{ij})_{i,j=0}^{n}$, структурну функцію мережі. $\Phi_1(x)$ – бінарна функція, що набуває значення 1, коли є хоч би один маршрут між джерелом O і адресатом n, і значення 0 в протилежному випадку. Суперпозиція структурної функції $\Phi_1(x)$ з випадковою матрицею X дає нам випадкову величину $\Phi_1(X)$, що характеризує надійність зв'язку між джерелом O і адресатом n. Саме.

$$P\{\Phi_1(X)=1\}=\mathrm{E}\Phi(X)=\Psi_1(P) \tag{1.2}$$

де $P=(P_{ij})_{ij=0}^{n}$- матриця, в якій елементи $P_{ij}=0$ для несуміжних вузлів i і j. Функцію $\Psi_1(P)$ прийнято називати функцією надійності мережі. Вона характеризує надійність зв'язку без врахування часових обмежень.

Перейдемо до обліку часових обмежень на передачу повідомлення. Для цього введемо функцію $\Phi_s(S), S=(S_{ij})_{i,j=0}^{n}$, де S_{ij}– невипадковий час проведення повідомлення між сусідніми пунктами i і j. Ця функція на фіксованій матриці S набуває значення, що дорівнює мінімальному сумарному часу проведення повідомлення від джерела O до адресата n по різних маршрутах. Суперпозиція функції $\Phi_2(S)$ з випадковою матрицею T, дає нам випадкову величину $\Phi_2(T)$, що характеризує час передачі повідомлення. Особливий інтерес для нас представляє функція розподілу випадкової величини $\Phi_2(T)$. Очевидно, що величина $\Phi_2(T)$ вироджена і її дефект прямо пов'язаний з функцією $\Psi_1(P)$. Саме

$$P\{\Phi_2(T)<\infty\}\equiv\Psi_1(P). \tag{1.3}$$

Якщо нас цікавить ситуація, коли повідомлення має бути передане за час не більший ніж (штатний час), то ми можемо розглянути функцію

$$\Psi_2(P,\tau,t_0)=P\{\Phi_2(T)<t_0\}, \tag{1.4}$$

де $P=(P_{ij})_{i,j=0}^{n}$, $\tau=(\tau_{ij})_{i,j=0}^{n}$.

Функції Φ_1, Φ_2, можуть грати роль цільової функції в різного роду оптимізаційних задачах.

З формальної точки зору обчислення значень функцій $\Psi_1(P)$, $\Psi_2(P,\tau,t_0)$ має бути наступним

$$\Psi_1(P)=\sum_y \Phi_1(y)\prod_{i,j} P_{ij}^{y_{ij}}(1-P_{ij})^{1-y_{ij}}\ ; \tag{1.5}$$

$$\Psi_2(P,\tau,t_0)=\sum_y 1_{(\Phi_2(y_1)<t_0)}\prod_{i,j} P_{ij}^{y_{ij}}(1-P_{ij})^{1-y_{ij}}\ , \tag{1.6}$$

де $y=(y_{ij})_{i,j=0}^{n}$, $y_{ij}=0$ або 1; $y_1=(y_{ij}^{(1)})_{i,j=0}^{n}$; $y_{ij}^{(1)}=\dfrac{\tau_{ij}}{y_{ij}}$;

1_A – індикаторна функція множини А, $O^0=1$.

Доцільним способом обчислення даних функцій є метод статистичного моделювання (метод Монте-Карло). Зміст методу надзвичайно простий. Матриця P, що несе на собі характеристику всієї випадковості в системі, замінюється випадковою матрицею π з незалежними елементами, які мають біноміальний розподіл з параметрами 1 і P_{ij} для відповідних величин. Маючи в своєму розпорядженні послідовність $\{\pi_K\}_{K=1}^{\infty}$ незалежних копій матриці π, ми приходимо до наступних граничних співвідношень

$$\frac{1}{N}\sum_{K=1}^{N}\Phi_1(\pi_K)\xrightarrow[N\to\infty]{P}\Psi_1(P) \tag{1.7}$$

$$\frac{1}{N}\sum_{K=1}^{N}1_{(\Phi_2(y_1(K))<t_0)}\xrightarrow[N\to\infty]{P}\Psi_2(P,\tau,t_0), \tag{1.8}$$

де $y_1(K)=(y_{ij}^{(1)}(K))_{i,j=0}^{n}$; $y_{ij}^{(1)}=\dfrac{\tau_{ij}}{\pi_k^{(ij)}}$; $\pi_K=(\pi_K^{(ij)})_{i,j=0}^{n}$.

Співвідношення (1.6), (1.7) виходять із закону великих чисел. Для отримання оцінок функцій Ψ_1, Ψ_2, за допомогою граничних співвідношень (1.6), (1.7) необхідно вирішити завдання оцінки точності.

Обидві функції Ψ_1, Ψ_2, представляють вірогідності, тому оцінка точності наближення з допомогою (1.6) і (1.7) зводиться до класичного завдання побудови довірчого інтервалу для невідомої вірогідності в біноміальному розподілі. Довірчий інтервал для вірогідності $q = \Psi_1, \Psi_2$, що використовує нормальну асимптотику добре відомий

і має наступний вигляд

$$\frac{N}{N+\varphi_\alpha^2}\left(q^* + \frac{\varphi_\alpha^2}{2N} \pm \varphi_\alpha \sqrt{\frac{q^*\left(1-q^*\right)}{N} + \frac{\varphi_\alpha^2}{4N^2}}\right) \tag{1.9}$$

де N – число реалізацій; φ_α – $\alpha \cdot 100\%$ – верхній квантиль для модуля стандартної нормальної випадкової величини; $q^* N$ число реалізацій з N, які дають позитивну відповідь; α – рівень значущості.

Просте застосування довірчого інтервалу полягає в тому, що по заданому рівню значущості α і поточному значенню q^* підібрати N (зупинити випробування) в той момент, коли вперше довжина довірчого інтервалу $\frac{2N\varphi_\alpha}{N+\varphi_\alpha^2}\sqrt{\frac{q^*\left(1-q^*\right)}{N} + \frac{\varphi_\alpha^2}{4N^2}}$ досягне необхідного рівня точності β. У простому випадку можна звільнитися від поточного значення q^* і обмежитися грубою оцінкою $\frac{N\varphi_\alpha}{N+\varphi_\alpha^2}\sqrt{\frac{1}{N} + \frac{\varphi_\alpha^2}{N^2}}$, замінивши величину $q^*\left(1-q^*\right)$ її максимальним значенням рівним 1/4. Останній вираз показує, що кількість випробувань для досягнення рівня точності β повинна мати порядок β^{-2} .

Точніші оцінки можна отримати, використовуючи поточне значення q^*, що обчислюється періодично. Скористаємося евристичним правилом. Довжина довірчого інтервалу обчислюється через кожних $\left[\frac{0.1}{\beta^2}\right]$ ітерацій. Зупинка обчислень проходить в той момент, коли вперше довжина довірчого інтервалу досягає необхідної величини.

Дане правило дозволяє дещо скоротити число реалізацій у визначенні величини цільової функції, зберігаючи в цілому порядок обчислень. Проте, в реальних мережах, де потрібна висока надійність передачі повідомлення від джерела до

адресата, можливий інший підхід, що базується на перевірці статистичних гіпотез і дозволяє різко скоротити число реалізацій в процедурі моделювання. Суть цього підходу полягає в наступному. Виберемо значення відповідної цільової функції рівним q_0, вважаючи його гранично низьким в наших умовах. Маючи в своєму розпорядженні N реалізацій випадкового графа, що відображає відмови каналів зв'язку, ми приходимо до наступного статистичної задачі. Є N незалежних біноміально розподілених випадкових величин $U_1(w),\ldots,U_N(w)$ з параметрами 1 і q $(0<q<1)$. Вірогідність q невідома. По єдиній реалізації $U_1(w_0),\ldots,U_N(w_0)$ потрібно перевірити статистичну гіпотезу $H_0: q=q_0$ проти альтернативи $H_1: q<q_0$. Як відомо із загальної теорії перевірки статистичних гіпотез, в нашому випадку існує достатня статистика $z=\sum_{k=1}^{N} U_k$, за допомогою якої легко побудувати найбільш потужну критичну область K^* для перевірки гіпотези H_0 проти альтернативи H_1. Область K^* має простий вид $K^*=\{0,1,2,\ldots,n_\alpha\}$, де число n_α визначається з умови $P_{q_0}\{z\le n_\alpha\}=\alpha$, де α – вірогідність помилки першого роду. Вибір необхідного обсягу вибірки N здійснюється виходячи з величини вірогідності помилки другого роду. Можливі різні плани досліджень, такі як - до першої появи фіксованого числа відмов або послідовні плани.

Відзначимо, що вказана критична область, як відомо, існує не для всіх значень вірогідності помилки першого роду α. Аби зняти це обмеження переходять або до нерівності, або до процедури рандомізації. Заропонована процедура не дає нам точного або навіть кількісного значення цільової функції, а відповідає на питання, чи задовольняє наша цільова функція нерівності $\Psi\ge q_0$. Відповідь на це питання проводиться скороченням числа ітерацій в методі Монте-Карло в порівнянні із звичайними методами зупинки.

Всі розглянуті попередні випадки це, коли в інфокомунікаційній мережі є одне джерело і один адресат. Для конкретної інфокомунікаційної мережі декількох джерел і адресатів, можливе проведення обчислень надійності мережі з одним з джерел, передбачаючи, що останні п'ять працюють як комунікаційні пункти. В цьому випадку можливі точні розрахунки по формулах (1.4) і (1.5). У складнішому випадку, коли необхідно враховувати всі адресати, формули (1.4), (1.5) доцільно використовувати

для розрахунків. Отже, перейдемо до загального випадку. Хай в даній інфокомунікаційній мережі виділено n_1 джерел і n_2 адресатів. Тоді по аналогії з попереднім випадком, якщо не враховувати часові обмеження, розглянемо матрицю $\overline{\Phi}_1 = \left(\Phi^{(k,l)}(x)\right)_{k=1}^{n}{}_{l=1}^{n}$, де $x = \left(x_{ij}\right)_{i,j=0}^{n}$ (x_{ij} – бінарні змінні), $\Phi_1^{(k,l)}(x)$ – структурні функції, що характеризують зв'язність джерела k з адресатом l для даної невипадкової конфігурації мережі x. Повертаючись до введеної вище випадкової матриці $X = \left(X_{ij}\right)_{i,j=0}^{n}$, що характеризує завади в каналах зв'язку, розглянемо випадкову матрицю $\overline{\Phi_1}(X)$, що є показником надійності зв'язку між джерелами і адресатами. Різні числові характеристики матриці $\overline{\Phi_1}(X)$ можуть служити критеріями якості функціонування інфокомунікаційної мережі. Наприклад, такими характеристиками можуть бути:

а) $E\overline{\Phi_1}(X) = \left(\Psi_1^{(k,l)}(P)\right)_{k=1}^{n_1}{}_{l=1}^{n_2}$ – матриця функцій надійності інфокомунікаційної мережі як функція надійності її каналів зв'язку;

б) $P\left\{\max_{k,l} \Phi_1^{(k,l)}(X) = 1\right\}$ – вірогідність наявності зв'язку між хоч би одним джерелом і хоч би одним адресатом;

в) $P\left\{\min_{k,l} \Phi_1^{(k,l)}(X) = 1\right\}$ - вірогідність наявності зв'язку між всіма джерелами і всіма адресатами.

Відзначимо, що для дослідження характеристики *б*) можна скористатися моделлю з одним джерелом і одним адресатом. Для цього досить додати одне фіктивне джерело, з'єднавши його абсолютно надійними каналами зв'язку з n_1 джерелами і, аналогічно, один фіктивний адресат, сполучений з n_2адресатами абсолютно надійними каналами зв'язку. Тоді функція надійності нової мережі співпадає з вірогідністю з пункту *б*). Для дослідження випадку *в*), а також інших складніших випадків, можна скористатися відповідними системами. Розглядаючи різні варіанти цільової функції в моделі з багатьма джерелами і адресатами, ми не враховували часові обмеження. По аналогії з моделлю одного джерела і одного адресата, розглянемо випадкову матрицю

$$\overline{\Phi_2}(T) = \left(\Phi_2^{(k,l)}(T)\right)_{k=1}^{n_1}{}_{l=1}^{n_2},$$

де $\Phi_2^{(k,l)}(T)$ – випадковий час проведення повідомлення від джерела k до адресата l. Введені вище характеристики *а*)-*в*) для моделі, що не враховує часових обмежень, можуть бути перенесені на даний випадок з використанням матриці $\overline{\Phi_2}(T)$.

а) $E\overline{\Phi_1}(X) = \left(P\left\{\Phi_2^{(k,l)}(T) < \infty\right\}\right)_{k=1\,l=1}^{n_1\quad n_2}$;

б) $P\left\{\max_{k,l} \Phi_1^{(k,l)}(X) = 1\right\} = P\left\{\min_{k,l} \Phi_2^{(k,l)}(T) < \infty\right\}$;

в) $P\left\{\min_{k,l} \Phi_1^{(k,l)}(X) = 1\right\} = P\left\{\max_{k,l} \Phi_2^{(k,l)}(T) < \infty\right\}$.

Крім того, до характеристик в разі обліку часових обмежень можуть бути додані наступні:

г) $\left(P\left\{\Phi_2^{(k,l)}(T) < t_0\right\}\right)_{k=1\,l=1}^{n_1\quad n_2}$ – матриця вірогідності своєчасної передачі повідомлення від джерел k до адресатів l;

д) $P\left\{\max_{k,l} \Phi_2^{(k,l)}(T) < t_0\right\}$– вірогідність того, що час передачі повідомлення хоч би від одного джерела до хоч би одного адресата не перевищує штатного часу t_0;

е) $P\left\{\min \Phi_2^{(k,l)}(T) < t_0\right\}$ - вірогідність того, що час передачі повідомлення від всіх джерел до всіх адресатів не перевершує штатного часу t_0.

Всі ці характеристики для мереж не дуже великій розмірності можуть бути розраховані по формулах, аналогічних формулах (1.4) (1.5). Проте, для мереж великої розмірності, так само як і в разі одного джерела і одного адресата, аналоги формул (1.4), (1.5) мало придатні для розрахунків. В цьому випадку основна надія покладається на метод статистичного моделювання з використанням моментів зупинки описаних вище. Цим способом можна отримати оцінки для функцій з пунктів би), у), д), е). Деякі складнощі технічного характеру виникають у випадках а) і г), де потрібно одночасно стежити за декількома параметрами. До опису відповідної процедури ми переходимо.

1.4 Організації управління програмно-конфігурованими мережами SDN

Організація потоків даних в телекомунікаційних структурах вимагає нових підходів до управління у зв’язку із лавиноподібним збільшення комутаційних

правил та трудомісткості управління мережною інфраструктурою. Традиційний підхід до вирішення проблеми мережних взаємодій припускає послідовну обробку одиниць передачі, зокрема пакетів, на кожному рівні еталонної моделі мережної взаємодії ISO/OSI. Одним їх напрямів “модернізації” класичного підходу до організації мережної архітектури є створення програмно- конфігурованих мереж, що використовують протокол OpenFlow [2].

До основних переваг програмно-конфігурованих мереж (ПКМ, англ. Software Defined Networks – SDN) відносять централізоване управління в мультівендорному середовищі, зменшення складності мережі за рахунок автоматизації, вищу швидкість впровадження інновацій, збільшення надійності та безпеки мережі, забезпечення узгодженості політик управління доступом, інжинірингу трафіку, параметрів якості послуг, безпеки та ін., вузькоспрямоване (“точкове”) управління мережею, поліпшення якості сприйняття послуг користувачами. Передбачається, що з урахуванням вказаних переваг SDN стане новим стандартом для майбутніх мереж.

Основна ідея ПКМ полягає в тому, щоб не змінюючи існуючого мережного устаткування відокремити (перехопити) управління цим устаткуванням (маршрутизаторами та комутаторами) за рахунок створення спеціального програмного забезпечення (ПЗ), яке може працювати на звичайному окремому комп’ютері, і яке знаходиться під контролем адміністратора мережі.

Висновки до розділу 1

1. Досліджено архітектуру програмно конфігурованих - мереж, визначено, що за допомогою протоколу OpenFlow можливо застосовувати комутатори Ethernet з використанням вбудованих таблиць потоків.
2. В розділі визначено, що продуктивність програмно – конфігурованих мереж істотно залежить від розташування контролерів OpenFlow.
3. В розділі запропоновані методи організації управління програмно – конфігурованих мереж SDN. В роботі проаналізовано відмінність цих методів з методами управління традиційними мережами.

РОЗДІЛ 2. ІМІТАЦІЙНО-АНАЛІТИЧНІ МОДЕЛІ ГІБРИДНИХ ПРОГРАМНО-КОНФІГУРОВАНИХ ПРИСТРОЇВ

У другому розділі розділі наведено опис існуючих моделей комутаторів у системі пакетного моделювання OMNeT++, розглянуто особливості моделі комутатора OpenFlow. Розроблено модулі для імітації реалістичної затримки пакетів. Розроблено імітаційні моделі комутатора із підтримкою OpenFlow та мережного пристрою з підтримкою NFV. Запропоновано алгоритм автоматизованого синтезу гібридного програмно-конфігурованого обладнання, на основі якого побудовано імітаційно-аналітичну модель гібридного програмно-конфігурованого пристрою.

2.1 Розробка мережевого обладнання

2.1.1 Дослідження процесу проходження пакета крізь гібридний програмно-конфігурований пристрій на прикладі моделі комутатора

Розглянемо базову модель комутатора, представленого у фреймворку INET OMNeT++. Комутатор представлений набором модулів, що взаємодіють через систему передачі OMNeT [98]. Найбільш докладно реалізовані модулі інтерфейсів Ethernet, зокрема є реалізації вхідної черги типу Tail-Drop, вихідних черг типів FIFO , RED , WRED , PQ , CBFQ , також можлива реалізація затримки передачі на лінії. Однак у цій базовій моделі відсутня реалізація чи імітація процесів внутрішньої обробки пакетів, вся комутація представлена модулем MacAddressTable з пошуком адрес і модулем MacRelayUnit, що просто передає пакети на вихідний інтерфейс без затримок або імітує розсилку на вибрані або всі порти [24].

На рисунку 2.1 показано схему проходження пакета через комутатор стандартного модуля EtherSwitch, який використовують у більшості моделей.

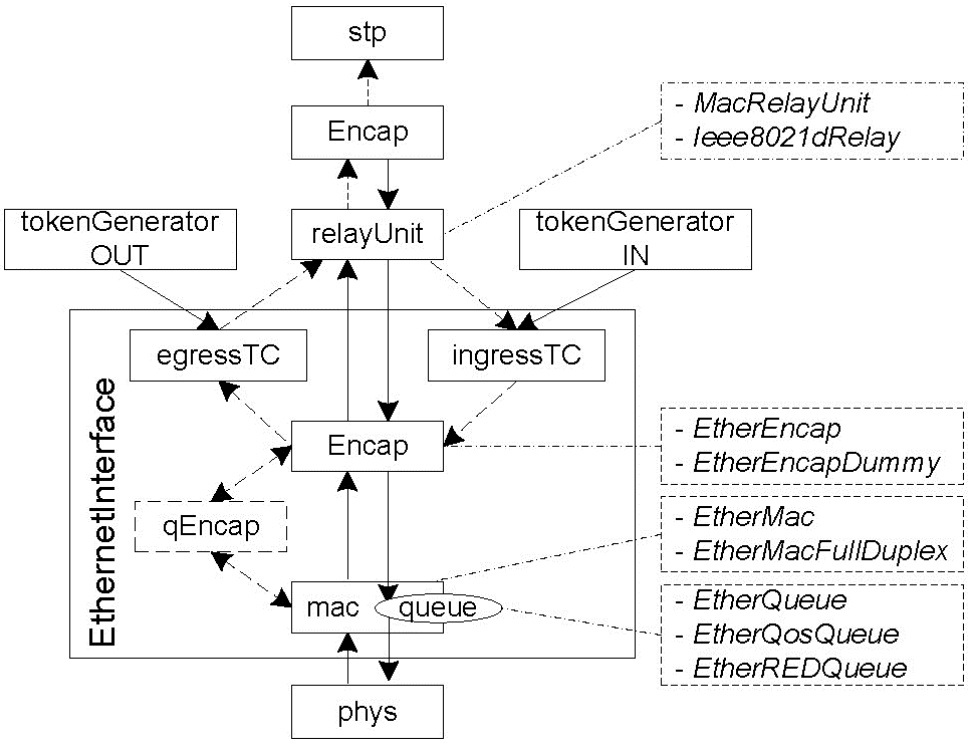


Рис.2.1. Схема проходження пакета через комутатор EtherSwitch фреймворку INET

Блок "mac" реалізує функції підрівня Media Access Control (MAC) канального рівня с використанням технології IEEE 802.3 CSMA/CD. При посиланні кадру з верхніх рівнів, кадри ставляться в чергу передачі (внутрішній submodule), крім того, за запитом верхніх рівнів може бути відправлено повідомлення PAUSE. Кадри, що надходять з мережі, обробляються так: перевіряється CRC (кадри з помилкою відкидаються), кадри PAUSE отримують відповідь, в режимі promiscuous (за замовчуванням для комутатора) відправляються всі отримані кадри, в іншому випадку (режим хоста), тільки кадри з MAC-адресами, що збігаються, і широкомовні кадри. Може бути встановлений напівдуплексний (EherMac) або повнодуплексний (EtherMacFullDuplex) режими роботи рівня mac [25].

Блок qEncap призначений для обробки тегів протоколу IEEE 802.1q (VLAN). Для обробки використовується фільтр VLANID (для забезпечення роботи лише

вибраних VLAN), а також правила перезапису тегів для додавання та видалення тегів у різних напрямках.

Блок Encap виконує функції підрівня LLC канального рівня. Пакети, що надходять із верхніх рівнів, інкапсулюються у кадр Ethernet і вирушають до MAC. Кадри Ethernet, що надходять від MAC, декапсулюються та відправляються на більш високі рівні. Для блоку Encap можуть бути встановлені режими EtherEncap (підтримує реальні кадри, які можна передати в систему моделювання як PCAP файли з дампом трафіку) або EtherEncapDummy (Пакети з випадковим внутрішнім вмістом).

Модулі ingressTC та egressTC - вхідний та вихідний формувач трафіку (Traffic conditioners). Блоки Traffic conditioners не є стандартними та мають бути створені розробником конкретних моделей. Вони можуть класифікувати вхідні пакети, вимірювати трафік у кожному класі, позначати або відкидати пакети в залежності від результату вимірювання, формувати трафік відповідно до бажаного профілю трафіку. Для зовнішнього управління поведінкою цих блоків введені модулі tokenGeneratorIN і tokenGeneratorOUT, більш детально описані далі.

У стандартному комутаторі EtherSwitch без додаткових налаштувань виникнення черги можливе лише у підмодулі queue модуля mac при проходженні пакета від верхніх рівнів до нижніх.

У маршрутизаторах модуль MAC використовує модуль зовнішньої черги для моделювання кінцевого буфера, реалізації QoS та/або RED та запиту пакетів із цієї зовнішньої черги. У хостах така черга не використовується, тому MAC містить внутрішню чергу для зберігання пакетів, що очікують на передачу. Концептуально черга нескінченного розміру, але для кращої діагностики у параметрі packetCapacity можна вказати жорстку межу, при перевищенні якої симуляція зупиняється з помилкою.

За умовчанням черга має тип EtherQueue. Схематично цю чергу показано рисунку 2.2.

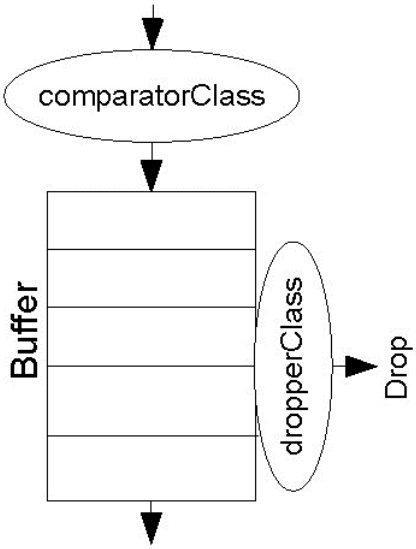


Рис.2.2. Черга EtherQueue

Клас comporatorClass визначає порядок пакетів у черзі та порядок вставки їх у чергу. Для черги EtherQueue він має значення "inet:: EthernetFrameTypeComparator", що забезпечує пріоритет кадрам PAUSE Ethernet. Параметр dropperClass визначає, які пакети відкидаються під час переповнення черги спочатку.

Окрім черги EtherQueue можуть бути встановлені черги EtherQosQueue (рисунок 2.3а) та EtherQosRedQueue (рисунок 2.3б).

Модуль черги EtherQosQueue дає кадрам PAUSE вищий пріоритет. Для обслуговування кадрів даних він може налаштуватись за допомогою інтерфейсу IPacketQueue. Модуль EtherQosQueue складається з класифікатора, двох черг та планувальника [96]. Класифікатор пакетів – це модуль, який має один пасивний вхід та кілька активних виходів. Пакети, надіслані пасивний вхід, передаються однією з активних виходів без будь-якої затримки і переупорядкування. За замовчуванням у модулі EtherQosQueue включено класифікатор, який перенаправляє кадри Ethernet PAUSE черга pauseOut, а інші кадри в чергу defaultOut. Обидві ці черги є чергами типу DropTailQueue, тобто, являють собою обмежену чергу пакетів, яка відкидає пакети у хвіст черги. Планувальник пакетів – це модуль, який має кілька активних входів та один пасивний вихід. Пакети, витягнуті з пасивного виходу, надаються одним із входів без будь-якої затримки та

переупорядкування. За замовчуванням у модулі EtherQosQueue увімкнений планувальник, який виштовхує пакети з першого непустого джерела пакетів.

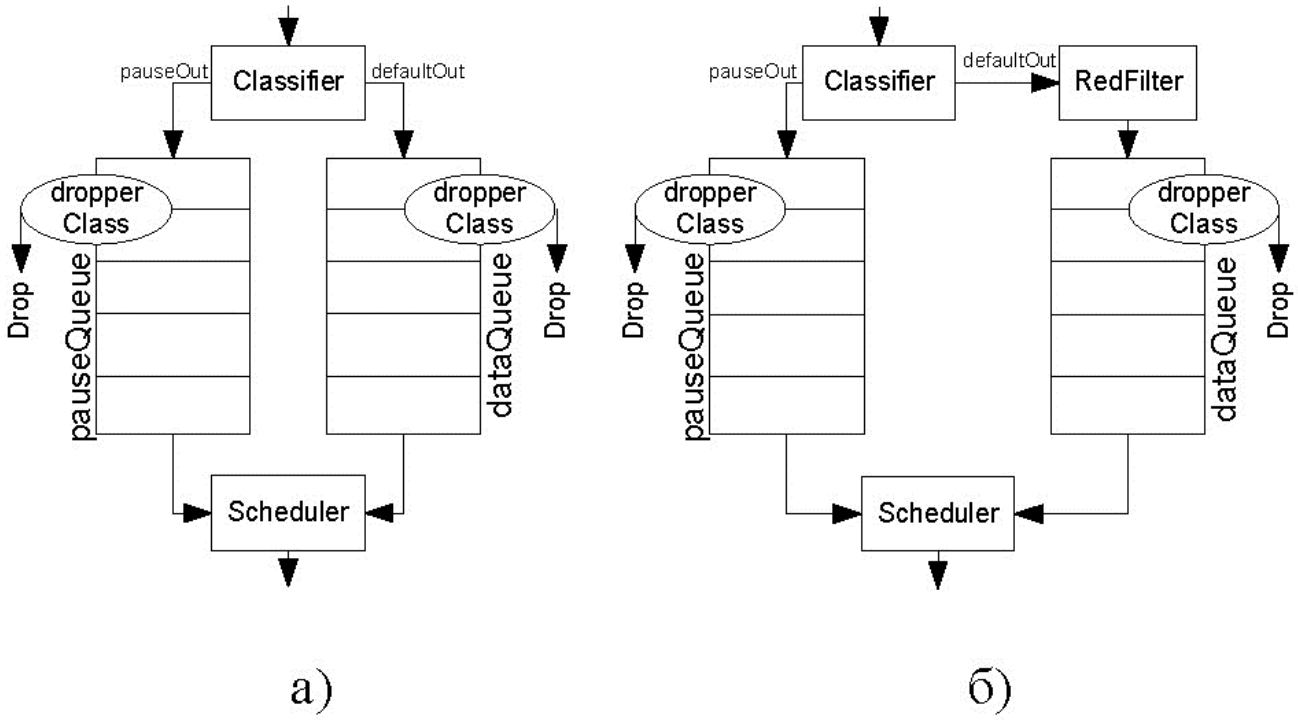


Рис.2.3. Проходження пакету через черги EtherQosQueue (а) і EtherQosRedQueue (б)

Модуль EtherQosRedQueue відрізняється від EtherQosQueue використанням алгоритму випадкового раннього виявлення кадрів даних [26]. Пакети, що надійшли в *i*-й вхідний шлюз, пересилаються в *i*-й вихідний шлюз або відкидаються. Модуль підсумовує буферний простір черг, що використовується, прикріплених до вихідних шлюзів. Якщо він нижчий за мінімальний поріг, пакет не буде відкинутий, якщо вищий за максимальний поріг, він буде відкинутий, якщо він знаходиться між мінімальним і максимальним порогом, він буде відкинутий з певною ймовірністю.

2.1.2 Розробка додаткових модулів моделі гібридного програмно-конфігурованого обладнання для реалізації внутрішніх процесів обробки трафіку мережевого пристрою

Для більш детальної реалізації моделі гібридного програмно-конфігурованого обладнання розроблено схеми додаткової обробки трафіку. На комутаторі опціонально можуть бути налаштовані такі функції, як фільтрація пакетів за допомогою списків ACL, профільники policing та shaping, класифікація та модифікація пакетів (рисунок 2.4). Всі ці функції викликаються при проходженні пакета через пристрій послідовно, вносячи затримку. Оскільки в реальному комутаторі складно чи неможливо оцінити затримку кожного елемента на різних навантаженнях, прийнято рішення додати модуль внесення агрегованої затримки «RealDelay».

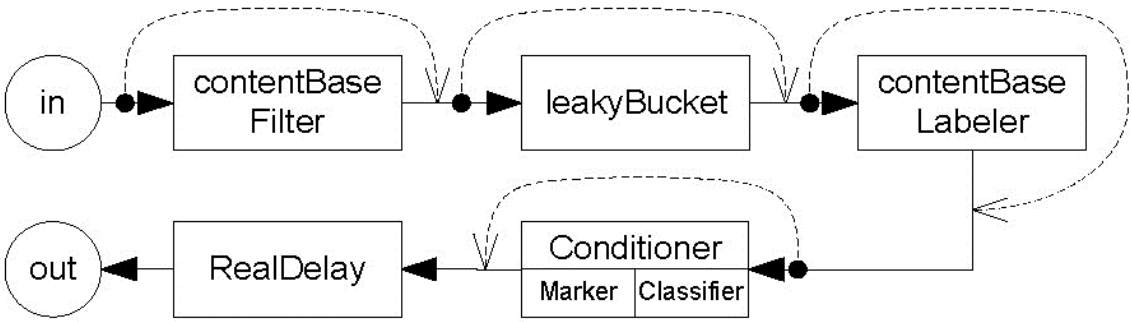


Рис.2.4. Черга EtherQueue

Фреймворк INET дозволяє реалізувати зазначені функції за допомогою наступних модулів:

- contentBaseFilter являє собою модуль, який відкидає пакети на основі даних, які вони містять, аналог Access Control List;
- leakyBucket - універсальний профільник з переповненням і швидкістю виведення, що настроюється, реалізує параметризований алгоритм з витоком, аналог Traffic Shaper;
- contentBaseLabeler прикріплює мітки до пакетів на основі даних, які вони містять, аналог Class-Map;
- як модифікатор може бути використаний модуль Conditioner (обробник за умовою), що складається із двох частин: Classifier і Marker. Classifier - класифікатор, що містить список фільтрів (модуль для перезапису полів протоколів пакета, наприклад DSCP), який ідентифікує потоки, визначає їх класи, робить

заміну полів класів. Кожен фільтр може збігатися з адресою джерела та призначення, номером протоколу IP, портами джерела та призначення або ToS/CoS/DSCP. Перший відповідний фільтр визначає індекс виходу. Якщо відповідний фільтр не знайдено, пакет буде надіслано через шлюз defaultOut. Marker - модуль, що встановлює у полі DSCP (молодші шість біт Tos / TrafficClass) IP - дейтаграм значення, вказане параметром dscps ;

- RealDelay – розроблений модуль для імітації затримки, що вноситься при проходженні пакетом описаних блоків. Схема модуля показано рисунку 2.5.

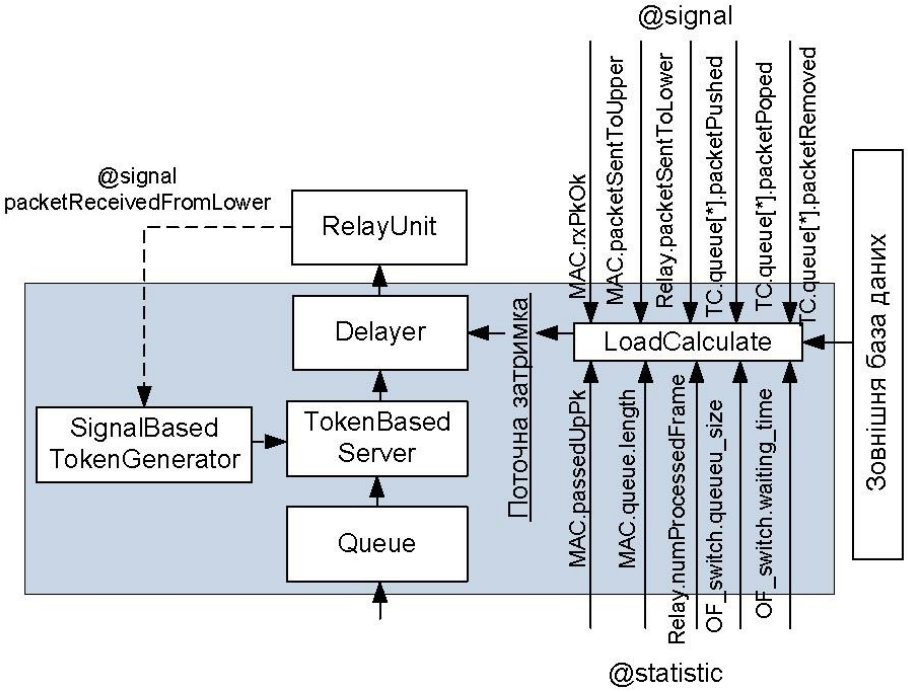


Рис.2.5. Модуль імітації затримки та обмеження ємності та продуктивності RealDelay

Розмір затримки може бути заданий як функція ймовірності, наприклад, exponential (lambda), або встановлений програмним чином. Черга емулює роботу вхідного буфера обробки механізмом Tail drop (FIFO) с обмеженою ємністю. TokenBasedServer пропускає пакети по сигналу SignalBasedTokenGenerator, який у свою чергу формує сигнал при отриманні пакета комутаційним модулем relayUnit. Для формування затримки за зовнішніми даними, зокрема за даними реального комутатора і даних від модельних модулів, розроблено модуль LoadCalculate. На

основі симплексної інтерполяції модуль здійснює розрахунок затримки за всіма вхідними даними. Розроблені модулі сумісні з існуючими моделями та можуть бути використані спільно у будь-яких схемах з комутаторами.

2.2 Розробка моделі комутатора із підтримкою OpenFlow

Різні рішення щодо моделювання OpenFlow обладнання мають джерелом одного предку [27], який працює на нових версіях системи. Для забезпечення працездатності на версії INET 4.3 довелося внести більше 100 змін у вихідний код C ++, а також файли опису елементів. Змінений дистрибутив наведено в [28]. Були додані MessageDispatcher модулі для зв'язку між рівнями з багатьма входами, також був приведений до існуючих класів весь вихідний код, логіка перероблена для роботи з новими сутностями пакетів та тегами, які тепер містять метадані мережного пакета. Версія підтримуваного протоколу OpenFlow - 1.5.1.

Модуль OpenFlow Swich структурно не відрізняється від звичайного комутатора з боку фізичного з'єднання з мережею, проте має друге виділене з'єднання для роботи з контролером. Сам модуль комутації OF_Switch за логікою роботи схожий на модуль RelayUnit, який є основою традиційного комутатора. Модуль OF_Switch містить механізм формування реалістичних затримок обробки з параметром serviceTime (використовується в режимі безумовної затримки будь-якого пакета, що прийшов на вхід, допускається використання генераторів для завдання випадкового часу заданим законам розподілу). Також на відміну від звичайного комутатора в OF_Switch є можливість завдання кінцевої черги через параметр bufferCapacity, причому внутрішній елемент buffer при цьому використовується тільки для спілкування з контролером, а повідомлення просто зберігаються в списку типу std::list час serviceTime . Як видно з рисунку 3.6, всі комунікації controlPlane для управління комутатором здійснюються виділеними каналами, що завжди відповідає реальному комутатору [29].

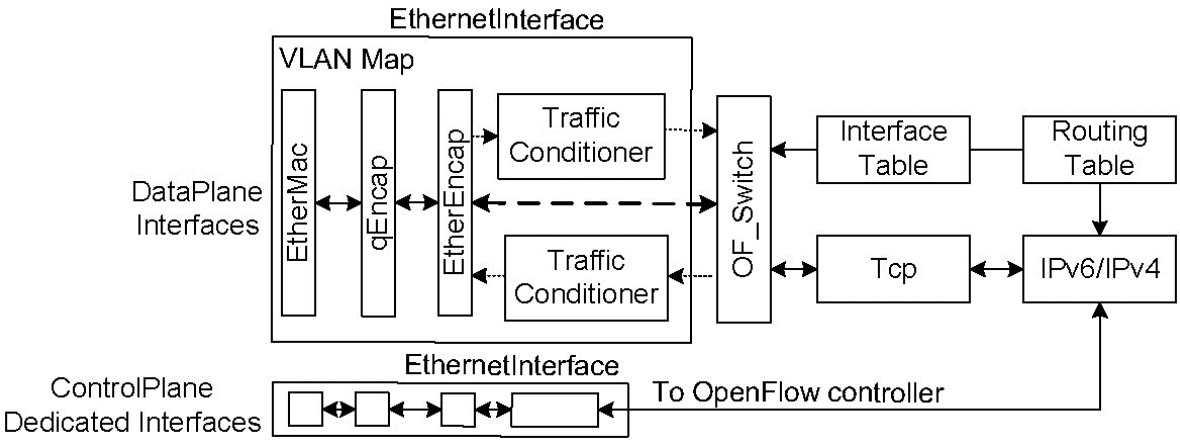


Рис.2.6. Структура комутатора OpenFlow

Крім того, модуль OpenFlow Switch не враховує, що у більшості випадків комутатори поєднують як режим класичної комутації, так і різні режими OpenFlow обробки пакетів [47].

Реальний комутатор з підтримкою OpenFlow має відмінність за принципом комутації інтерфейсів із OpenFlow (принаймні, комутатори HP ProCurve, HPE Aruba, Cisco Catalyst, NetGear M). Підтримка OpenFlow є в таких комутаторах однією з можливих технологій комутації всередині VLAN поряд з кешуванням маршрутів, класичною комутацією 2 і 3 рівня, а також маршрутизацією між VLAN [31]. Увімкнення OpenFlow проводиться у вибраних VLAN, при цьому класична комутація відключається не повністю, таблиця MAC TCAM відповідність адрес продовжує наповнюватися, але не використовується. У багатьох комутаторах (наприклад, HP, HPE) OpenFlow керує тільки рішенням про класифікацію, комутацію, маршрутизацію, перезапис полів пакета і генерацію пакетів. Усі функції, пов'язані з чергами, забезпеченням параметрів QoS (крім класифікації та маркування пакетів), обмеженням пропускної спроможності, динамічною піритизацією в таких комутаторах працюють тільки у звичайному режимі, та OpenFlow немає доступу до управління. Також зв'язок з контролером у реальних комутаторах здійснюється через адресу IPv 4/ IPv 6, встановлену на одному з VLAN або інтерфейсів з класичною комутацією, для зв'язку використовують традиційні протоколи ARP , маршрутизація через таблицю маршрутизації VLAN або VRF, фільтри ACL для контролю доступу до вбудованого сервера OpenFlow [22].

На рисунку 2.7 наведена розроблена гібридна схема для комутації, яка враховує можливість співіснування на різних портах (групованих за різними VLAN) як комутації через RelayModule, так і з обробкою пакетів OF_Switch.

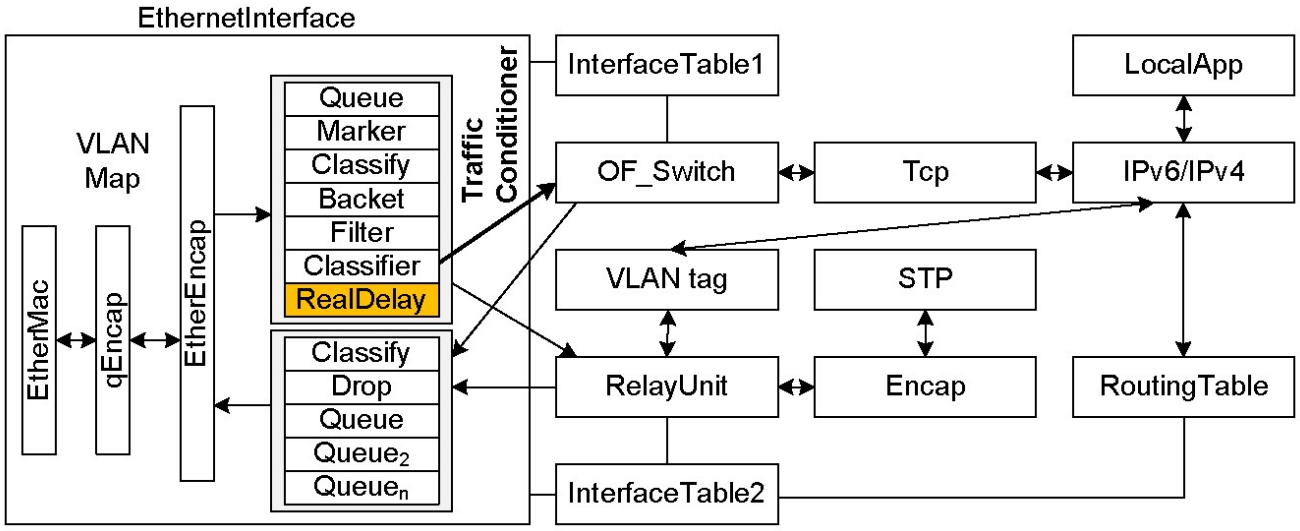


Рис.2.7. Схема гібридної комутації

Основна складність реалізації такого комутатора в OMNET ++ - автоматичне заповнення списку інтерфейсів модулем InterfaceTable, який бачитиме всі інтерфейси комутатора. Для вирішення цієї проблеми додано дві таблиці інтерфейсів, одна з яких використовується лише для OpenFlow. Інтерфейси до цієї таблиці додаються за фільтром VLAN.

Поділ вхідного трафіку для обробки OpenFlow також відбувається за VLANID за допомогою ContentBasedClassifier, при цьому зберігається можливість роботи комутатора за класичною схемою. Вихідні пакети обробляються однаково і проходять через черги, настроєні на інтерфейсах. Один інтерфейс може приймати пакети як від OpenFlow DataPlane, OpenFlow ControlPlane, так і від комутації.

Управління комутацією OpenFlow відбувається через модуль зв'язку з контролером на основі модуля NetworkLayer, який за таблицею маршрутизації вирішує як надіслати пакет, формує ARP запити, приймає та фільтрує всі IP пакети, що прийшли в комутатор, обробляє призначені для встановленого IP адреси пакети [32,33].

2.3 Розробка моделі мережевого пристрою NFV

Технологія віртуалізації мережевих функцій побудована на базі програмно-конфігурованих мереж, а найбільш популярні рішення на базі Linux використовують як програмно-керований комутатор Open vSwitch. Розглянемо методи формування моделі такого комутатора.

Open vSwitch архітектурно відрізняється як від класичного, так і від OpenFlow реального комутатора. Оскільки немає виділених блоків передобробки пакетів (крім мережевих карт з апаратною обробкою заголовків та FPGA), виділеної пам'яті та процесорів для комутації, Open vSwitch використовує модуль ядра (або модуль DPDK для прямої передачі без участі ядра) для передачі пакетів за інформацією з бази даних OVSDB або OpenFlow таблиць. Оскільки кількість зовнішніх інтерфейсів сервера істотно менше таких у комутатора, вхідний трафік не фільтрується автоматично по VLAN, не обробляється (крім декапсуляції Ethernet) до попадання у модуль комутації. Вся обробка трафіку здійснюється тим самим процесором, на якому виконуються віртуальні машини NFV, що є істотним фактором внесення випадкових затримок в обробку пакетів. Багато стандартних процедур в OVS немає, вони виконуються засобами відповідних Linux програм, наприклад, маршрутизація, маркування пакетів, фільтрація. В останніх версіях з'явилося маркування пакетів засобами ovsdb, але більшість NFV середовищ використовує OpenFlow для роботи, управління, забезпечення QoS та обмеження смуги пропускання, а також для призначення відповідності заздалегідь створеним чергам класів пакетів. Схематично структура NFV комутації показано рисунку 2.8.

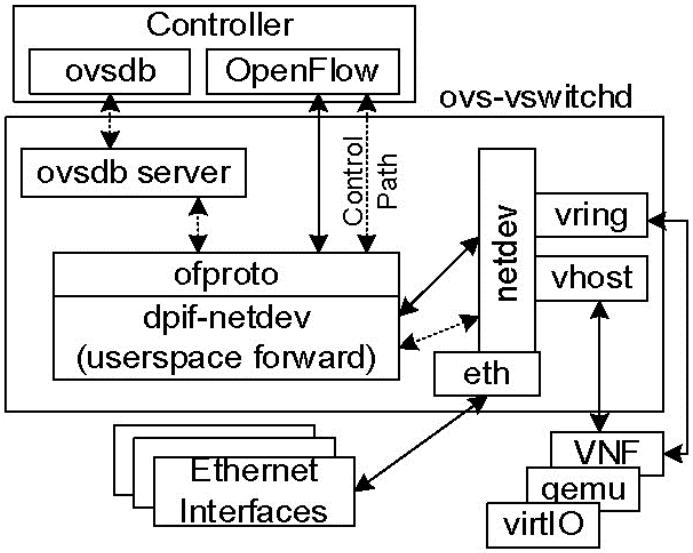


Рис.2.8. Структура NFV комутації

Моделювання Open vSwitch у складі NFV ускладнено тим, що безліч процесів, що відбуваються, приховано за контролером [34]. Наприклад, для нестаціонарних мереж 5G VANET, MANET, основним для роботи сервісів може бути Virtual Path Computation на основі даних про розміщення NFV та топології мережі, крім того, сам контролер також може бути одним з елементів NFV, у тому числі розподіленим на кілька серверів [35,36].

Оскільки Open vSwitch використовує загальну фабрику для обробки всіх правил (рисунок 2.9) схема роботи такого комутатора більш універсальна. Для створення моделі такого комутатора у системі OMNeT ++ потрібно поєднати швидку комутацію, стандартну L 2/ L 3/ L 4 комутацію та запам'ятовування даних, а також OpenFlow підхід.

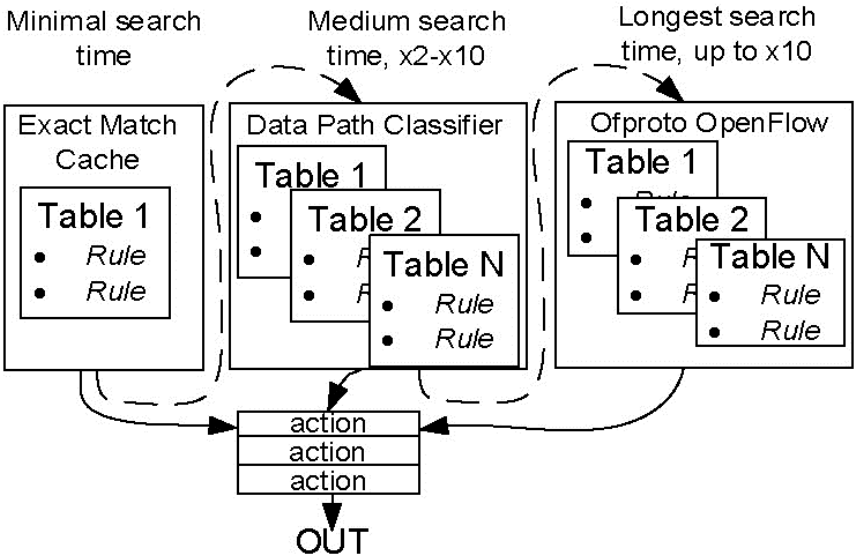


Рис.2.9. Пошук правил в Open vSwitch

Робота Data Path Classifier та OpenFlow є схожою, пошук ведеться за однаковими полями, розрізняються лише події над пакетами, які специфіковані кожним типом таблиць. На рисунку 2.10 показано схему Open vSwitch комутатора, де модуль ovs-controller відповідає за швидку комутацію при відключеному режимі OpenFlow.

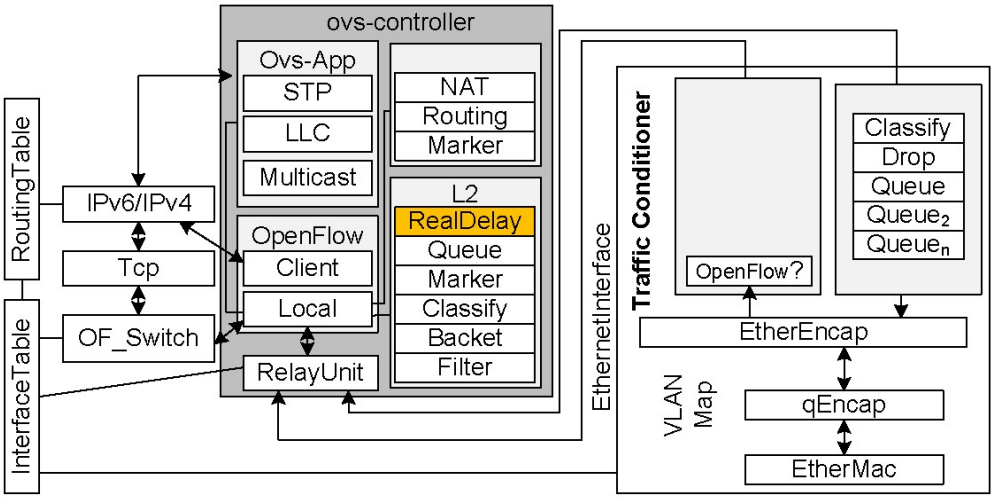


Рис.2.10. Схема Open vSwitch комутатора

Таблиці Datapath Classifier емулюються режимом роботи OpenFlow з локальним контролером, за наявності правил LOCAL звернення до зовнішнього контролера немає. Якщо правило вказує на здійснення дії, пакет проходить крізь вибраний модуль L2 або L3, модуль Ovs-App приймає та генерує пакети для

специфічних протоколів. Увімкнення режиму OpenFlow для пошуку модуль OpenFlow передає при надходженні пакета локальні таблиці та дані OpenFlow таблиць.

Такий режим роботи OpenFlow дозволяє емулювати весь функціонал Open vSwitch крім черг, для них залишено класичний режим роботи, хоча Open vSwitch може організовувати власні черги і сам комутувати пакети в них.

2.4 Алгоритм автоматизованого синтезу гібридного програмно-конфігурованого обладнання

Синтез гібридного програмно-конфігурованого пристрою починається після отримання даних експериментального дослідження, що являють собою набори вхідних та вихідних характеристик. Вхідні характеристики включають параметри генерації трафіку, дампи цього трафіку на виході схеми експериментального дослідження гібридного пристрою, а також налаштування цього пристрою. Вихідні дані є вимірювані затримки проходження трафіку, дані про навантаження на процесор, пам'ять і переривання гібридного програмно-конфігурованого пристрою, а також сумарні відомості про навантаження на всі інтерфейси і довжини черг.

Алгоритм автоматизованого синтезу гібридного програмно-конфігурованого обладнання складається з наступних кроків.

Крок 1. За даними конфігурації гібридного програмно-конфігурованого пристрою вибираються відповідні блоки для створення моделі.

1.1 За результатами аналізу кількості портів та їх типу додаються блоки інтерфейсів у файл моделі NED, дані про типи, швидкості, затримки додаються до файлу налаштувань INI.

1.2 За результатами аналізу налаштування вибираються блоки черг для інтерфейсів у файл моделі NED, а також додається блок RealDelay за кожен інтерфейс.

1.3 Файл з результатами вимірювання затримок, файл з даними про навантаження на ресурси ДВКО копіюються в директорію моделі, в INI файл прописуються шляхи до цих файлів для модуля RealDelay.

1.4 За даними аналізу параметрів за замовчуванням (наприклад, виведення команди show tech-support) на інтерфейси додаються вихідні черги з відповідними налаштуваннями типу та довжини.

1.5 Якщо комутатор містить логіку OpenFlow, то додається ланцюжок обробки OpenFlow, а також модуль локальної таблиці OpenFlow.

1.6 Якщо комутатор містить хоча б один порт не керований OpenFlow, додаються стандартні модулі комутації та таблиці адрес.

Крок 2. Підготовка вхідних даних.

2.1 Для аналізу результатів затримок у файл INI на інтерфейсах включаються обробники подій надходження пакета для запису подій у лог.

2.2 Файли pcap, отримані в результаті експериментального дослідження, нормалізуються за допомогою утиліт pcapedit та tshark для приведення часу до нульового базису та обчислення максимального часу циклу експерименту.

2.3 З файлів pcap формуються CSV файли, що містять час приходу пакетів та їх розмір за кожним пунктом плану експерименту.

Крок 3. З моделей pcaprecorder, pcapwriter, clientapp та створеного у пункті 1 комутатора створюється файл схеми мережі для експерименту над моделлю.

3.1 У Ned файл схеми мережі додаються посилання на всі необхідні модулі залежностей, що використовуються в пунктах 1 і 3 моделей.

3.2 В INI файл як модель для запуску додається схема мережі. Як її параметри створюються три профілю проведення експерименту:

- проведення експерименту одному значенні генератора трафіку, відповідного максимальним досягнутим інтенсивностям при експериментальному дослідженні;
- проведення серії експериментів відповідно до плану експерименту, отриманого в ході експериментального дослідження пристрою, без включення

додаткових налаштувань. У цьому використовуються відповідні посилання файл трафіку, ім'я якого має містити порядковий номер циклу експерименту. Цей номер буде використовуватися для визначення рядка в результатах вимірювання затримок і навантажень на ресурси;

- проведення серії експериментів відповідно до плану експерименту, отриманого в ході експериментального дослідження пристрою, з включенням додаткових налаштувань. Посилання на дампи трафіку генеруються аналогічно до попередньої серії. Дані про активовані настройки беруться з файлу налаштувань XML, який генерується для кожного циклу під час експериментального дослідження та містить список увімкнених налаштувань. Посилання на файл налаштувань подається в модель RealDelay.

Крок 4. Створюється файл компіляції моделі та запуск серії експериментів.

4.1 Для середовища Windows формується CMD файл, у якому запускається середовище mingw, у ній запускається компіляція моделі, потім кожному профілю проведення експерименту запускається виконання моделі зі збереженням всіх вихідних даних у окрему папку.

4.2 Для середовища Linux формується SH файл, в якому виконується файл налаштувань ENV, формується Makefile, запускається компіляція з виведенням моделі в окремий бінарний файл. Для кожного профілю проведення експерименту запускається виконання зі збереженням даних окремий каталог.

4.3 Формується Perl файл (для середовища Windows для виконання в середовищі mingw). У файл додаються рядки зі списком папок проведення експерименту, циклом читання файлів логів для пошуку рядків входу та виходу пакета з комутатора з групуванням за номером пакета, а також виведення цих даних CSV файлів по кожному циклу експерименту.

4.4 Формується файл SH для запуску tcpdump на читання pcap файлів кожного циклу експерименту та виведення в CSV файл часу та розміру кожного пакета.

4.5 Формується файл для інтерпретатора R із циклом за файлами CSV експерименту, файлами CSV з результатами експериментального дослідження,

файлами CSV з часом приходу кадрів з pcap файлів для експериментального дослідження, а також підключенням зовнішніх бібліотек fitdistrplus , ggplot2 та формуванням даних аналізу розподілу експериментальних даних.

Крок 5. Запуск сформованих у пунктах 4.2 – 4.5 файлів, аналіз вихідних результатів інтерпретатора R у частині перевірки гіпотез, а також у частині аналізу розподілу часу між кадрами в експериментальному дослідженні та для результатів проведення експерименту, формування файлу звіту CSV із цими результатами.

Складність даного алгоритму залежить кількості вхідних параметрів пристрою й у разі не перевищує O(n).

В результаті виконання алгоритму синтезується імітаційно-аналітична модель гібридного програмно-конфігурованого пристрою, що дозволяє досліджувати його поведінку за будь-яких поєднань вхідних параметрів.

Висновки до розділу 2

1. В розділі запропонована удосконалена модель комутатора із підтримкою OpenFlow. Ця модель дозволила обробляти трафік з довільною біфоркацією.
2. Реалізація алгоритму автоматизованого синтезу гібридного програмно-конфігурованого обладнання дозволила підвищити показники якості інфокомунікаційних мереж України.
3. В розділі описано режим роботи комутатора OpenFlow. Такий режим роботи дозволяє емулювати весь функціонал Open vSwitch крім черг, для них залишено класичний режим роботи. В розділі показано Open vSwitch може організовувати власні черги і сам комутувати пакети в них.

РОЗДІЛ 3. МЕТОДИ ТА АЛГОРИТМИ АВТОМАТИЗОВАНОГО ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ ГІБРИДНОГО ПРОГРАМНО-КОНФІГУРОВАНОГО ОБЛАДНАННЯ

Другий розділ присвячено розробленню методів і алгоритмів автоматизації експериментального дослідження гібридного програмно-конфігурованого обладнання. Для побудови структурної моделі вхідних і вихідних даних розроблено методику дослідження параметрів реального обладнання. Запропоновано методику автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання на основі синтезу віртуальної інфраструктури. Розроблено алгоритм автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання.

3.1 Отримання параметрів гібридного програмно-конфігурованого обладнання

Більшість модельних досліджень мережевих пристроїв орієнтуються на вивчення пропускної спроможності, затримок, втрат пакетів, поведінки черг, що вимагає деталізації до рівня інтерфейсів, але не внутрішніх процесів ухвалення рішень, пошуку адрес, класифікації, фільтрації та інших важливих процесів.

Для вивчення параметрів роботи гібридного програмно-конфігурованого пристрою на проміжку часу T позначимо задані вхідні параметри, що описують вхідний трафік і поточні налаштування пристрою, як $Z(t) = \{ Traffic(t), Device(t) \}$, $\forall t \in T$, а виміряні вихідні параметри, що містять часові характеристики, параметри вхідних і вихідних черг, статистику за чергами і зліпок метрик пристрою, як $Y(t) = \{ Latency(t), Qout_{ij}(t), Qin_{ij}(t), S_{ij}(t), State'_i(t) \}, \forall t \in T$.

Тоді функція відображення вхідних параметрів на вихідні матиме вигляд:

$$F(t) : Z(t) \rightarrow Y(t), \ \forall t \in T . \quad (3.1)$$

Представимо дерево вхідних параметрів $Z(t)$ у вигляді лінійної розгортки:

$$Z(t) = \{z_1(t), z_2(t)... z_n(t)\},\ \forall t \in T\ , \tag{3.2}$$

де $n \in N$ - кількість вхідних параметрів у розгортці.

Аналогічно відобразимо дерево вихідних параметрів $Y(t)$ у вигляді лінійної розгортки:

$$Y(t) = \{y_1(t), y_2(t)... y_m(t)\},\ \forall t \in T\ , \tag{3.3}$$

де $m \in N$ - кількість вихідних параметрів у розгортці.

Вхідна множина параметрів $Z(t_i)$ при $\forall t_i \in [T_{i-1}.. T_i) \in T$ не має змінюватися, а тривалість інтервалу часу $[T_{i-1}.. T_i)$ має бути достатня для необхідної кількості вимірювань вихідних параметрів і досягнення стаціонарності ймовірнісних закономірностей вихідних параметрів $\{y_1(t_i), y_2(t_i)... y_m(t_i)\}$.

Тоді відображення (3.1) можна привести до вигляду $F(t) = \{f_1(t), f_2(t)... f_m(t)\}$ так, щоб:

$$\begin{cases} f_1(t_i) : \{ z_1(t_i), z_2(t_i)\dots z_n(t_i) \} \rightarrow y_1(t_i) \\ f_2(t_i) : \{ z_1(t_i), z_2(t_i)\dots z_n(t_i) \} \rightarrow y_2(t_i) \\ \dots \\ f_k(t_i) : \{ z_1(t_i), z_2(t_i)\dots z_n(t_i) \} \rightarrow y_k(t_i) \end{cases} \tag{3.4}$$

Під час моделювання необхідно забезпечити відповідність одержуваних вихідних параметрів досліджуваного пристрою $Y(t_i)$ і одержуваних вихідних параметрів моделі $Y^*(t)$ за одних і тих самих наборів вхідних параметрів. Для цього знайдемо таке відображення $F(t) : Z(t) \rightarrow Y^*(t), \forall t \in T$, що

$$\left|\frac{y_j(t_i) - y_j*(t_i)}{y_j(t_i)}\right| < \varepsilon,\ \forall j \in [1..m]\ , \tag{3.5}$$

де, $y_j*(t_i) \in Y*(t_i)$ – виміряні вихідні параметри моделі;

ε - задана точність.

Дослідження пристрою за всіх сполучень вхідних параметрів $Z(t)$ ускладнюється тим, що кількість усіх можливих сполучень параметрів залежить від кількості та діапазонів параметрів конкретних пристроїв із конкретними версіями системного ПЗ, і в загальному вигляді є NP-повною задачею [37,38].

З огляду на те, що такі ресурси пристрою, як пам'ять, об'єм буферів, апаратні та програмні можливості оброблення пакетів, шина, є обмеженими та розподіленими між багатьма інтерфейсами та функціями пристрою, вплив

значення кожного вхідного параметра та сполучень значень вхідних параметрів може призводити до нелінійної поведінки відображення *F(t)*. Так, у разі вичерпання доступних буферів або ресурсів апаратних обробників різко збільшуватиметься час обробки пакетів, можуть виникати втрати у вхідних буферах і на шині комутації. Якщо робота з чергами і втратами пакетів широко представлена практично у всіх системах моделювання, то неоднозначна поведінка часу, необхідного на обробку вхідного пакета від поточної завантаженості ресурсів і необхідного оброблення пакета, представлена слабо. Для пристроїв із підтримкою ПКЗ час ухвалення рішення для опрацювання пакета за реактивного способу комутації залежить від кількості таблиць і переліку необхідних дій із пакетом, тривалість яких може мати нелінійну залежність від завантаження процесора.

Оскільки затримка пакета складається з часу перебування в чергах і часу оброблення та комутації, а час оброблення залежить від налаштувань і застосованих способів оброблення і може змінюватися в широких межах, потрібно більш детально вивчити процеси, що впливають на цю затримку. Однак імовірність втрати пакета під час обробки внутрішніми програмними модулями, його фільтрації, зміни істотних полів заголовків, що впливають на потрапляння в різні черги, не враховуються в стандартних моделях більшої частини систем моделювання [83].

Розглянемо схему проходження пакета крізь гібридне програмно-конфігуроване обладнання на прикладі програмного забезпечення для комутатора Cisco IOS як такого, що якнайповніше відображає всі функції комутатора в кампусних мережах [40], MikroTik як представника Linux комутації та маршрутизації [41], Huawei як представника альтернативного підходу до розподілу управління потоком даних [42]. На рисунку 3.1 показано узагальнену схему проходження пакета крізь комутатор за умови комутації другого рівня.

Пристрої різних виробників можуть мати різний порядок дій, деяких кроків може не бути, або можуть бути присутніми додаткові кроки, однак самого принципу проходження пакета крізь комутатор дотримуються. Усі необхідні дії на вході в інтерфейс виконуються послідовно, вносячи водночас затримки, позначені

на рисунку 3.1 як $t_1 - t_9$ [43]. Кожна дія всередині складається з розрахунків (наприклад, у профілювальниках і обмежувачах потоку), безлічі порівнянь зі списками змінної довжини (фільтри, модифікатори, класифікатори), доступом до загальної асоціативної програмної пам'яті (швидка комутація) або апаратних способів роботи з таблицями TCAM (класична комутація) [44].

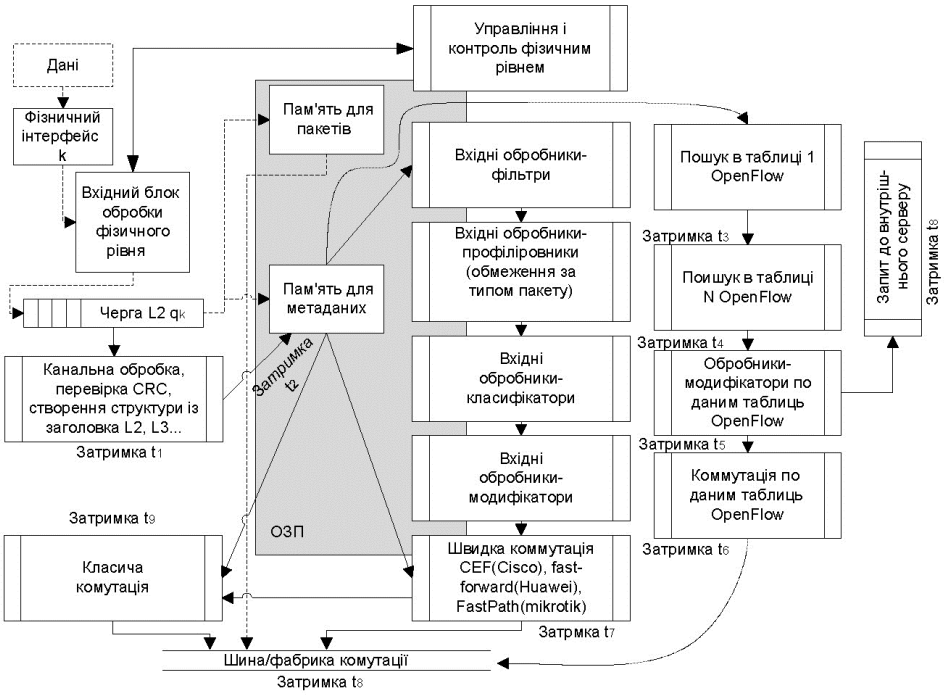


Рис.3.1. Схема обробки даних на вході в гібридне програмно- конфігуроване обладнання

Велику затримку можуть вносити довгі списки в операціях фільтрації, класифікації та модифікації, а також зайнятість процесору, пам'яті та апаратних модулів під час роботи із зовнішніми даними (кешами та таблицями) [45].

У сучасних гібридних програмно-конфігурованих пристроях розділяють рівень керування та рівень передавання даних і часто виділяють їм свої процесори/ядра та пам'ять для зменшення впливу навантаження процесора, кількості запитів до пам'яті та модулів на загальну затримку. Тим не менш, загальний вплив усіх цих процесів на продуктивність може бути суттєвим [46]. Так MikroTik заявляє про дворазове зниження продуктивності при ввімкненні фільтрації. У комутаторах Cisco частина фільтрів ACL, Route-map обробляється в апаратних TCAM таблицях (наприклад, у Catalyst 4500 тільки за масками підмереж

/24 і /32), решта фільтрів обробляються процесором і знижують загальну продуктивність.

Загальну затримку проходження пакета через гібридне програмно-конфігуроване обладнання можна прийняти за суму затримок обробки, очікування в черзі, надсилання.

$$t_{switch} = t_{process} + t_{queue} + t_{transmit}, \quad (3.6)$$

де $t_{process}$ - сумарний час обробки пакета від отримання останнього біта на вході до розміщення пакета в чергу;

t_{queue} - час очікування в черзі;

$t_{transmit}$ - час на передачу даних у лінію зв'язку, для повнодуплексних ліній $t_{transmit} = L/C$, де L - довжина переданих даних, разом із преамбулою і закінченням, у біт; C - швидкість лінії, у біт/с.

Розглянемо склад затримки обробки:

$$t_{process} = \sum_i (t_i^{op} n_i + t_i^{'op} n_i' + t_i^{wait} + t_i'^{wait}), \quad (3.7)$$

де $i \in N$ - кількість блоків обробки;

t_i^{op} - час виконання однієї операції i-го блоку обробки в програмному режимі;

n_i - кількість операцій i-го блоку обробки, що виконується програмним способом, для кожного пакета може бути різною;

$t_i^{'op}$ - час виконання однієї операції i-го блоку обробки в апаратному режимі;

n_i' - кількість операцій і-го блоку обробки, що виконується апаратними модулями, для кожного пакета може бути різною;

t_i^{wait} - сумарний час очікування доступу до поділюваних ресурсів процесора під час виконання i-го блоку в програмному режимі;

$t_i'^{wait}$ - сумарний час очікування доступу до поділюваних ресурсів процесора під час виконання i-го блоку в апаратному режимі.

Оскільки сучасний гібридний програмно-конфігурований пристрій являє собою пристрій із кількома обчислювальними ядрами, десятками інтерфейсів,

сучасною операційною системою та декількома шинами даних, затримка виконання кожної операції може варіюватися в широких межах, але не буде меншою за $t_i^{op}n_i + t_i'^{op}n_i'$. Оскільки максимальне значення сумарної затримки залежить від очікування доступу до ресурсів під час підвищення навантаження на комутатор загалом, необхідно оцінити ймовірності виникнення доступу до поділюваних ресурсів [30].

Накладення безлічі незалежних випадкових потоків призводить до найпростішого потоку з експоненціальним законом розподілу [19]. Функція розподілу $F_{in}(t)$ часу t між надходженнями двох завдань обробки одного класу (завдань роботи з одними поділюваними ресурсами) визначається як:

$$F_{in}(t) = 1 - e^{-t \cdot \lambda_i} \tag{3.8}$$

де λ_i - інтенсивність надходження завдань одного класу.

Отримання даних про затримку опрацювання пакета в реальному пристрої можливе з довжини вхідної черги q_i (для комутаторів з апаратними обробниками портів, наприклад, у Cisco командою *show buffers input-interface*), у Linux довжину черги драйвера можна отримати через утиліту *ethtool* з ключем - *S*, для Open vSwitch командою *ovs-ofctl queue-stats* [48].

З теорії систем масового обслуговування відомо, що час очікування в черзі пов'язаний з довжиною черги співвідношенням:

$$\bar{T}k = \frac{\bar{L}k}{\lambda_k}, \tag{3.9}$$

де $\bar{L}k$ - середня довжина черги в даний часовий інтервал на інтерфейсі k;

λ_k- інтенсивність надходження нових пакетів на інтерфейсі k, $\lambda_k = 1/\tau_k$ для експоненціального закону розподілу часу між пакетами τ_r.

Для апаратних комутаторів $\bar{T}_k = t_{process}$ для порту k, оскільки пакет перебуває у вхідній черзі до моменту комутації [49]. Для задання параметрів моделі, адекватної реальному комутатору в плані затримок з обробки пакетів до комутації, необхідно створити модуль черги з оброблювальним пристроєм, що вносить затримки відповідно до функції затримки реального комутатора [50]. У загальному випадку для цього потрібен розподіл (2.3).

Для синтезу необхідно дослідити гібридний програмно-конфігурований пристрій за допомогою вимірювання характеристик за різних поєднань параметрів вхідного трафіку, кількості задіяних інтерфейсів. Для всебічного вивчення роботи пристрою необхідно скласти план дослідження, що містить у собі генерацію трафіку за експоненціальним законом розподілу часу між пакетами, з поступовим збільшенням інтенсивності та кількості задіяних інтерфейсів:

$$\Lambda = \begin{bmatrix} \lambda_{11} \; \lambda_{12} \dots \lambda_{1k} \\ \lambda_{21} \; \lambda_{22} \dots \lambda_{2k} \\ \dots \quad \dots \quad \dots \quad \dots \\ \lambda_{m1} \; \lambda_{m2} \dots \lambda_{mk} \end{bmatrix}, \tag{3.10}$$

де k - кількість інтерфейсів для тестування;

m - кількість планів експерименту.

$$\lambda_{i,j} = h(i, j) \cdot \lambda_{i,j-1},\ i \in (1,k],\ j \in (2,m]\ , \tag{3.11}$$

де функція $h(i,j) = (i \cdot m + j) \cdot 2^{2R}$ надає прогресивний коефіцієнт для показового збільшення трафіку, коефіцієнт R використовується для вибору такого кроку, щоб

$$R : \lambda_{m,k} \leq \sum_{i=1}^{k} \mu_i\ , \tag{3.12}$$

де μ_i- продуктивність обслуговування i-го інтерфейсу.

Кожен елемент матриці Λ використовується для генерації трафіку інтенсивністю λ_i,j за кількості задіяних вхідних інтерфейсів, що дорівнює j. Результати записуються у вихідні матриці D (середня повна затримка передавання), Q_i (середній розмір вхідної черги), Q_o (середній розмір вихідної черги - тільки якщо використовується один інтерфейс, що виходить у генератор трафіку), L (кількість втрачених пакетів).

Якщо вже наявна матриця результатів експерименту занадто докладна для використання в моделі, її апроксимують матрицею меншого рангу будь-яким обраним методом, наприклад, методом найменших квадратів або сплайнами.

Для отримання адекватної моделі на рівні перших моментів розподілу часу між пакетами достатньо симулювати поведінку функції обробки за затримкою з тими самими першими моментами розподілів.

Розглянемо створення блоку функції затримки обробки на прикладі вихідної матриці середньої довжини вхідних черг Q_i, що має ту саму розмірність, що й Λ, тобто кожному значенню $\lambda_{i,j}$ однозначно відповідає значення $q_{i,j}$. Для того щоб визначити середнє значення черги $F(\lambda_0,j)$ для довільного λ_0 за заданої кількості інтерфейсів j, необхідно знайти таке значення i, за якого $\lambda_{i,j} \leq \lambda_0 < \lambda_{i+1,j}$.

Тоді в разі використання лінійної інтерполяції відрізків між вузлами, утвореними стовпчиком j матриці Λ, значення довжини черги для матриці результатів (наприклад, матриці Q_i) обчислюється за формулою:

$$F(\lambda_0,j) = q_{i,j} + u \cdot (q_{i+1,j} - q_{i,j}) \quad (3.13)$$

де коефіцієнт $u = \frac{\lambda_0 - \lambda_{i,j}}{\lambda_{i+1,j} - \lambda_{i,j}}$.

Як правило, функції залежності затримки від інтенсивності надходження пакетів мають вигляд степеневих функцій, похибка кусочно-лінійної інтерполяції порівняно з використанням більш високих порядків інтерполяції залежить від числа рядків матриці Λ і кроку зміни параметрів. Вона може бути порівнянна з похибкою експериментальних вимірювань за короткого кроку і вносити суттєві зміни у функціонування мережі на довгих кроках. Для кроку між вимірюваннями понад 50 пакетів краще використовувати інтерполяцію вищими степенями.

3.2 Розроблення структурної моделі гібридного програмно-конфігурованого обладнання для проведення експериментального дослідження

3.2.1 Структурна модель вхідних параметрів гібридного програмно-конфігурованого обладнання

Нехай *Traffic* - безліч потоків, що надходять на гібридний програмно-конфігурований пристрій. Кожен потік $Traffic_i(t) \in Traffic$ у момент часу t визначається такими параметрами та динамічними характеристиками:

$$Traffic_i(t) = \{\tau_i(t), PSize_i(t), Type_i(t), DSCP_i(t), VID_i(t), L2_i(t), L3_i(t), L4_i(t)\}, \quad (3.14)$$

де $\tau_i(t)$ - середній час між пакетами;

$PSize_i(t)$ - розмір пакета;

$Type_i(t)$ – тип пакета (одноадресний, багатоадресний, широкомовний);

$DSCP_i(t)$ - поля DSCP IP-заголовка;

$VID_i(t)$ - поле VLAN ID в заголовку 802.1Q;

$L2_i(t) = \{ProtoL2_i(t), SrcL2_i(t), DstL2_i(t)\}$ - характеристики протоколу канального рівня: $ProtoL2_i(t)$ - протокол, $SrcL2_i(t)$, $DstL2_i(t)$ – адреси джерела та призначення канального рівня відповідно;

$L3_i(t) = \{ProtoL3_i(t), SrcIP_i(t), DstIP_i(t)\}$ - характеристики протоколу мережевого рівня: $ProtoL3_i(t)$ - протокол, $SrcIP_i(t)$, $DstIP_i(t)$ - IP-адреси джерела та призначення відповідно;

$L4_i(t) = \{ProtoL4_i(t), SrcPort_i(t), DstPort_i(t)\}$ - характеристики протоколу транспортного рівня: $ProtoL4_i(t)$ - протокол, $SrcPort_i(t)$, $DstPort_i(t)$ - порти джерела та призначення відповідно.

Нехай *Devices* — множина гібридних програмно- конфігурованих пристроїв. Кожен пристрій $Device_i(t) \in Devices$ в момент часу t визначається наступними параметрами та динамічними характеристиками:

$$Device_i(t) = \{State_i(t), Tables_i(t), Ports_i(t), s_i(t)\} \quad (3.15)$$

де $State_i(t) = \{Proc_i(t), Mem_i(t), Flow_i(t), Intr_i(t), Ctx_i(t)\}$ - зліпок метрик пристрою: $Proc_i(t)$ - загрузка процесора пристрою, $Mem_i(t)$ - завантаження пам’яті пристрою; $Flow_i(t)$ - кількість потоків; $Intr_i(t)$ - кількість переривань; $Ctx_i(t)$ - кількість зміни контексту;

$Tables_t(t)$ - множина, що визначає параметри таблиць гібридного програмно-конфігурованого пристрою;

$Ports_t(t) = \{port_{i1}(t), port_{i1}(t), ..., port_{ij}(t)\}$ - кількість j мережевих портів;

$s_i(t) \in \{"online", "offline"\}$ визначає стан пристрою у момент часу t.

Множина, що визначає параметри таблиць гібридного програмно-конфігурованого пристрою:

$$Tables_i(t) = \{RTable_i(t), CAMTable_i(t), ARPTable_i(t), OFTables_i(t)\}, \quad (3.16)$$

де $RTable_i(t)$ - розмір таблиці маршрутизації;

$CAMTable_i(t)$ - розмір таблиці комутації;

$ARPTable_i(t)$ - розмір таблиці ARP;

$OFTables_i(t) = \{OFTableil_{i1}(t), OFTableil_{i2}(t), \ldots, OFTableil_{ik}(t)\}$ - множина розмірів k таблиці OpenFlow.

Мережні порти гібридного програмно-конфігурованого пристрою в момент часу t мають наступні параметри та характеристики:

$$port_{ij}(t) = [Qin_{ij}(t), Buf_{ij}(t), Qout_{ij}(t), Cashe_{ij}(t), macALC_{ij}(t), ACL_{ij}(t), RouteMap_{ij}(t), Features_{ij}(t), VLAN_{ij}(t), Shaping_{ij}\text{-}(t)\}, \quad (3.17)$$

де $Qin_{ij}(t) = \{QinType_{ij}(t), QinLength_{ij}(t), QinMaxLength_{ij}(t)\}$ – параметри вхідної черги, що включають в себе тип, довжину та максимальну довжину черги;

$Buf_{ij}(t)$ – розмір виділеної пам’яті буфера обробки;

$Cashe_{ij}(t) = \{CacheState_{ij}(t), CacheSize_{ij}(t)\}$ - стан та розмір кешу оброблювача;

$Qout_{ij}(t) = \{QoutType_{ij}^{<q>}(t), QoutLength_{ij}^{<q>}(t), QoutMaxLength_{ij}^{<q>}(t)\}$ – множина параметрів вихідних черг, що включає тип, довжину і максимальну довжину черги; q - індекс конкретної черги, кількість черг залежить від налаштування на конкретному інтерфейсі;

$macACL_{ij}(t) = \{macACLin_{ij}(t), macACLout_{ij}(t)\}$ - кількість записів вхідних та вихідних фільтрів 2 рівня;

$ACL_{ij}(t) = \{ACLin_{ij}(t), ACLout_{ij}(t)\}$ - кількість записів вхідних та вихідних фільтрів 3 рівня;

$RouteMap_{ij}(t) = \{RouteMapin_{ij}^{<r>}(t), RouteMapout_{ij}^{<r>}(t)\}$ - множини модифікаторів: $RouteMapin_{ij}^{<r>}(t) = \{ACLin_{ij}^{<r>}(t), Actionin_{ij}^{<r>}(t)\}$ – параметри вхідних модифікаторів, $RouteMapout_{ij}^{<r>}(t) = \{ACLout_{ij}^{<r>}(t), Actionout_{ij}^{<r>}(t)\}$ - параметри вихідних модифікаторів; $\{ACLin_{ij}^{<r>}(t), ACLout_{ij}^{<r>}(t)$ - кількість записів у відповідному списку, $Actionin_{ij}^{<r>}(t)$, $Actionout_{ij}^{<r>}(t)$ - одна з дій з пакетом (установка DSCP, заміна TTL, IP-адрес, встановлення вихідних інтерфейсів, фільтрація тощо), $r \in [0...65535)$ - індекс запису в *RoutMap,* поточна кількість записів залежить від конкретного *RoutMap;*

$VLAN_{ij}(t) = \{VLANMode_{ij}(t), VLANList_{ij}(t)\}$ - параметры VLAN, $VLANMode_{ij}(t) \in \{"trunk","access"\}$ - тип інкапсуляції, $VLANList_{ij}(t) = \{VLANID_{ij}^{v}\}$, $VLANID_{ij}^{v} \in [1,2...4096)$ - список VLAN, допущених до комутації на порту;

$Shaping_{ij}(t) = \{ShapeType_{ij}(t), ShapeBW_{ij}(t)\}$ - стан та розмір обмеження смуги пропускання на вхід та вихід, $ShapeType_{ij}(t) \in \{"shaping","policing"\}$, $ShapeBW_{ij}(t)$ - обмеження смуги пропускання, кбіт/с;

$Features_{ij}(t) = \{FeatureType_{ij}^{<f>}, FeatureState_{ij}^{<f>}(t)\}$ –: множина активованих функцій гібридного програмно-конфігурованого пристрою, що включають функції пристрою і їх стан:

$$FeatureType_{ij}^{<f>}(t) \in \left\{ \begin{matrix} "no\ switchport", "switchport\ port-security", "ip\ arp \\ nspection", "ip\ dhcp\ snooping", "channel-group", \\ "no\ shutdown", "spanning-treeportfast" \end{matrix} \right\}, \quad (3.18)$$

де *no switchport* - маршрутизований порт;

switchport port-security - використання безпечних MAC-адрес;

ip arp inspection - захист від атак з використанням протоколу ARP;

ip dhcp snooping - захист від атак з використанням протоколу DHCP;

channel-group - агрегація каналів EtherChannel;

no shutdown - стан порту;

spanning-tree portfast - режим пересилання Spanning-Tree.

Стан кожної *f*-тої функції визначається параметром $FeatureState_{ij}^{<f>}(t) \in \{"on","off"\}$.

Множина $Z(t) = \{Traffic(t), Device(t)\}$ - це набір поєднань значень параметрів параметрів у момент часу *t*, наприклад, певний патерн трафіку з певною інтенсивністю та експоненціальним розподілом часу між пакетами, нульові значення черг, завантаження процесора, пам'яті, розмірів таблиць, налаштування за замовчуванням усіх портів комутатора без додаткових функцій.

Необхідно скласти такі набори значень параметрів, за яких буде можлива інтерполяція проміжних значень із необхідною точністю. Множина *Z(t)* у загальному випадку є точковою обмеженою множиною, оскільки всі параметри, крім часу між пакетами, є дискретними або бінарними. За кожним параметром

існує діапазон можливих значень, отриманий або з документації на обладнання (наприклад, максимальний розмір таблиць), або заданий вручну, виходячи зі статистики досліджуваних випадків. Оскільки кожному певному набору *j* вхідних параметрів *Z* відповідає один і тільки один набір вихідних параметрів, описаних у пункті 3.3.2, то необхідно виявити ступінь впливу кожного вхідного параметра на результат. При цьому виникає проблема: деякі вхідні параметри починають впливати тільки в поєднанні з певними значеннями інших параметрів. Наприклад, залежність часу опрацювання пакета від кількості рядків фільтра може виникати тільки за кількості цих рядків понад те, що може поміститися в кеш у поєднанні з метаданими кожного потоку. Тому необхідно перевірити, як поєднання пар вхідних параметрів впливає на вихідний результат. Надалі залежні параметри будуть об'єднані в групу для зменшення кількості поєднань.

3.2.2 Структурна модель вихідних параметрів гібридного програмно-конфігурованого обладнання

Вихідними параметрами *Y(t)* є результати вимірювань часових і кількісних характеристик за певного поєднання вхідних параметрів *Z(t)* . Ці параметри можна розділити на:

1) Тимчасові параметри, які є результатом опрацювання вибірки вимірювань: перші *k* моментів часу проходження пакетів крізь комутатор

$$Latency(Z(t)) = \{\mu_{1..k}\,[L(t)]\}; \tag{3.19}$$

2) Кількісні параметри:

2.1) параметри вихідних черг

$$Qout_{ij}(t) = \begin{Bmatrix} QoutType_{ij}^{<q>}(t), QoutLength_{ij}^{<q>}(t), QoutDrop_{ij}^{<q>}(t), \\ QoutCurDelay_{ij}^{<q>}(t), QoutMaxLength_{ij}^{<q>}(t) \end{Bmatrix}, \tag{3.20}$$

де $QoutType_{ij}^{<q>}(t)$ - тип вихідної черги;

$QoutLength_{ij}^{<q>}(t)$ - розмір вихідної черги;

$QoutDrop_{ij}^{<q>}(t)$ - кількість відкинутих пакетів;

$QoutCurDelay_{ij}^{<q>}(t)$ - поточна затримка проходження крізь чергу за поточної швидкості обробки пакетів (залежить від швидкості лінії та мінімальних інтервалів між вихідними пакетами);

$QoutMaxLength_{ij}^{<q>}(t)$ - максимальний розмір вихідної черги;

2.2) статистика по кожній конкретній черзі конкретного порту - кількість вхідних і вихідних пакетів у черзі, кількість відкинутих пакетів, середня затримка;

2.3) параметри вхідних черг

$$Qin_{ij}(t) = \{QinLength_{ij}^{<q>}(t), QinDrop_{ij}^{<q>}(t), QinMaxLength_{ij}^{<q>}(t)\}, \quad (3.21)$$

де $QinLength_{ij}^{<q>}(t)$ - розмір вхідної черги;

$QinDrop_{ij}^{<q>}(t)$ - кількість відкинутих пакетів;

$QinMaxLength_{ij}^{<q>}(t)$ - максимальний розмір вхідної черги;

2.4) зліпок метрик пристрою після проведення експерименту

$$State'_i(t) = \{ Proc_i(t), Mem_i(t), Flow_i(t), Intr_i(t), Ctx_i(t)\}. \quad (3.22)$$

Оскільки механізми вихідних черг описані практично у всіх системах моделювання дуже докладно, інформація про їхній стан буде потрібна тільки під час складання кінцевої моделі.

Стан вхідних черг найбільш важливий для опису причин зміни затримок пакетів усередині пристрою, якщо виключити вплив вихідних черг. Тимчасові характеристики і стан черг будуть основними параметрами, що переносяться в модель у вигляді результату відношення $F(t):Z(t) \rightarrow Y(t)$.

Якщо вимірювання проводитимуться на точковій множині $Z(t)$, то $Y(t)$ також буде точковою множиною. У моделях здебільшого потрібно використовувати безперервні множини, що визначають набори параметрів часових характеристик, оскільки заздалегідь невідомий набір вхідних параметрів. Тому необхідно отримати безперервні множини за допомогою обраних методів інтерполяції.

3.3 Методики автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання

Для дослідження гібридного програмно-конфігурованого пристрою може бути використана методика апаратної фіксації часу (рисунок 3.2).

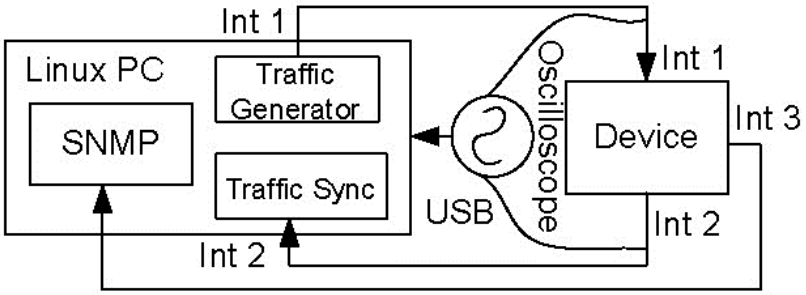


Рис.3.2. Вимірювання затримки пакета методикою апаратної фіксації часу

Ця методика передбачає дослідження гібридного програмно-конфігурованого пристрою на трафіку різних інтенсивностей і розмірів пакетів. Результати при цьому вимірюються осцилографом і записуються в базу даних для подальшого використання при розрахунку затримки, як і дані про використання пам'яті, процесора, вхідних буферів і вихідних черг. Оскільки метою експерименту насамперед є вивчення адекватності поведінки затримки оброблення пакета, особливо залежно від навантаження на процесор, доцільно використовувати осцилограф з USB виходом для точного вимірювання різниці в часі виходу пакета з генератора і виходу пакета з комутатора. При цьому необхідно контролювати вихідну чергу, яка може внести спотворення в вимірювання.

На рисунку 3.3 наведено схему експериментального дослідження мережевого гібридного програмно-конфігурованого обладнання за допомогою методики на основі синтезу віртуальної інфраструктури, яка полягає в наступному.

Для експериментального дослідження збирається схема, яка складається з досліджуваного пристрою, опціональних допоміжних комутаторів у режимі моста (для Cisco це ввімкнення всіх портів у *bridge-group* зі встановленим *protocol ieee*),

а також Linux-сервера з мережевими картами з апаратною обробкою трафіку (для зниження використання процесора під час обробки трафіку). Трафік генерується в контейнері Docker 1, приймається в контейнері Docker 2, дамп трафіку записується з часовими мітками точністю до мікросекунд.

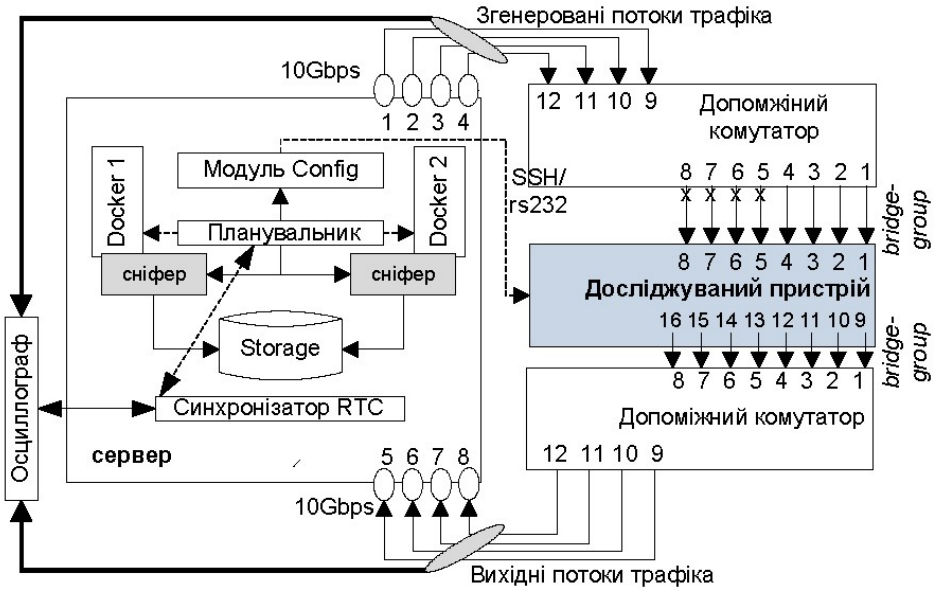


Рис.3.3. Схема експериментального дослідження мережевого гібридного програмно-конфігурованого обладнання

Планувальник запускає різні схеми поширення трафіку через увімкнення нових інтерфейсів (через пряме керування пристроєм), варіює інтенсивність генератора трафіку, запитує дані про стан черг, втрачені пакети, завантаження пам'яті, процесора, переривання, спрацьовування кешу. На основі фіксації параметрів трафіку і часових зрізів стану пристрою створюється набір зліпків станів, параметрів пристрою, генераторів трафіку і відповідні до них вибірки часу затримки з осцилографа і дампи вхідного і вихідного трафіку з синхронізованим часом. У процесі зняття показань затримки опрацювання фрейма динамічно формується план дослідження, що складається з розрахованих параметрів генерації трафіку із заданими параметрами закону розподілу часу між фреймами, переліку включених елементів конфігурації пристрою, кількості задіяних інтерфейсів. Для формування нового рядка плану використовуються дані попередніх рядків і

результати дослідження таким чином, щоб вивчити весь діапазон задіяних параметрів із необхідною деталізацією.

Для дослідження роботи віртуального комутатора в різних умовах необхідно точно виміряти час опрацювання фрейму і вивчити вплив завантаження процесора на продуктивність, а також вплив конфігурації комутатора на вихідні характеристики [105]. На рисунку 3.4 показано логічну схему експериментального дослідження мережевого пристрою із зазначенням можливих шляхів трафіку.

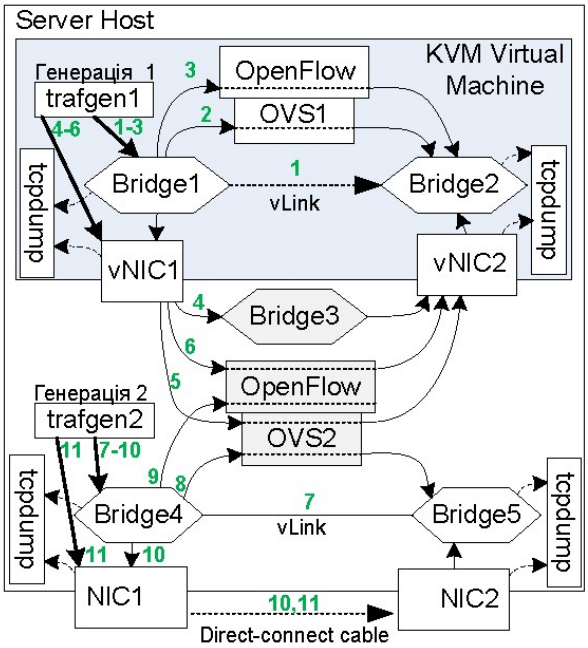


Рис.3.4. Логічна схема експериментального дослідження мережевого пристрою

Відповідно до можливих шляхів трафіку 1-11 обрано такі конфігурації:

1) Вивчення роботи з'єднання через Linux Bridge всередині віртуальної машини (шлях 1) і на обладнанні (шлях 4).

2) Вивчення роботи з'єднання через міст Open vSwitch всередині віртуальної машини (шлях 2) і на обладнанні (шлях 5).

3) Вивчення роботи з'єднання через міст Open vSwitch у режимі OpenFlow з різною кількістю встановлених правил, як усередині віртуальної машини (шлях 3), так і на хостовій машині (шлях 6).

4) Вивчення роботи з'єднання Bridge (шлях 7), Open vSwitch (шлях 8), Open vSwitch + OpenFlow (шлях 9) на хостовій машині.

5) Вивчення роботи з'єднання через зовнішні мережеві карти (шлях 11) і через Bridge і зовнішні мережеві карти (шлях 10).

Оскільки віртуальний комутатор під час передачі пакетів не має затримки на серіалізацію, передачу в лінію, передачу по кабелю, то затримки в мережевій картці під час використання NFV можуть становити суттєву частку від загальної затримки за низьких навантажень [100].

3.4 Алгоритм автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання

Перед проведенням першого експериментального дослідження необхідно встановити налаштування пристрою за замовчуванням і відключити всі налаштування, пов'язані з додатковою функціональністю. Початковий набір параметрів трафіку (вираз 3.9) задається як:

$$Traffic_0^{<s>} = \{ \tau_0^{<s>}, PSize_0^{<s>}, Type0, DSCP0, VID0, L20, L30, L40 \}, \quad (3.23)$$

де $\tau_0^{<s>}$ = F ($\lambda_0^{<s>}$) , $PSize_0^{<s>}$ = {64,768,1500} , $Type_0$ = '*unicast*', DSCP0 = 0, VID_0 = 1, $L2_0$ = $\{ProtoL2_0 = ' ethernet', SrcL2_0 = ' pc_mac_ 1', DstL2_0 = ' pc_mac_2'\}, L3_0 = \{ProtoL3_0 = 'IPv4', SrcIP_0 = 'pc_ip_1', DstIP_0 = '$pc ip 2} , $L4_0 = \{ProtoL4_0 = 'UDP', SrcPort_0 = '0', DstPort_0 = '0'\}$, $F(\lambda_0^{<s>})$ – функція розподілу часу між пакетами, яка використовується генератором трафіку. Інші параметри функції задаються окремо.

Вихідні параметри можна поділити таким чином:

1. Параметри, що задаються явно: параметри генерації трафіку, усі налаштування комутатора;

2. Параметри, на які може бути здійснено непрямий вплив: завантаження процесора, пам'яті, переривання, розмір черг, відкинуті пакети, ймовірність потрапляння в різні місця фільтрів, різні черги;

3. Параметри, на які впливу бути не може: максимальні значення довжин черг, максимальна кількість рядків у списках, об'єми доступної пам'яті та буферів.

Серед параметрів першого виду можна виділити параметри, спільне використання яких дає інший результат, ніж сума результатів їх використання окремо:

- від розміру пакета залежить час передавання в канал передавання даних, розмір буфера і пам'яті, що залишився;
- від DSCP залежить ймовірність потрапляння в конкретні модифікатори, класифікатори, черги і відповідно дії, що здійснюються над цим пакетом;
- від VLAN залежить, чи пройде пакет фільтрацію VLAN-фільтрами, і на які порти буде проведена комутація;
- від адрес L2, L3 і полів L4 залежить імовірність потрапляння в модифікатори, класифікатори, фільтри;
- від часу між пакетами залежить довжина черг, навантаження на всі компоненти пристрою.

Час обробки пакета в гібридному програмно-конфігурованому обладнанні залежить не тільки від задіяних функцій і розмірів таблиць і черг, а й від можливості процесора приділити час на опрацювання всіх необхідних для цього пакета процесів. Більшість комутаторів має механізм обробки паралельних процесів на основі linux/unix-подібного ядра, деякі комутатори також мають апаратні модулі обробки частини даних, наприклад TCAMTable, апаратний кеш. Linux-подібне ядро обробляє пакет у мережевому стеку, який залежить від NUMA-архітектури і від режиму апаратної обробки пакетів мережевою картою. Якщо у мережевої карти немає виділеного апаратного буфера, дані з внутрішньої пам'яті мережевої карти записуються в загальну пам'ять (кільцевий буфер - вхідна черга) за перериванням, про що буде повідомлено ядро. Коли черга обробника подій ядра дійде до обробки пакета, запускаються необхідні обробники всередині іншого контексту (фільтри, модифікатори).

Якщо пам'ять, у якій було записано пакет, належить NUMA-вузлу процесора, що не займається наразі обробкою цього пакета, то додатково може бути викликано

переривання на переміщення даних. Для виявлення залежності часу обробки пакета від параметрів завантаження процесора і ядра системи (час, кількість переривань, кількість зміни контексту, кількість активних процесів) необхідно перед вимірюванням будь-яких параметрів отримати час обробки пакета для завантажень процесора від 0 до 100%, а також отримати дані про час обробки на граничних значеннях кожного параметра окремо.

Нехай $Params = \{Param(d,v)\}$ є розгорткою дерева параметрів пристрою Device, де d - посилання на конкретний параметр (індекс параметра в дереві), v - коефіцієнт для задавання значення параметра від мінімального ($v=0$) до максимального ($v=1$). Кожен кінцевий індекс параметра буде включено до лінійної множини $Idx=\{d\}$ до множини включено тільки кінцеві параметри в кількості, граничній для обраного пристрою. Усі параметри впорядковані за батьківськими вузлами.

Нехай $X(\{x_d\}) = \{Param(d, x_d)\}$, $x_d = [0,1]$ – набір значень параметрів, які будуть задані для дослідження характеристик пристрою.

Алгоритм автоматизованого експериментального дослідження гібридного програмно-конфігурованого обладнання складається з таких кроків.

1. Для визначення базового рівня затримки при ненавантаженому обладнанні на парі портів пристрою $Pair_1 = \{PortIn,PortOut\}_1$, під'єднаних до системи збирання даних, генерують трафік для кожного розміру пакету $PSize_0^{<s>}$ з такою інтенсивністю $\lambda_{baseline}^{<s>}$, щоб виконувалася умова для швидкості порту $\tau_{baseline}^{<s>} \leq 2 \cdot \left(\frac{8 \cdot \left(PSize_0^{<s>} + (header = 18) + (preambule = 8) + (gap = 12)\right)}{\max(bandwidt_j)}\right)$ для швидкості порту $bandwidt_j$. Вимірюється $Latency_{baseline}^{<s>}$, що складається з перших k моментів розподілу часу затримки обробки пакета в пристрої. Якщо на пристрої є порти з різною швидкістю, $\lambda_{baseline}^{<s>}$ вимірюється для таких портів окремо.

2. На пристрої необхідно визначити таку інтенсивність пакетів $\lambda_0^{<s>}$ для кожного розміру пакета $PSize_0^{<s>}$, щоб вихідна черга була не більшою за 1 для мінімізації впливу черги на результат. Для цього $\lambda_0^{<s>}$ ітераційно збільшується до

моменту початку збільшення вихідної черги. Якщо на пристрої є порти з різною швидкістю, $\lambda_0^{<s>}$ вимірюється для таких портів окремо.

3. Нехай пара портів $Pair_1 = \{PortIn, PortOut\}_1$ підключена до системи збору даних. Ця пара буде основною для зняття вихідних характеристик. Якщо на пристрої є порти з різною швидкістю, $Pair_1$ задається для таких портів окремо.

4. Приймемо r рівним кількості потоків через пари портів $\{Pair_j\}r = \{\{PortIn, PortOut\}_j\}_r$, відмінних від $= \{PortIn, PortOut\}_1$, при цьому на один вихідний порт і на один вхідний порт має припадати тільки один потік, кількість пар у множині $\{Pair_j\}_r$ має дорівнювати r. Для першої ітерації приймемо $r=0$.

5. Генерується r потоків на парах $\{Pair_j\}_r$. На парі портів $Pair_1$ генерується трафік для кожного розміру пакета $PSize_r^{<s>}$ з інтенсивністю $\lambda_0^{<s>}$ по обраному закону розподілення F часу між пакетами. З результуючої вибірки виміряних параметрів обчислюється $Latency_r^{<s>}$ і зліпок метрик пристрою $State'^{<s>}_r$.

6. Для кожного вхідного параметра з індексом d, на який можна впливати і який не виключений як незначний, виконується включення і налаштування на максимально можливій межі (x_d=1) с відключенням при цьому всіх інших параметрів ($x_k = 0$, $\forall k \in Id, k \neq d$). Для кожного такого набору параметрів $X(x_d)$ виконується пункт 4, якщо завантаження процесора в жодний з моментів не досягло 100%.

7. Для кожного вхідного параметра x_m, для якого виявлені залежні параметри, $\{x_{m'_u}\}$, $m'_u \in Idx$, $m'_u \neq m$, де $\{x_{m'_u}\}$ - множина включених залежних параметрів $\{m'_u\}$, виконується включення і налаштування на максимально можливій межі одного з залежних параметрів ($u = [1..n]$) з відключенням при цьому всіх інших параметрів:

$$\left\{x_m\{x_{m'_u}\},\{x_k\}\right\}, x_m = 1, x_{m'_u} = 1, x_k = 0, \forall k \in Idx, k \neq m, k \neq m'_u,$$

При цьому $\left|\{x_{m'_u}\}\right| = 1$, $|\{x_k\}| = n - 1$.

8. Для кожного набору значень параметрів $X(\{x_m,\{x_{m'_u}\},\{x_k\}\})$ на парі портів $Pair_1$ генерується трафік для кожного розміру пакету $PSize_r^{<s>}$ з інтенсивністю $\lambda_0^{<s>}$ по обраному закону розподілення F часу між пакетами, при

цьому генеруємо r потоків на парах $\{Pair_j\}_r$. З результуючої вибірки виміряних параметрів обчислюється $Latency_r^{<s>}$ і зліпок метрик пристрою $State'^{<s>}_r$.

9. Кількість включених залежних параметрів з пункту 6 збільшується на одиницю, $|\{x_{m'_u}\}| \leq n$, $|\{x_k\}| = n - |\{x_{m'_u}\}|$. Якщо завантаження процесору на наборі параметрів не досягло 100% і якщо $\{x_k\} \neq \emptyset$ виконується п. 8.

10. Якщо завантаження процесору в жоден із моментів не сягнуло 100% і є ще доступні для підключення до системи збору даних порти, збільшити r на одиницю і перейти до пункту 4.

11. За отриманими даними на першій парі будується функція залежності часу обробки пакета від набору завантажень комутатора.

У результаті виконання алгоритму отримують набори всіх вхідних, виміряних і вихідних параметрів, що утворюють багатопараметричну дискретну функцію з нерегулярною сіткою. Для планування подальшого дослідження пристрою необхідно перевірити:

а) наявність кореляції між параметрами, оголошеними на першому етапі, як залежні. Для цього візьмемо ті набори результуючих експериментальних даних, у яких брала участь кожна пара умовно залежних параметрів u_1 і $u2$: $X\,(\{x_m,\{x_{m'_{u1}}\},\{x_k\}\}) \rightarrow \left\{Latency_r^{<s,u1>},\ State'^{<s,u1>}_r\right\}$ і $X\,(\{x_m,\{x_{m'_{u2}}\},\{x_k\}\}) \rightarrow \left\{Latency_r^{<s,u2>},\ State'^{<s,u2>}_r\right\}$ для $x_m = 1$, $x_{m'_{u1}} = 1, x_{m'_{u2}} = 1$, $x_k = 0$, $\forall k \in Idx, k \notin \{m, m'_{u1}, m'_{u2}\}$. Для цих параметрів виберемо такий $X\,(\{x_m, x_{m'_{u1}}, x_{m'_{u2}}, \{x_{m'_u}\}, \{x_k\}\}) \rightarrow \left\{Latency_r^{<s,u12>},\ State'^{<s,u12>}_r\right\}$, щоб $x_m = 1$, $x_{m'_{u1}} = 1, x_{m'_{u2}} = 1$, $x_{m'_u} = 1, x_k = 0$, $\forall k$, $m_u \in Idx$, $k \notin \{m, m'_{u1}, m'_{u2}, m'_u\}$, $\left|\{x_{m'_u}\}\right| \rightarrow \min$.

З множини вихідних параметрів $\left\{Latency_r^{<s,\{u2,u1,u12\}>},\ State'^{<s,\{u1,u2,u12\}>}_r\right\}$ беруться вибірки експериментальних даних і перевіряються гіпотези про рівність математичних очікувань виходячи з того, що математичне очікування суми незалежних випадкових величин дорівнює сумі їхніх математичних очікувань: $H_0 : \mu_1\left(Latency_r^{<s,u1>}\right) + \mu_1\left(Latency_r^{<s,u2>}\right) = \mu_1\left(Latency_r^{<s,u12>}\right) +$

$\mu_1\left(Latency_r^{<s,u0>}\right)$ де $Latency_r^{<s,0>}$ - вибірка для всіх мінімальних вхідних параметрів $X\left(\{x_k\}\right) \rightarrow \left\{Latency_r^{<s,0>},\ State'^{<s,0>}_r\right\}$, $x_k = 0$, $\forall k \in Idx$.

Якщо нульова гіпотезу не приймається, то це підтверджує залежність параметрів один від одного, якщо гіпотеза приймається, то параметри незалежні і надалі результати їхніх спільних вимірювань будуть прийняті за суму незалежних вимірювань цих параметрів.

3.5 Багатопараметрична симплексна інтерполяція емпіричних залежностей параметрів пристрою

Для отримання адекватної моделі на рівні перших моментів розподілу часу між пакетами достатньо симулювати поведінку функції обробки за затримкою з такими самими моментами розподілів.

У результаті виконання алгоритму отримують набори всіх вхідних, виміряних і вихідних параметрів, що утворюють багатопараметричну дискретну функцію з нерегулярною сіткою. Оскільки лінійна інтерполяція на нерегулярній сітці в просторовій розмірності понад 1 виявляється неоднозначною, завдання лінійної інтерполяції потребує регуляризації побудови симплексного покриття [33].

Відомо, що k-вимірним симплексом $S = S\ (X_0, X_1 ... Xk)$ у n-вимірному просторі називається опукла оболонка $k+1$ точок X_i, $i = 0, ..., k \leq n$, таких, що вектори $X_1 - X_0$, $X_2 - X_0$, ..., $X_k - X_0$ лінійно незалежні. Необхідно побудувати симплексне покриття сітки, що складається з набору симплексів. При цьому обов'язковою умовою є відсутність спільних точок симплексів набору, винятком можуть бути тільки точки, що належать граням симплексів. Також кожна вершина обов'язково має збігатися з одним із вузлів сітки [13].

Для побудови симплексного покриття використовується алгоритм ранжування симплексів за відстанню від опорної гіперплощини [11]. Тут під

опорною гіперплощиною розуміють будь-які гіперплощини, що проходять через n різних точок сітки гіперплощини.

Відстань між будь-якою точкою сітки та i-тою опорною гіперплощиною S_i розраховується з точністю до постійного множника за формулою:

$$L\left(X, S_{X_1 - X_n}\right) = \frac{\Delta\ \left(X, S_{X_1 - X_n}\right)}{V\left(S_{X_1 - X_n}\right)}, \tag{3.24}$$

де $V(S_i)$ = n-1 мірний об'єм симплекса $S_{X_1 - X_n}$,

$\Delta\ \left(X,\ S_{X_1 - X_n}\right)$ - об'єм симплекса з вершинами в точках $X\ (x^1, x^2\ , \dots, x^n)$, $X_1\ (x_1^1, x_1^2, \dots, x_1^n)$, $X_2\ (x_2^1, x_2^2, \dots, x_2^n)$, $X_n\ (x_n^1, x_n^2, \dots, x_n^n)$. Важливо, що точки X_1, $X_2, \dots, X_n$ належать опорній гіперплощині $S_{X_1 \dots X_n}$, де n – кількість ввідних параметрів пристрою (мірність простору). Обчислення об'єму симплексу виконується за формулою:

$$\Delta\ (X,\ S_{X_1 \dots X_n}) = \det \begin{pmatrix} x^1\ x^2\ \dots\ x^n\ 1 \\ x_1^1\ x_1^2 \dots x_1^n\ 1 \\ \dots\ \dots\ \dots\ \dots\ \dots \\ x_n^1\ x_n^2 \dots x_n^n\ 1 \end{pmatrix}. \tag{3.25}$$

Функція відстані $L(X,\ S_{X_1 \dots X_n})$ дає змогу ранжувати точки сітки за відстанню від опорного симплексу.

Нехай N - кількість точок сітки, причому $N>n$. Симплексним ланцюгом називається послідовність симплексів $S_{X_1 \dots x_n, j}$, де $j = N - n$ - номер симплексного ланцюга. Побудова симплексного ланцюга виконується послідовним приєднанням наступного симплексу до вже включених у симплексне покриття. При цьому кожен попередній симплекс не повинен мати перетинів із наступним, а також кожен n-вимірний симплекс ланцюга повинен мати спільну грань максимум із двома іншими симплексами.

Нехай $\Omega(X',k)$ – лінійна функція виду:

$$\Omega(X',k) = \det \left|\Delta_{ijk}\ (X')\right|, j \in [1, N+1], i \in [1, n+1] \tag{3.26}$$

де $\Delta_{ijk}\ (X') = \begin{cases} X_i^j, j > 1 \cap j \neq k \cap i < n+1 \\ X_i^j, j = 1 \cap i < n+1 \\ 1, i = n+1 \end{cases}$.

Тоді на кожному симплексі ланцюга функція інтерполяції набуде вигляду:

$$F\left(X', S_{X_1 \dots X_{n+1}}\right) = \sum_{k=1}^{n+1} \frac{\Omega(X',k)}{\Omega(X_k,k)} Y(X_k), \quad (3.27)$$

де X' - вхідний набір параметрів;

$S_{X_1 \dots X_{n+1}}$ – симплекс, обраний алгоритмом ранжування для точки X';

$Y(X_k)$ – результати вимірювань.

Максимальна кількість можливих сиплексних ланцюгів можна обчислити за формулою:

$$M = \frac{N!}{(N-n)!n!}. \quad (3.28)$$

Оскільки симплексне покриття дає змогу одержати сімейство гіперповерхонь, результатом інтерполяції буде нормована сума функцій за кожним з обраних симплексів із розрахованими заздалегідь коефіцієнтами. Кожен із виміряних параметрів (моменти розподілу затримки) розраховується окремо.

Висновоки до розділу 3

1. В розділі досліджено проблеми синтезу моделей мережевого обладнання та запропоновані методи їх вирішення.

2. Представлені структурні моделі вхідних та вихідних параметрів гібридного програмно - конфігурованого обладнання, що дозволило підвищити показники якості інфокомунікаційних мереж.

3. В розділі запропонована багато параметрична симплексна інтерполяція залежності параметрів пристроїв, за допомогою якої визначені кількісні характеристики параметрів мереж SDN.

РОЗДІЛ 4. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМНО-КОНФІГУРОВАНИХ ПРИСТРОЇВ

4.1 Автоматизована система експериментальних досліджень

4.1.1 Архітектура випробувального стенду

При вивченні роботи програмно-конфігурованих мереж виникає складність, пов'язана із закритістю архітектури комутаторів, що ускладнює аналіз тимчасових характеристик процесів комутації та взаємозв'язку з OpenFlow.

Для вивчення внутрішніх процесів комутації реалізовано програмно- -апаратний комплекс автоматизованої системи експериментальних досліджень (ACEI), що складається з сервера з великою кількістю високопродуктивних мережевих карт з апаратною обробкою пакетів (сервер Xeon Gold 6240, 128Гб ОЗУ, NVMe SSD + Optane DC, чотири мережеві плати /с на базі чотирипортового модуля AOC-STG-b4S) та розробленого програмного забезпечення. Також до складу комплексу на етапі його первинного налаштування входить осцилограф для точного вимірювання часу затримки пакетів при обробці на мережевих картах і зовнішніх пристроях розподілу трафіку (комутаторах) та USB годинник точного часу на основі мікроконтролера з температурною компенсацією із зовнішнім виведенням синхроімпульсів різної форми для осцилографа. Схема автоматизованої системи експериментальних досліджень показано рисунку 4.1.

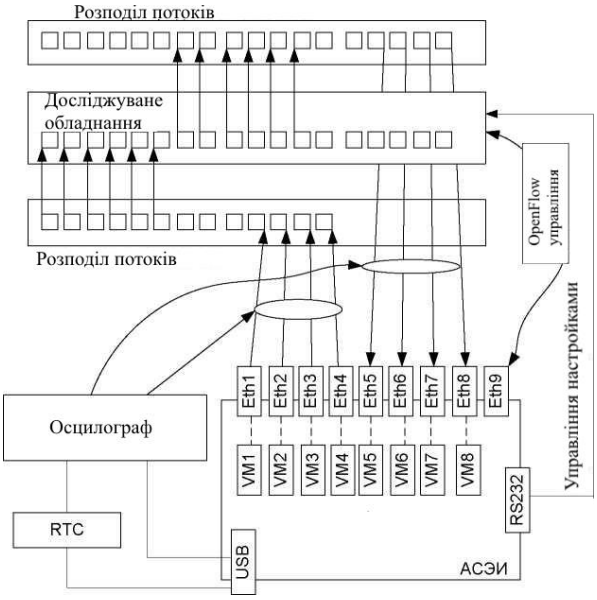


Рис.4.1. Схема автоматизованої системи експериментальних досліджень

Перед початком дослідження з високою точністю визначено параметри розподілу часу між фіксацією пакета на виході з логічного інтерфейсу аналізатором трафіку (на запит до зовнішніх таймерів) і початком генерації преамбули в лінії. Для цього за допомогою осцилографа Tektronix серії DPO4032 (або іншого серії DPO7000) та програмного забезпечення Automated Compliance Software, яке дозволяє з перехопленої осцилограми отримати необроблений сигнал XGMII(10G)/SGMII(1G),відбувається розмітка службових міток за контрольними символами. З розмічених даних за допомогою спеціалізованих блоків аналізу програми Automated Compliance Software вилучається час Start_Of_Packet (SOP) та End_Of_Packet (EOP), а також IDLE, PREAMBLE та TERM. Після вибірки даних про час кожного контрольного символу та зіставлення з синхронізованими мітками RTC (на окремому вході осцилографа та на вході USB/TTL сервера) виходить час, зафіксований ядром від сигналу RTC до команди планувальнику на виведення пакета в мережеву картку (при генерації) або до сигналу планувальнику від мережевої карти про прихід пакета (при прийомі кадру), а також час реального виходу преамбули та старту кадру на фізичному інтерфейсі. Різниця цих часів використовується для попереднього калібрування системи з метою забезпечення точності фіксації часу виходу та входу пакетів.

Потім заміряється час проходження сигналу через пристрій розподілу з вимкненими чергами в режимі комутації без буферизації та відключеними чергами.

Далі досліджуване обладнання встановлюється на стенд до відповідного серверу, програмному комплексу повідомляється виробник і модель комутатора для визначення шаблонів налаштувань та переліку можливостей (заздалегідь були згенеровані за документацією).

Генерація пакетів на кожній віртуальній машині (контейнері) VM={VM1, VM2..VM8 } може відбуватися за нормальним, експонентним, рівномірним або будь-яким окремо реалізованим законом розподілу інтервалів часу між пакетами. Адреса джерела та призначення при цьому може бути як однаковою, так і випадковою. Трафік може генеруватися в стресовому режимі за умови абсолютного завантаження інтерфейсів. Крім того, однією з функцій планувальника ядра Linux є фіксація переривань на відправлення або прийом пакета. Ця можливість пов'язана з наявністю у всіх віртуальних машин загального годинника та загального ядра. Налаштовані фільтри за подіями дозволяють згенерувати переривання для годинника реального часу RTC, які записують зліпок часу в буфер по DMA.

4.1.2 Опис обладнання та ПЗ

Апаратне забезпечення ACEI складається з сервера, комутаторів розподілу, досліджуваного комутатора, кабелів Direct-Attach SFP та Ethernet Cat6 RJ45, а також для підвищення точності вимірювань використовується осцилограф та модуль реального часу з виходом на USB та зовнішній кабель. Сервер побудований на базі платформи Supermicro SSG-6049P-E1CR24H з встановленими 2xXeon Gold 6240, DDR4 128Гб ОЗУ, 4xNVMe SSD 960Gb, з додатковими високопродуктивними мережевими картами 4x10Gb SFP з апаратом G-b4S). Як комутатори розподілу трафіку використовуються Cisco WS-C2960X-48LPD-L з прошивкою IOS 15.2 та підтримкою OpenFlow, кабелі DirectAttach довжиною 0,5 м для мінімізації затримок поширення. Також до складу комплексу на етапі його первинного налаштування входить осцилограф для точного вимірювання часу

затримки пакетів при обробці на мережевих картах і зовнішніх пристроях розподілу трафіку (комутаторах) та USB годинник точного часу на основі мікроконтролера з температурною компенсацією із зовнішнім виведенням синхроімпульсів різної форми для осцилографа. Як операційна система використана ОС Debian 10 з ядром linux-image-rt-amd64 для підтримки м'якого реального часу та eXpress Data Path для забезпечення альтернативного доступу до мережевого стека Linux. Також були проведені попередні налаштування пріоритезації IRQ мережевих карток і жорстке закріплення черг за ядром процесора, що використовувався для ізоляції ядер тільки під обробку мережевого трафіку.

4.2 Екпериментальне дослідження комутатора

4.2.1 Планування експериментального дослідження

Схема для зняття показань виконана відповідно до рисунку 4.1. Як комутатор для розподілу використовувався Cisco WS-C2960X-48LPD-L, як досліджуваний комутатор HP 3500yl-24.

Специфікація досліджуваного комутатора містить відомості про наявність 3000 записів у таблиці TCAM для різних правил, включаючи класифікацію, ACL. Крім цього, стверджується одночасна комутація на швидкості лінії через всі порти. Але при включенні OpenFlow режиму все може змінитись, навіть при реактивному способі завдання правил. У першій частині експерименту для завантаження портів 1-20 трафіком розмір пакета встановлюється в 64 байти (як найбільш ресурсозатратний). Максимальний теоретичний максимум передачі з урахуванням преамбули та інтервалу між фреймами складає 1,488 млн фреймів/с із корисним використанням смуги пропускання 761Мбіт/с. 20 портів теоретично можуть передати 28960000 фреймів / с. У таблиці 4.1 показаний прогресивний план експерименту (матриця Л).

Таблиця 4.1.

План експерименту - матриця Л , тис. пакетів/с

№	Кількість задіяних інтерфейсів					
	1	2	5	10	15	20
1	2	5	38	313	1352	4145
2	38	62	219	1043	3380	8679
3	219	313	793	2731	7291	16464
4	793	1043	2185	6084	14139	28959

Трафік UDP генерується утилітою mgen за згенерованим скриптом посилки пакетів з необхідним законом розподілу. Знімання надісланого та прийнятого трафіку здійснюється утилітою tcpdump із записом на tmpfs диск та опцією запису тимчасових міток із мікросекундами. Кожен пакет має різний номер UDP порту, що дозволить відстежити його затримку простими утилітами порівняння linux .

Для автоматизації аналізу параметрів комутатора розроблено скрипт, який за планом експерименту запускає генератор трафіку на необхідну кількість портів з необхідними параметрами розподілу, збирає дампи трафіку (використовувалося обмеження – записувалися лише заголовки та час кадру) [5]. Кожен пакет посилається на наступний порт UDP для порядку, щоб швидко знайти цей пакет на приймальній стороні в разі різного порядку отримання. Запис відбувається на файлову систему TMPFS із заздалегідь виділеним ділянкою пам'яті зниження часу виділення нової пам'яті, включеної опцією «noatime» з високою швидкістю запису і майже нульовим часом пошуку. Після кожного циклу експерименту із запису дампів витягується час і номер UDP порту по кожному інтерфейсу, потім розраховуються різниці часу в мікросекундах. Потім файли запису трафіку очищалися для звільнення місця нового циклу.

При тестуванні з'ясувалося, що наявна материнська плата має обмеження 40Гбіт/с на всю шину з мережевими картами, тому максимальна смуга пропускання

на виході з двох мережевих карток склала не більше 23,5 Гбіт/с з урахуванням накладних витрат [54].

4.2.2 Експериментальне дослідження комутатора на стенді

Для вимірювання затримки на мережному інтерфейсі проведено експеримент зі з'єднанням безпосередньо входів та виходів 10Гбіт/с (таблиця 4.2). У таблиці 4.3 показано затримки, що виникають у службовому комутаторі Cisco.

Таблиця 4.2

Пряме з'єднання мережевих карт

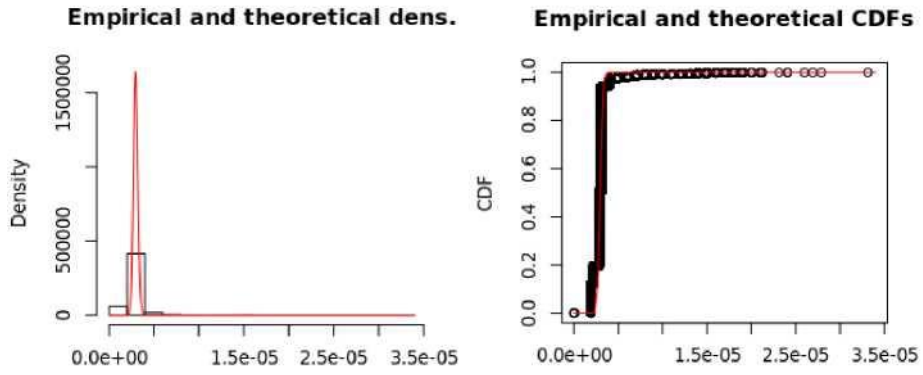
Параметр	Затримка пакетів		
Розмір пакета, байт	64	512	1518
Затримка, мкс	2.04	2.19	2.21

Таблиця 4.3.

Затримки у службовому комутаторі Cisco

Параметр	Затримка пакетів		
Розмір пакета, байт	64	512	1518
Затримка передачі 10 ^ 10 Гбіт/с, мкс	4.27	4.89	5.96
Затримка передачі 10 ^ 1 Гбіт/с, мкс	7.21	12.1	18.41

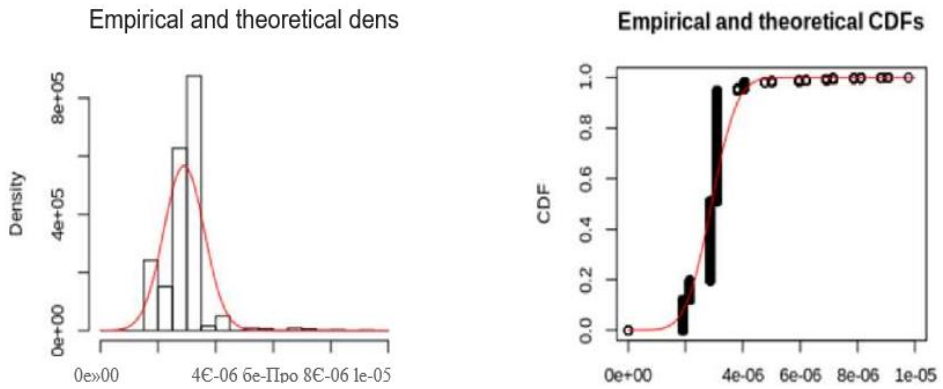
Аналіз трафіку проводився засобами мови R, що має широкий вибір бібліотек статистичного аналізу. На рисунку 4.2 показано розподіл затримки проходження пакета в комутаторі при максимальному навантаженні 28 960 пакетів/з розміром 64 байт.



а) б)

Рис.4.2. Гістограма (а) та функція розподілу (б) бета-розподілу (лінія) та вибірки часу затримки на всьому діапазоні даних

Довгий хвіст розподілу утворюється через попадання окремих пакетів у вихідну чергу, і, відповідно, значно більше часу очікування посилки [4]. При збільшенні деталізації в діапазоні [0; 10 -5] видно явні неоднорідності (ступеня), хоча обсяг вибірки понад 1 млн. (рисунок 4.3). Ці затримки відповідають роботі систем кешування та апаратних черг.



а) б)

Рис.4.3. Гістограма (а) та функція розподілу (б) нормального розподілу (лінія) та вибірки часу затримки в діапазоні до 1е-5

Потім проведення експерименту за планом, описаним у таблиці 4.1, показав такі значення (таблиця 4.4).

Таблиця 4.4

Затримки в досліджуваному комутаторі HP 3500yl у звичайному режимі, мкс

№	Кількість задіяних інтерфейсів											
	Звичайний режим						Режим OpenFlow					
	1	2	5	10	15	20	1	2	5	10	15	20
1	6,91	6,91	6,94	7,19	7,4	7,55	267,43	261,98	267,23	272,62	278,37	273,52
2	6,94	7,05	7,08	7,33	7,55	7,7	268,59	267,28	272,62	277,93	284,01	278,96
3	6,98	7,19	7,22	7,48	7,7	7,86	270,14	272,59	278,01	283,61	289,66	284,75
4	7,01	7,33	7,36	7,63	7,85	8,51	271,3	277,9	283,4	289,3	295,3	308,3

Середня довжина черги, що входить у всіх експериментах, крім останнього (20 інтерфейсів, 28959 тис. пакетів/с, черга - 3 пакети), склала 1 пакет. Далі був проведений експеримент по 4 рядку матриці Л для Openflow реактивному режимі з роздільною здатністю 0,5 мкс. За графіком поведінки затримки (рисунок 4.4) видно певні (практично дискретні) рівні затримки, що найчастіше зустрічаються - близько 4,5,10,11 мкс. Значення вище 10 мкс – очікування у вихідній черзі.

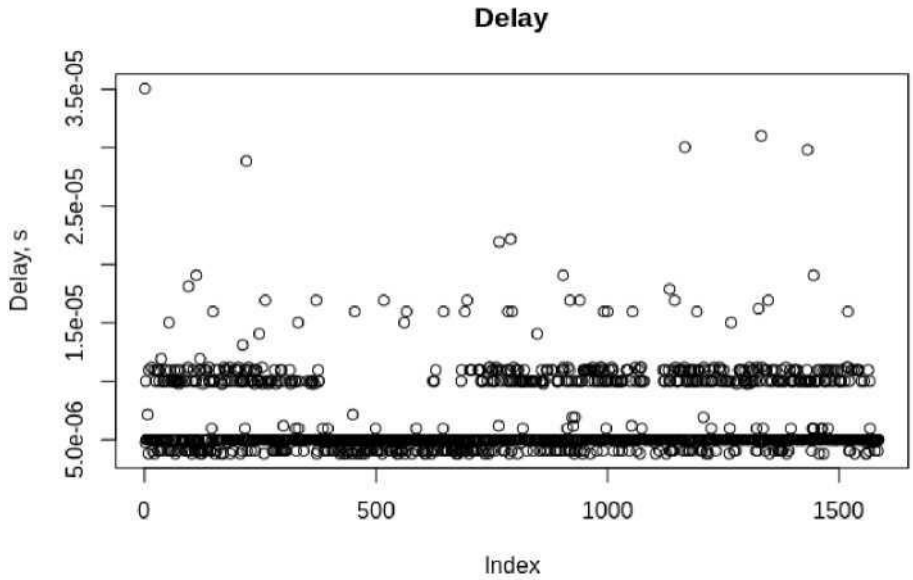


Рис.4.4. Графік затримки всередині комутатора

4.2.3 Сформована модель OMNeT++ для досліджуваного комутатора

В результаті роботи програмно-апаратного комплексу після дослідження гібридного комутатора HP3500yl на стенді в автоматичному режимі сформовано його імітаційно-аналітичну модель у системі OMNeT++. Як очевидно із загальної

схеми синтезованої моделі, показаної рисунку 3.15, зазначений комутатор може бути досліджено моделі як у режимі OpenFlow, і під час використання традиційної комутації. Крім блоків, що забезпечують різні режими комутації, модель містить дві таблиці інтерфейсів, одна з яких використовується лише модулем OF_Switch. Блок ContentBasedClassifier використовується для поділу вхідного трафіку при обробці OpenFlow з VLANID. Також у модель комутатора включено розроблений модуль RealDelay, що імітує затримки, що вносяться при проходження пакетом крізь комутатор.

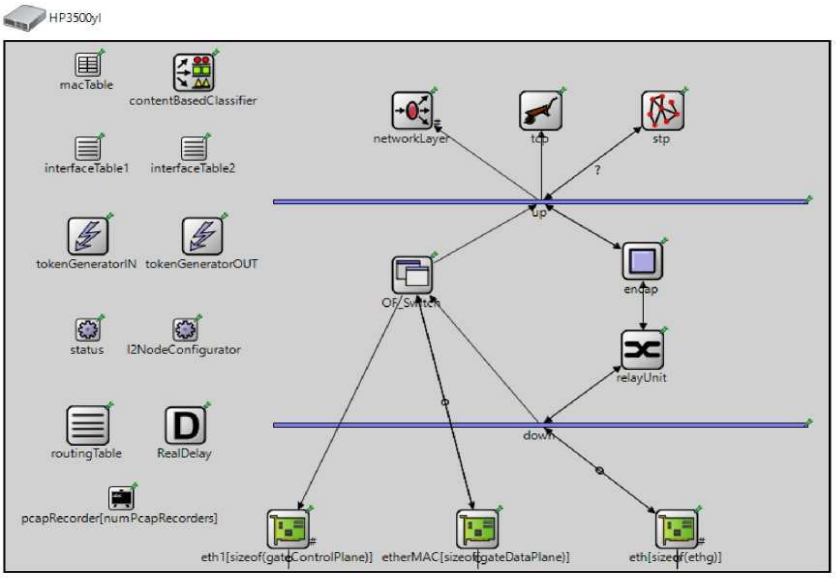


Рис.4.5. Синтезована модель досліджуваного комутатора HP3500yl

Для зовнішнього управління поведінкою блоків, призначених для класифікації вхідних пакетів, забезпечення QoS, а також формування трафіку відповідно до бажаного профілю, модель комутатора HP3500yl включені модулі tokenGeneratorIN і tokenGeneratorOUT, загальні схеми яких показані на рисунках 4.6 і 4.7 відповідно.

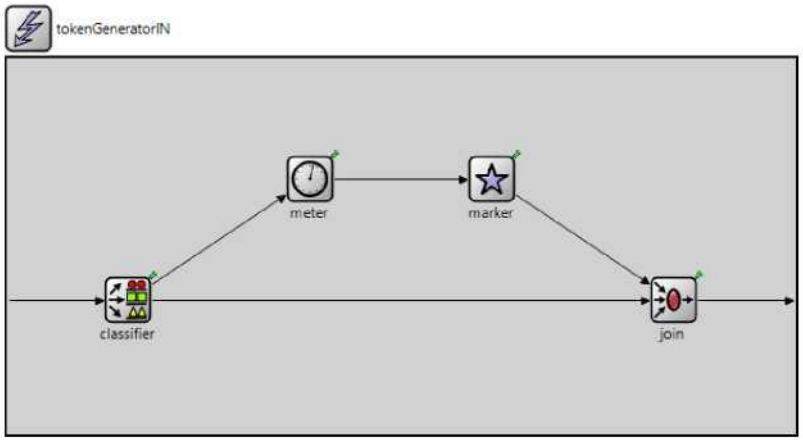


Рис.4.6. Схема модуля tokenGeneratorIN

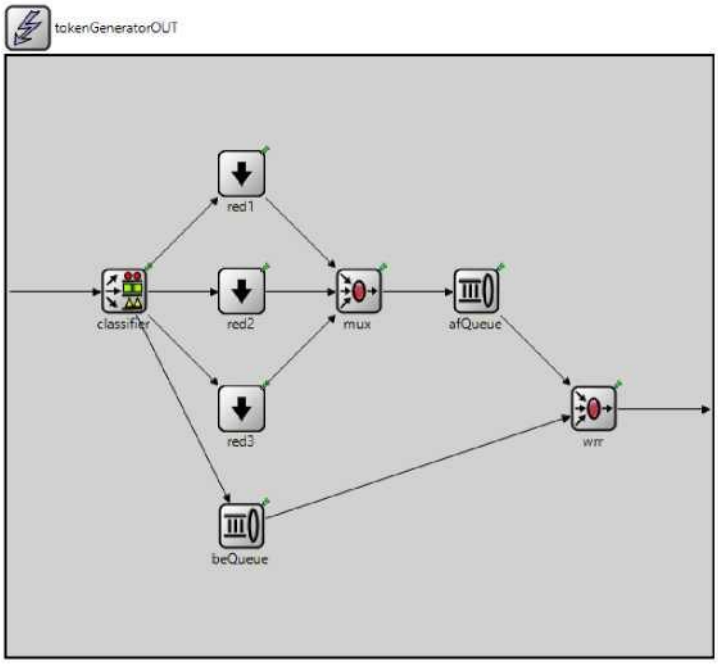


Рис.4.7. Схема модуля tokenGeneratorOUT

Для порівняння даних модель комутатора OMNeT++ тестувалася на 4-му рядку плану експерименту (таблиця 4.1) як у звичайному, так і в режимі OpenFlow. Було задіяно стільки генераторів експоненційного трафіку, скільки інтерфейсів вимагалося щодо експерименту. Статистика середньої затримки розраховувалася по дамп подій пакетів, збережених у скалярний файл. Результати дослідження комутатора та його моделей показані у таблиці 4.5.

Таблиця 4.5

Затримки в досліджуваному та модельному комутаторах, мкс

Пристрій	Кількість задіяних інтерфейсів 1 Гбіт/с					
	1	2	5	10	15	20
Досліджуваний комутатор у звичайному режимі	7,01	7,33	7,36	7,63	7,85	8,51
Досліджуваний комутатор у режимі OpenFlow	271,30	277,90	283,40	289,30	295,30	308,30
Модельний комутатор у звичайному режимі	7,00	7,15	7,25	7,38	7,50	8,25
Модельний комутатор у режимі OpenFlow	268,60	278,60	285,23	290,50	298,25	310,02

Середній розмір черги при k=20 становив 3 пакети, дані моделі узгоджуються з даними експерименту в межах 5%.

Як видно з таблиці 4.5, оптимізація роботи з Openflow на цьому комутаторі відсутня і латентність пошуку в таблиці підвищується в кілька разів.

Для оцінки відмінностей між вибіркою з експериментальними даними про затримку і вибіркою, отриманої в експерименті з моделлю за кожною позицією плану експерименту, використовувався статистичний U -критерій Манна-Уітні як менш схильний до впливу виду статистичного розподілу. Для використання непараметричного U -критерію Манна-Уітні наявність нормального розподілу порівнюваних сукупностей не є обов'язковою умовою. Для рівня значущості 0,05 значення U -критерію за кожною позицією перевищило критичне, а це означає, що модель адекватно описує реальний комутатор за критерієм затримки, що виникає при обробці пакетів.

4.3 Експериментальне дослідження комутаторів за різних параметрів трафіку

Для вивчення роботи модельних пристроїв та зіставлення результатів було обрано пристрої, налаштування яких наведено у таблиці 4.6. Версії ПЗ на пристрої встановлено останні на момент експерименту.

Таблиця 4.6.

Устаткування для експерименту

Клас пристрою	Модель обладнання	
	Модель	Налаштування
Комутатор L2	Cisco Catalyst 3750	Single VLAN, RSTP Enabled, CEF Disabled, 2 interfaces
Комутатор OpenFlow-only	HPE Aruba 3800	Single VLAN, OpenFlow Enabled, Reactive mode, Internal controller only, 2 interfaces
Гібридний комутатор	HPE Aruba 3800	Two VLANs, One-OpenFlow Enabled, Second-OpenFlow disabled, Reactive mode, Internal controller only, 2 interfaces
Комутатор Open vSwitch	v2.11.1, Debian 10, DPDK 18.11.2, Xeon e5 2678 v3	VLANs, One-OpenFlow Enabled, Second-OpenFlow disabled, Pre-defined OpenFlow rules 1 physical interface, 1 virtual interface

Кожен комутатор досліджувався на трафіку різних інтенсивностей і розмірів пакетів, результати вимірювалися осцилографом і записувалися в базу даних SQLite для подальшого використання в модулі розрахунку затримки, як і дані про використання пам'яті, процесора, буферів, що входять, і вихідних черг. Оскільки метою експерименту в першу чергу є вивчення адекватності поведінки затримки обробки пакета, особливо в залежності від навантаження на процесор, використовувався осцилограф з USB виходом для точного вимірювання різниці в

часі виходу пакета з генератора комутатора. При цьому контролювалася вихідна черга, яка могла внести спотворення у виміри.

Для апробації моделей зібрана найпростіша схема з генератора трафіку (UdpHost), в якому реалізований генератор трафіку із заданим законом та параметрами розподілу, розміром пакета, портами тощо. Затримки у вузлі OMNeT++ не обчислюються автоматично, із затримок є інформація про час у черзі, у деяких додатках є наскрізна затримка та внутрішня затримка. Для розрахунку затримки, джиттера та інших характеристик трафіку потрібний запис часу входу і виходу конкретного пакета в кожен модуль, що нас цікавить. Для запобігання впливу черг на вивчення поведінки затримки параметр packetCount у всіх чергах встановлений рівним нулю.

Для забезпечення рівних умов експерименту в моделі та експерименту на обладнанні можна забезпечити два підходи:

1. Використовувати аналогічні параметри генератора трафіку експерименту на обладнанні, якщо це можливо.
2. Використовувати PCAP файли зі збереженим вихідним трафіком експерименту на обладнанні.

Вибрано другий варіант для кращої точності та подальшого використання у різних експериментах.

Стандартний модуль PcapReader не може працювати з абсолютним часом, обчислюючи з нього відносне, а також жорстко задає формат записаного часу з масштабом в мікросекундах (SIMTIME_US). Тому при використанні PCAP файлу, отриманого за допомогою Tcpdump, потрібна попередня обробка часу для приведення часу першого пакета нанівець, а також, якщо потрібно, зміни масштабу часу. Для цього можна використовувати утиліти із комплекту WireShark. За допомогою команди

```
"time0=$(tshark -te -r file.pcap -c 1| cut -d' ' -f1)"
```

або

```
"time0=$(tcpdump -r file.pcap -tt -c1 | cut -d' ' -f1)"
```

необхідно отримати Unix timestamp для першого пакета. У системах Windows можна використовувати Powershell конструкцію

```
"$time0=(tshark -te -r file.pcap -c1 ).Split(" ")[3]».
```

Потім утилітою editcap відняти отриманий час

```
"editcap -F pcap -t -$time0 file.pcap out.pcap"
```

При необхідності з'єднання з реальною мережею (тільки на Linux) відбувається через модуль ExtInterface. При цьому створюється віртуальний інтерфейс veth, який є вхідною і вихідною точкою трафіку в модель. Для цього необхідно додати віртуальні інтерфейси veth, а також додати другий інтерфейс із пари до bridge br0:

```
ip link add eth1-br0 type veth peer name eth1-gw1
brctl addif br0 peer1-br0»,
```

а також налаштувати маршрутизацію та iptables.

Після цього можна генерувати трафік будь-якими утилітами, трафік потрапить у модель через модуль ExtInterface.

Оскільки для повторення експерименту було обрано перший шлях з читанням з файлу PCAP , експеримент був автоматизований скриптом на bash. У цьому кожен новий цикл експерименту виконувався з іншим набором вхідних файлів. Вихідні файли були записані через модуль PcapRecorder кожного інтерфейсу комутатора (модуля EtherEncap). Для порівняння до експерименту додано стандартні моделі EtherSwitch і OF_Switch. Так як вимірювання затримки відбувається від моменту надходження на вхід фізичного порту до моменту виходу з фізичного порту комутатора, враховуються всі затримки, в тому числі і фізично (додається затримка Ethernet каналу), які були відраховані з результату. Стандартна затримка лінії Ethernet для довжини 10 м становить 5 нс, затримка приймання кадру визначається ставленням довжини цього кадру в бітах до швидкості лінії, що вимірюється в біт/с. Для OpenFlow використовувався «Реактивний» режим із заздалегідь збереженими адресами та “action=output:port”.

Таблиця 4.7 показує результати вимірювання затримки на портах CopperEthernet 1 Гбіт/с за різних розмірів пакетів при зазначених параметрах

генерації трафіку (експоненційний закон розподілу часу між пакетами з умовою мінімальної паузи між пакетами) на пристроях та їх моделях.

Таблиця 4.7 .

Результати вимірювання затримки

Пристрій	**Затримка, мкс**		
	Кадр 64 б	***Кадр 512 б***	***Кадр 1518 б***
Устаткування			
Комутатор L2	4.20	4.23	4.30
Комутатор OpenFlow-only	179.51	181.60	208.10
Гібридний комутатор	180.31	182.20	210.01
Комутатор OpenvSwitch	32.99	37.40	47.72
Модельне обладнання			
L2 комутатор із модулем RealDelay	4,07	4,06	4,17
OpenFlow-only комутатор із модулем RealDelay	174,12	174,34	201,86
Гібридний комутатор з модулем RealDelay	174,90	174,91	203,71
OpenvSwitch із модулем RealDelay	32,00	35,90	46,29
Стандартна модель INET4 EtherSwitch	0	0	0
Стандартна модель INET4 OF_Switch	0	0	0

Таблиця 4.7 демонструє, що при використанні стандартної моделі комутатора і відніманні затримок фізичного рівня неможливо досліджувати затримки в мережі. Для моделей, синтезованих за допомогою програмно-апаратного комплексу похибка в середньому не перевищує 10%, при цьому такий високий показник може бути пов'язаний з PCAP форматом та обробкою реальних пакетів канальним рівнем [104].

На рисунку 4.8 показані результати у графічному вигляді.

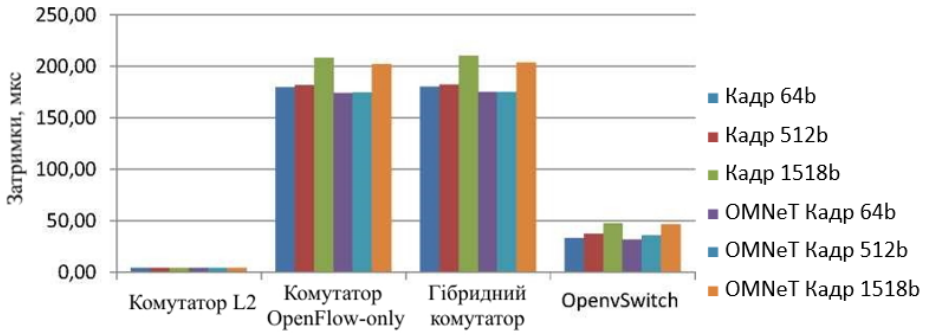


Рис.4.8. Результати вимірювання затримки пакетів

Для дослідження впливу включення додаткових функцій на пристрої було вивчено поведінку обладнання до та після включення функції ACL. Таблиця 4.8 показує результати вимірювання затримки на інтерфейсах різної пропускної спроможності різних інтенсивності при розмірі кадру 1518 б.

Таблиця 4.8.

Результати експериментального вимірювання затримки на обладнанні, мкс

		Тис. фреймів у сік (розмір 1518 б)					
		38	219	793	38	219	793
Пристрій	Режим	Без додаткових налаштувань			З включенням ACL		
Huawey S5850-48SQ6	10 Гб/с	1,05	1,05	1,05	1,06	1,07	1,07
Huawey S5850-48SQ6	1 Гб/с	2,05	2,06	2,06	2,08	2,08	2,08
HP 5406zl	10 Гб/с	281,72	291,24	301,09	376,94	389,68	402,85
HP 5406zl	1 Гб/с	316,99	335,75	355,63	504,64	534,52	566,16
HP5406zl	100 Мб/с	313,79	326,59	339,92	441,82	459,84	478,61
Aruba 3800E	10 Гб/с	210,13	213,30	216,52	241,85	245,51	249,21
Aruba 3800E	1 Гб/с	222,33	237,63	253,98	375,30	401,12	428,71
Aruba 3800E	100 Мб/с	215,18	216,82	218,47	231,54	233,30	235,07
OpenvSwitch	1 Гб/с	47,37	48,05	48,75	54,24	55,02	55,82
OpenvSwitch	100 Мб/с	48,88	49,52	50,18	55,33	56,06	56,80

Графічна інтерпретація результатів показана рисунку 4.9.

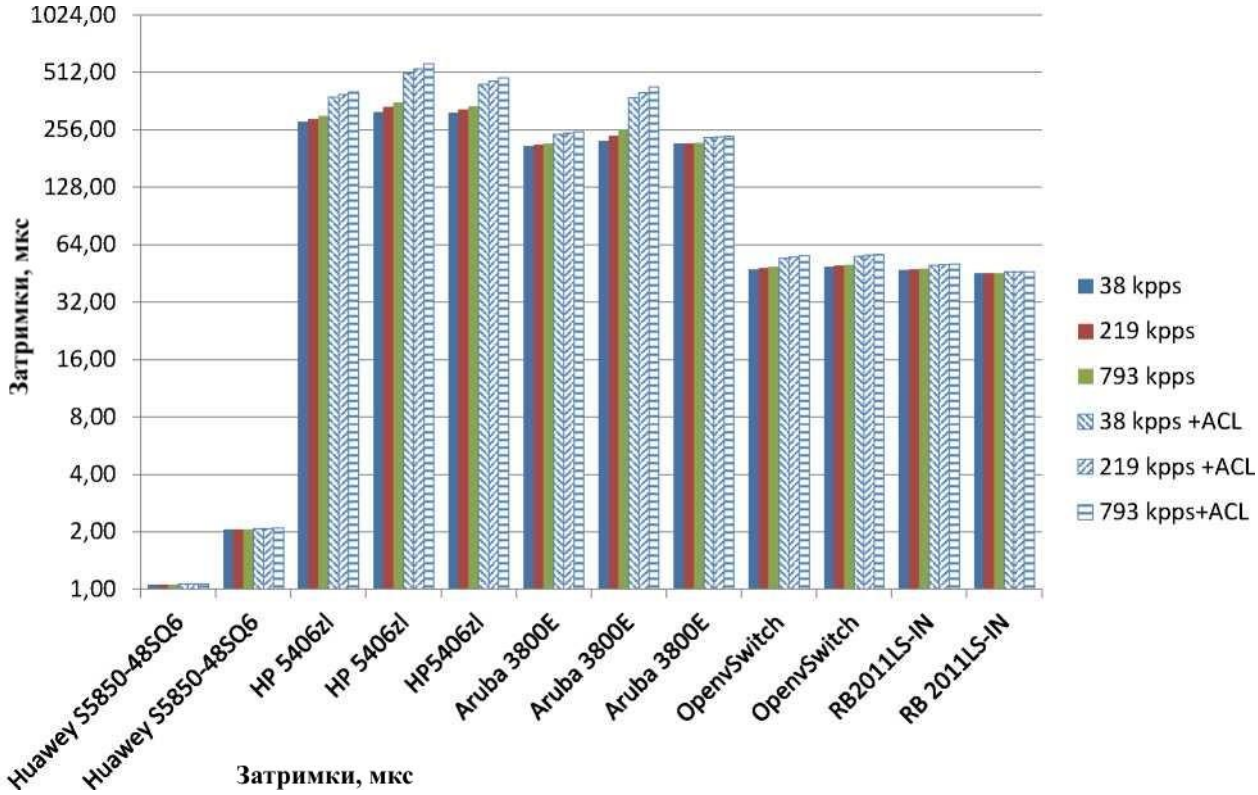


Рис.4.9. Результати вимірювання затримки пакетів при включенні ACL

Рисунок показує, що навіть один включений додатковий параметр на одному пристрої при одній інтенсивності трафіку може істотно збільшувати затримку. Поєднання кількох ввімкнених параметрів може призвести до збільшення затримки в кілька разів.

4.4 Експериментальне дослідження комутаторів за різних конфігураціях

Для проведення експериментального дослідження розроблено та реалізовано алгоритм, який запускає серію експериментів за заданими заздалегідь мережевими інтерфейсами. При виборі режиму OVS+Openflow генерується необхідна кількість правил із випадковими адресами та номерами портів. Для генерації необхідна рандомізація MAC адрес для зниження впливу кеша [45].

Кожен цикл експерименту проходив розміру пакетів 64 байт, з параметрами генерації пакетів, зазначених у таблиці 3.9. Ці параметри отримані шляхом

попереднього експерименту та пошуку таких інтенсивностей генерації, при яких втрата пакетів не перевищує 1% загальної кількості. Конфігурації з'єднань задані відповідно до можливих шляхів трафіку 1-11, показаних на рисунку 4.4.

Таблиця 4.9.

План досліджень – інтервал між пакетами при генерації трафіку

Шлях	Назва конфігурації	Місце проведення експерименту	
		Host	KVM
1,7	Linux Bridge	1,10,50	1,10,50
2,8	OVS	1,10,50	1,10,50
3,9	OVS+OpenFlow 100 правил	1,10,50	10,50,100
3,9	OVS+OpenFlow 1000 правил	1,10,50	10,50,100
4,10	Linux Bridge для зовнішньої мережі	1,10,50	10,50,100
5	OVS для зовнішньої мережі	1,10,50	1,10,50
6	OVS+OpenFlow для зовнішньої мережі, 1, 100, 1000	1,10,50	1,10,50
10,11	Пряме з'єднання	1,10,50	1,10,50

Попередній експеримент показав, що продуктивність у різних конфігураціях відрізняється у кілька разів.

Експеримент проводився на сервері 2x Intel Xeon E5 6 ядер, 64Гб ОЗУ DDR3-ECC, 2x SAS SSD RAID0. Перша серія експериментів запускалася на хостовій ОС Ubuntu server 20.04. Друга серія на віртуальній машині KVM з такою самою конфігурацією операційної системи та мережевими адаптерами e1000 та комутацією OVS. Третя серія експериментів проходила на віртуальній машині VMware ESXi з аналогічною конфігурацією та мережним комутатором vSwitch.

Для генерації трафіку використовувалася утиліта trafgen з рандомізацією MAC адреси одержувача та відправника, а також параметром інтервалу часу між пакетами в мкс. Розмір виміряних затримок при різних конфігураціях представлено рисунку 4.10.

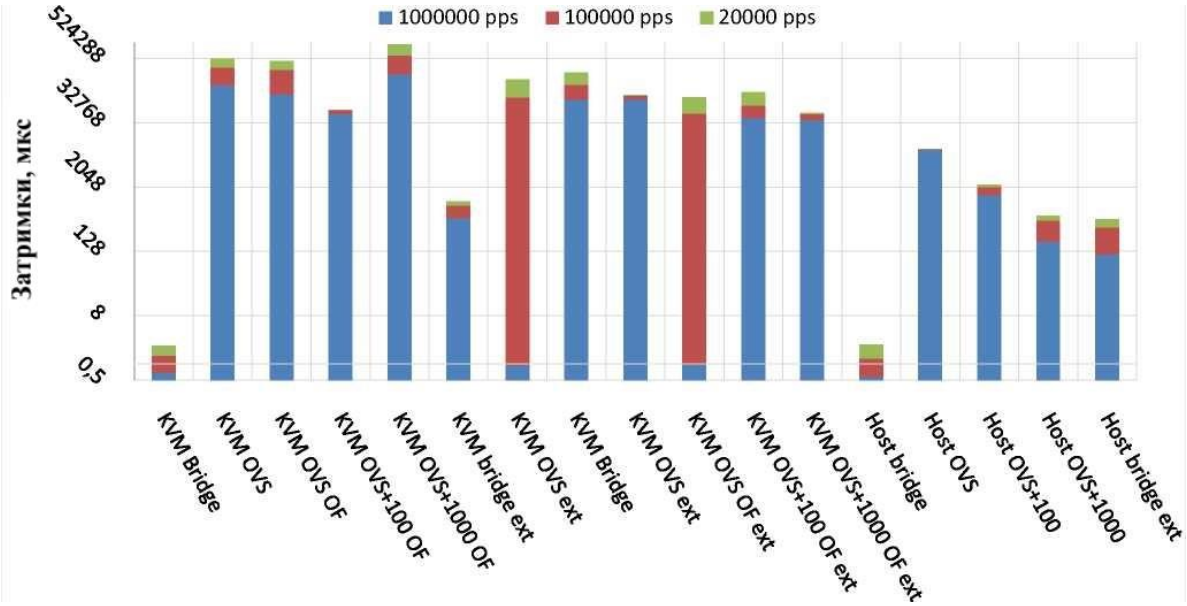


Рис.4.10. Вимірювання затримок при різних варіантах конфігурації

Втрати пакетів розміру 64 байт при використанні OpenFlow у віртуальному середовищі показані на графіку (рис. 4.11).

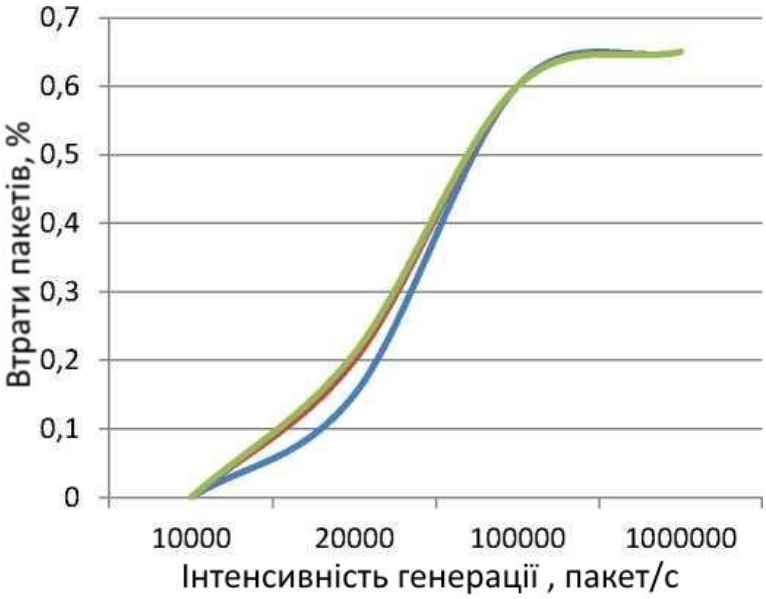


Рис.4.11. Втрати пакетів OVS+OpenFlow у KVM

На апаратному сервері втрат пакетів немає при будь-яких поєднаннях параметрів генерації трафіку, і навіть при розмірі пакета понад 500 байт в KVM. Затримки в режимах, де починаються втрати пакетів, збільшуються експонентно (рисунок 4.12).

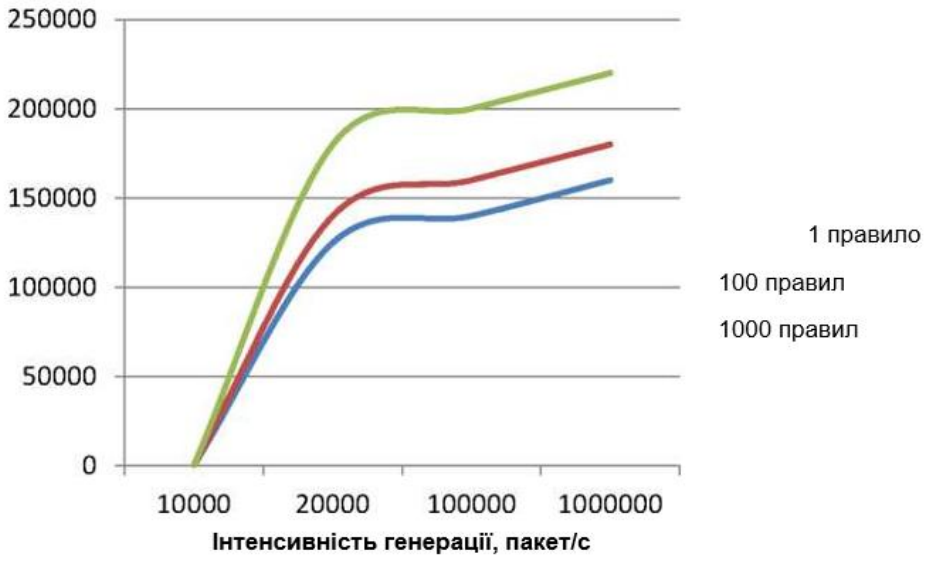


Рис.4.12. - Затримки OVS+OpenFlow у KVM

Аналіз процесу проходження трафіку через дві мережеві карти віртуальної машини через комутатор віртуалізації показав, що втрати на серіалізацію та емуляцію відсилання пакета становлять близько 200-300 мкс через комутатор OVS та 120 мкс при використанні Linux Bridge. Але максимальна продуктивність Linux Bridge при комутації віртуальної машини на пакетах 64 байт нижче майже вдвічі, ніж OVS. Втрати пакетів при використанні Linux Bridge починалися при швидкості генерації від 100 000 пакетів/с по 64 байт, тоді як OVS не втрачав пакети і при швидкості генерації 100 000 пакетів/с по 64 байт.

4.5 Дослідження параметрів мережі з прикладу експериментального сегмента центру обробки даних

Вивчення роботи модельних пристроїв, об'єднаних у мережу, проводилося з прикладу нового експериментального сегмента центру обробки даних.

Сегмент реалізований для забезпечення високопродуктивної обробки даних та створення приватної хмари для навчальних та виробничих завдань, у тому числі для запису та обробки відеопотоків, збереження дампів трафіку Wi-Fi клієнтів, створення навчальних віртуальних машин, завдань обробки великих даних. До складу сегмента ЦОД входять два комутатори, які грають роль рівня розподілу. Ці комутатори з'єднуються каналами 10 GbE з кожним із семи комутаторів доступу. Кожен сервер з'єднується з двома комутаторами доступу двома каналами 10 GbE (сегмент зберігання даних та сегмент комутації) та двома каналами 1 GbE (управління). Оскільки система зберігання побудована на базі CEPH, заснованому на дисках тих самих серверів, що використовуються для запуску всіх віртуальних машин, вимоги до швидкості та затримки сегмента зберігання дуже високі. При записі в сховищі відбувається дуплікація на кілька серверів, крім цього відбувається регулярне перебалансування та шардинг даних, що суттєво збільшує трафік. Сегмент комутації використовується для міжмережевої взаємодії віртуальних машин із зовнішніми споживачами та між собою. При використання розподілених нейромереж та звернення до сховищ на базі Hadoop, які фізично зберігають дані на окремих накопичувачах тих самих серверів, вимоги до продуктивності сегмента комутації також мають обмеження знизу.

На рисунку 4.13 показано топологію мережі.

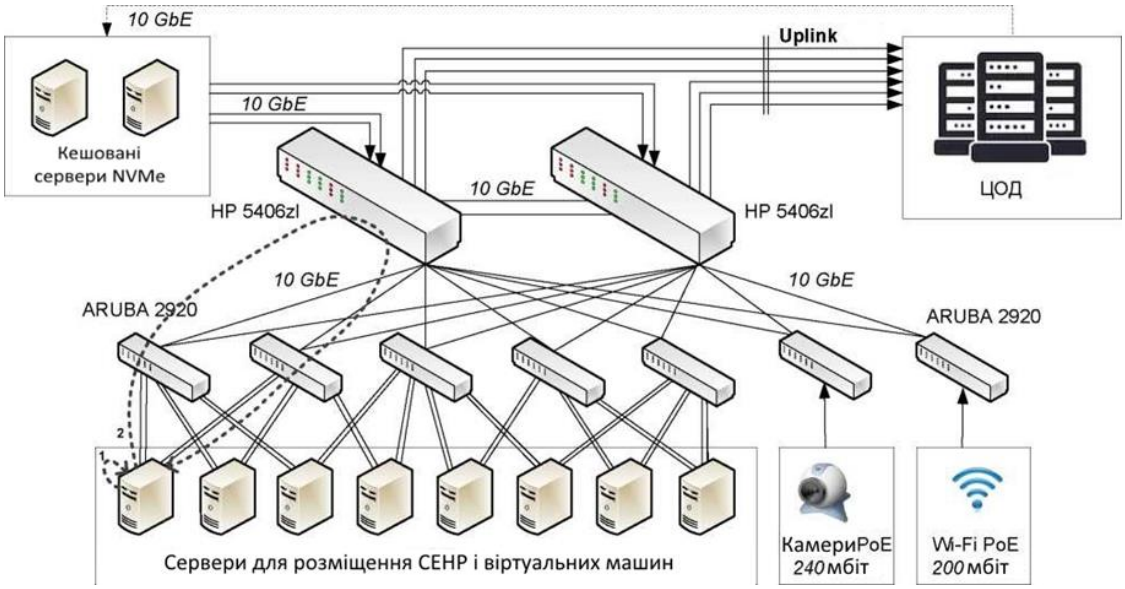


Рис.4.13. Топологія експериментального сегмента ЦОД

На момент досліджень в експериментальний сегмент включав вісім серверів (Xeon Gold 6240, 128Гб ОЗУ, NVMe SSD + Optane DC, чотири мережні плати 10 Гбіт/с на базі чотирипортового модуля AOC-STG-b4S) для розміщення CEPH та віртуальних машин, два сервери роботи тільки як системи зберігання, сім комутаторів доступу (HPE ARUBA 2920) і два комутатори розподілу (HP ProCurve Switch 5406zl з п'ятьма модулями HP J9309A ProCurve 4-Port 10GbE SFP+ zl). Для зв'язку з основним ЦОД використовується канал агрегації, що складається із шести 10GbE каналів. Віртуальні машини працюють під управлінням системи оркестрації OpenNebula, як віртуальні комутатори використовується Open vSwitch в режимі гібридної комутації з проактивним режимом OpenFlow зменшення латентності обробки. На даний момент хмара містить понад 350 віртуальних машин.

Віртуальні машини поділяються на групи, між якими має бути виключено мережевий доступ. Таких груп більше ста, причому у групах можуть бути свої маршрутизатори, DHCP-сервера, що повторюються VLAN та VxLAN. Тому замість створення накладених мереж за допомогою вбудованих OpenNebula можливостей на базі VxLAN прийнято рішення ізолювати трафік через OpenFlow.

Оскільки сервери використовуються одночасно для запуску віртуальних машин та для забезпечення системи зберігання CEPH, для роботи віртуальних

машин виділено дві мережеві карти, які підключаються до Open vSwitch, а дві інші мережні карти використовуються для підсистеми зберігання. Топологія взаємодії сховища CEPH передбачає обмін трафіком між усіма вузлами. Для зменшення латентності прийнято рішення про використання обох 10 GbE карток у режимі балансування по одержувачу. Для забезпечення одночасної роботи та ізоляції трафіків віртуальних машин та підсистем зберігання, а також балансування за паралельними маршрутами використовується OpenFlow на комутаторах доступу та розподілу.

Кожен сервер містить усередині комутатор Open vSwitch, вузол віртуалізації під керуванням оркестратора, вузол сервера CEPH із зберіганням на локальних дисках, інтегрованих із загальною підсистемою зберігання (рисунок 4.14).

За допомогою програмної частини ACEI досліджено затримки в експериментальному сегменті ЦОД у випадкові проміжки часу. Під час зняття зліпків стану мережі відбувався запис дампів трафіку для подальшого використання на моделях.

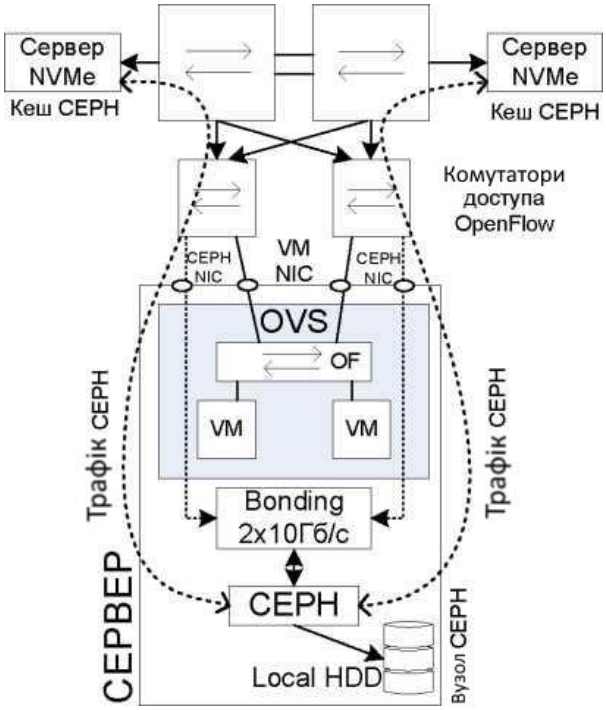


Рис.4.14 - Схема організації сервера

На основі топології, показаної на рисунку 4.13, у системі моделювання OMNeT++ сформовано модель даної мережі з використанням стандартних засобів та проведено дослідження затримок у модельній мережі (рисунок 4.15).

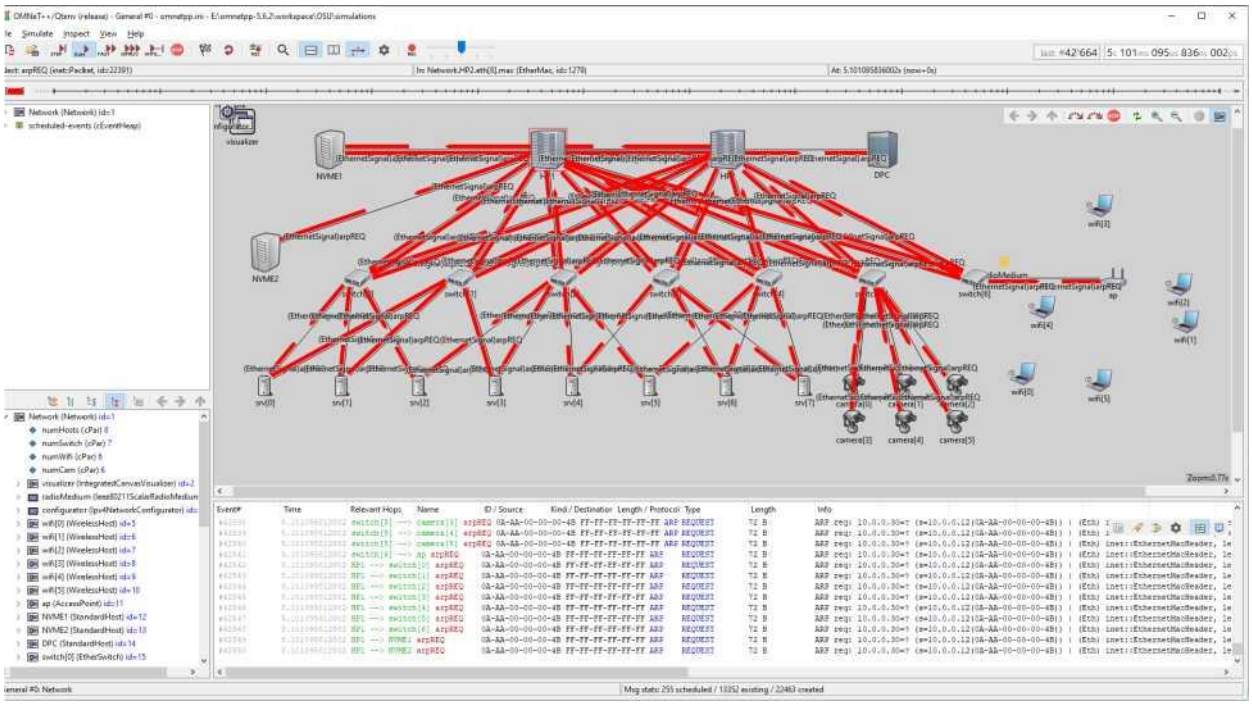


Рис.4.15 - Модель мережі з використанням стандартних засобів OMNeT++

За допомогою розробленого програмно-апаратного комплексу проведено дослідження всіх мережевих пристроїв, що входять до складу досліджуваної мережі та синтезовані моделі цих пристроїв. У сформованій раніше моделі мережі стандартні блоки мережевих пристроїв OMNeT++ замінені на отримані. Проведено дослідження затримок у новій моделі мережі.

На рисунку 4.16 наведено графічну інтерпретацію результатів дослідження затримки у мережі без використання черг (на устаткуванні черги було мінімізовано), її моделі, отриманої за допомогою стандартних засобів системи моделювання OMNeT++, а також моделі, отриманої засобами розробленого програмно-апаратного комплексу. Дослідження проводилося за різних сценаріїв роботи для різних інтенсивностей генерації трафіку.

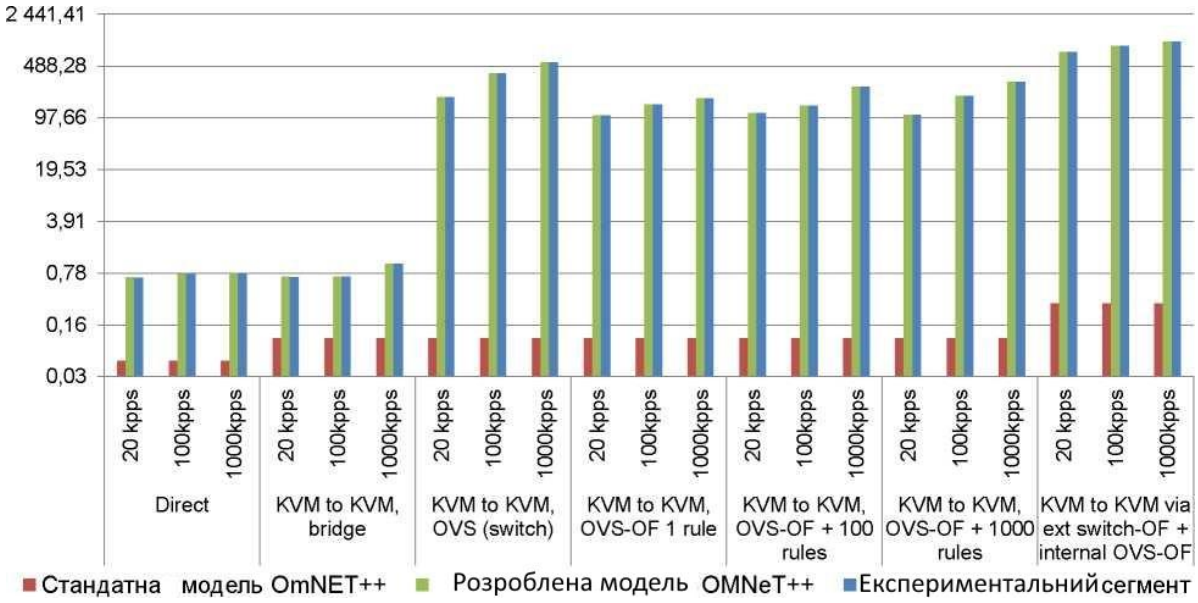


Рис.4.16 - Результати дослідження затримки у мережі та її моделях

У таблиці 4.10 показані зведені результати вимірювання затримок у реальній мережі, моделі мережі, отриманої стандартними засобами системи моделювання, та моделі мережі, отриманої з використанням засобів розробленого програмно-апаратного комплексу.

Таблиця 4.10.

Середня затримка, мкс

Режим роботи	Інтенсивність	Реальна мережа	Стандартна модель мережі OMNeT++	Розроблена модель мережі OMNeT++
Direct	20 kpps	0,679	0,051	0,684
	100kpps	0,757	0,051	0,762
	1000kpps	0,769	0,051	0,774
KVM to KVM, bridge	20 kpps	0,687	0,102	0,697
	100kpps	0,696	0,102	0,706
	1000kpps	1,042	0,102	1,052
KVM to KVM, OVS (switch)	20 kpps	185,019	0,102	185,029
	100kpps	390,493	0,102	390,504
	1000kpps	548,669	0,102	548,679
KVM to KVM, OVS-OF 1 rule	20 kpps	104,913	0,102	104,923
	100kpps	146,284	0,102	146,294
	1000kpps	179,147	0,102	179,157
KVM to KVM, OVS-OF + 100 правил	20 kpps	114,155	0,102	114,165
	100kpps	140,678	0,102	140,689
	1000kpps	253,802	0,102	253,812
KVM to KVM, OVS-OF + 1000 rules	20 kpps	107,267	0,102	107,277
	100kpps	193,589	0,102	193,599
	1000kpps	298,667	0,102	298,677
KVM до KVM за допомогою ext switch-OF + internal OVS-OF	20 kpps	754,384	0,306	754,394
	100kpps	920,220	0,306	920,230
	1000kpps	1044,120	0,306	1044,130

Згідно з отриманими результатами, при використанні вхідних у поставку OMNeT++ моделей комутатора неможливо досліджувати вплив часу обробки кадру комутатором на загальну затримку передачі пакета в мережі. Порівняння

результатів дослідження моделі, отриманої з використанням засобів розробленого програмно-апаратного комплексу, із затримками експериментального сегмента ЦОД, показав збіг на аналогічних інтенсивностях генерації фреймів у межах 5%.

4.6 Розрахунок завадостійкості сигнальних сузір'їв

Оскільки статистичний метод багатократної імітації сумарного вектора сигналу і шуму з подальшим прийняттям рішення за правилом Котєльнікова потребує для достатньої точності не менше, ніж *20/P** спроб імітації кожного сигналу (де *P**-середнє значення ймовірності помилки розрізнення сигналів при заданому співвідношенні сигнал/шум), статистичний розрахунок в області великих співвідношень сигнал/шум (де властивості маніпуляційних кодів проявляються найбільш повно) потребує дуже великого числа вимірювань. Тому розрахунок проводився аналітично, методом інтегрування двовимірної функції розподілу щільності ймовірності значень суміші сигналу з білим шумом в областях сигналів. Такий розрахунок дозволяє визначити як імовірності помилок розрізнення кожної пари сигналів сузір'я, так і ймовірності помилок у кожному двійковому розряді кодів сигналів. Як відомо, функції розподілу ймовірностей миттєвих значень сигналу, що складається з інформаційного сигналу з координатами *(x,y)* та завади, яка відповідає моделі білого шуму, описуються (4.1):

$$\omega(x) := \frac{1}{\sqrt{2 \cdot x \cdot \sigma_x}} \cdot e^{-\left[\frac{(\bar{x}-x)^2}{2\cdot(\sigma_x)^2}\right]};$$
$$\omega(y) := \frac{1}{\sqrt{2 \cdot x \cdot \sigma_y}} \cdot e^{-\left[\frac{(\bar{y}-y)^2}{2\cdot(\sigma_y)^2}\right]}. \qquad (4.1)$$

Враховуючи те, що параметри розподілу ймовірностей значень шуму не залежать від координатної системи приймача:

$$\sigma_e=\sigma_x=\sigma\ . \qquad (4.2)$$

Тоді інтегральні функції розподілу будуть мати такий вигляд:

$$F(X) := \int_{-\infty}^{X} \omega(x)dx = \frac{1}{2\cdot x}\cdot \int_{-\infty}^{\frac{(\bar{x}-x)}{\sigma}} e^{\frac{z^2}{2}} dz = \Phi(Z) = \Phi\left[\frac{(\bar{x}-x)}{\sigma}\right];$$
$$F(Y) := \int_{-\infty}^{Y} \omega(y)dy = \frac{1}{2\cdot x}\cdot \int_{-\infty}^{\frac{(\bar{y}-y)}{\sigma}} e^{\frac{z^2}{2}} dz = \Phi(Z) = \Phi\left[\frac{(\bar{y}-y)}{\sigma}\right], \qquad (4.3)$$

де *Ф(Z)* - інтеграл Лапласа.

Для сигнальних сузір’їв *QAM* областями сигналів (крім периферійних сигналів сузір'я) є квадрати, центри яких співпадають з сигнальними точками, а сторони перпендикулярні до відрізків, що з’єднують сигнальні точки. В такому разі, згідно основних положень теорії ймовірностей, інтегральне значення ймовірності прийняття рішення за правилом Котельникова на користь сигнальної точки *“J”* при передачі через канал з білим шумом сигнала, що відповідає точці *”I”* визначається сумісною ймовірністю:

$$P_{ij}=Px_{ij}Py_{ij}=(\Phi((x_a-x_i)/\sigma)-\Phi((x_в-x_i)/\sigma))(\Phi((y_c-y_i)/\sigma)-\Phi((y_в-x_i)/\sigma)), \qquad (4.4)$$

де:

Px_{ij}-імовірність попадання проекції сигнальної точки на виході каналу з білим шумом, в діапазон значень координат *x*, який належить околу точки *J*, якщо на вході каналу - сигнал I; Py_{ij}-імовірність попадання в діапазон значень координат *y*, який належить околу точки *J*, проекції сигнальної точки на виході каналу з білим шумом, якщо на вході каналу - сигнал *I;* $\Phi((x_a-x_i)/\sigma)$- значення функції Лапласа, коли аргументом є відношення різниці координат *x* точок *A* та *I* до дисперсії миттєвих значень білого шуму *σ* ($\sigma=\sqrt{Pз}$, де *Рз* - середня потужність завади).

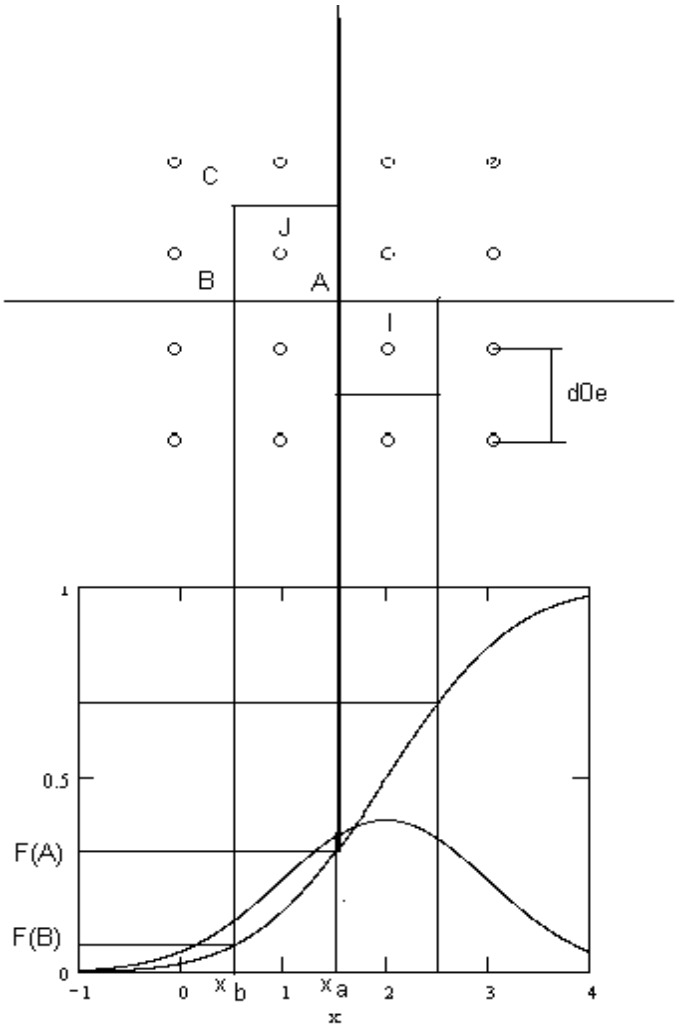


Рис.4.17 Функції розподілу ймовірностей амплітуд суміші сигналу з гаусовським шумом в області сигналу сузір'я *QAM16*

Позначивши $\Delta x=x_j-x_i$, $\Delta y=y_j-y_i$ та врахувавши, що $x_a-x_i=0.5d0e$, $x_в-x_i=0.5d0e$, $y_c-y_i=0.5d0e$, $y_в-y_i=0.5d0e$ отримаємо:

$$P_{ij}=(\Phi((\Delta x+0.5d0e)/\sigma)-\Phi((\Delta x-0.5d0e)/\sigma))(\Phi((\Delta y+0.5d0e)/\sigma)-\Phi((\Delta y-0.5d0e)/\sigma)). \quad (4.5)$$

Формула (4.5) описує залежність ймовірності прийняття рішення про прийом не периферійної точки сузір´я *QAM* з координатами $(x_i+\Delta x,\ y_i+\Delta y)$ при передачі сигналу (x_i,y_i) через канал з білим шумом.

Для периферійних точок розрахункова формула дещо спрощується. Дійсно, якщо у (4.5) одна із змінних x_a, $x_в$, $y_в$ або y_c, відповідно до положення точки, приймає значення $\pm\infty$, тоді один з інтегралів Лапласа приймає значення “0” або “1”. Незалежно від того, яке саме значення приймає один з інтегралів Лапласа (“0” або “1”) величина однієї з імовірностей Px_{ij} або Py_{ij} приймає те ж саме значення:

$$Px_{ij}=\Phi((\Delta x+0.5d0e)/\sigma)=1-\Phi((\Delta x-0.5d0e)/\sigma);$$

$$Py_{ij}=\Phi((\Delta y+0.5d0e)/\sigma)=1-\Phi((\Delta y-0.5d0e)/\sigma); \qquad (4.6)$$

$$P_{ij}= Px_{ij}\, Py_{ij}.$$

Отже, в обох випадках, геометричній симетрії сигнального сузір’я відповідає симетрія ймовірностей помилок.

Для кутових периферійних точок розрахункова формула спрощується ще більше. В такому випадку, у (4.5) значення $\pm\infty$ приймає не одна із змінних x_a, $x_в$, $y_в$ або y_c, *а* - дві. Тоді значення “0” або “1” приймають два з інтегралів Лапласа. Отже обидві ймовірності Px_{ij} та Py_{ij} приймають значення:

$$Px_{ij}=\Phi((\Delta x+0.5d0e)/\sigma)=1-\Phi((\Delta x-0.5d0e)/\sigma);$$

$$Py_{ij}=\Phi((\Delta y+0.5d0e)/\sigma)=1-\Phi((\Delta y-0.5d0e)/\sigma); \qquad (4.7)$$

$$P_{ij}=(1-\Phi((\Delta x-0.5d0e)/\sigma))(1-\Phi((\Delta y-0.5d0e)/\sigma)).$$

Аналогічно вирішується задача і для сузір'їв *НАР*. Відмінність полягає у змінних межах інтегралів функції розподілу імовірностей що є наслідком шестикутної форми областей сигналів.

Розрахунок проводився окремо для кожного сигнального сузір'я за допомогою системи “MathCAD ® v.7.0 Professional Edition”.

Графіки відношень імовірностей помилок розрізнення сигналів $P_{сигн}$ (усереднених за всіма значеннями у сузір'ї) до ймовірностей помилок у двійкових розрядах P_2 (також усереднених за всіма значеннями у сузір'ї) (див. рис.4.18)

ілюструють ефективність маніпуляційного кодування. Як видно із графіків, маніпуляційні коди сузір'їв *QAM* (неперервні лінії) забезпечують значно більший виграш у завадостійкості, ніж коди сузір'їв *HAP* (пунктирні лінії). Це пояснюється тим, що відношення кількості інверсій до кількості мінімальних евклідових відстаней у сузір'ях *HAP (6/5)* більше, ніж у сузір'ях *QAM (4/4).* Графіки мають дві ділянки: на першій ділянці із ростом співвідношення сигнал/шум серед усього числа помилок розрізнення сигналів зростає відсоток помилок розрізнення сусідніх сигналів сузір'я, хемінгова відстань між якими мінімальна. Отже, перша ділянка характеризує величину виграшу, який забезпечує розроблений маніпуляційний код порівняно з випадковим кодуванням. Друга ділянка характеризує потенційні можливості коду. При цих значеннях співвідношення сигнал/шум усі помилки розрізнення сигналів пов'язані з розрізненням сусідніх сигналів. При співвідношенні сигнал/шум близько 25 дБ ймовірності помилок у сузір'ях *HAP8* і *QAM8* (нижні криві на рис.4.18) зменшуються до нуля.

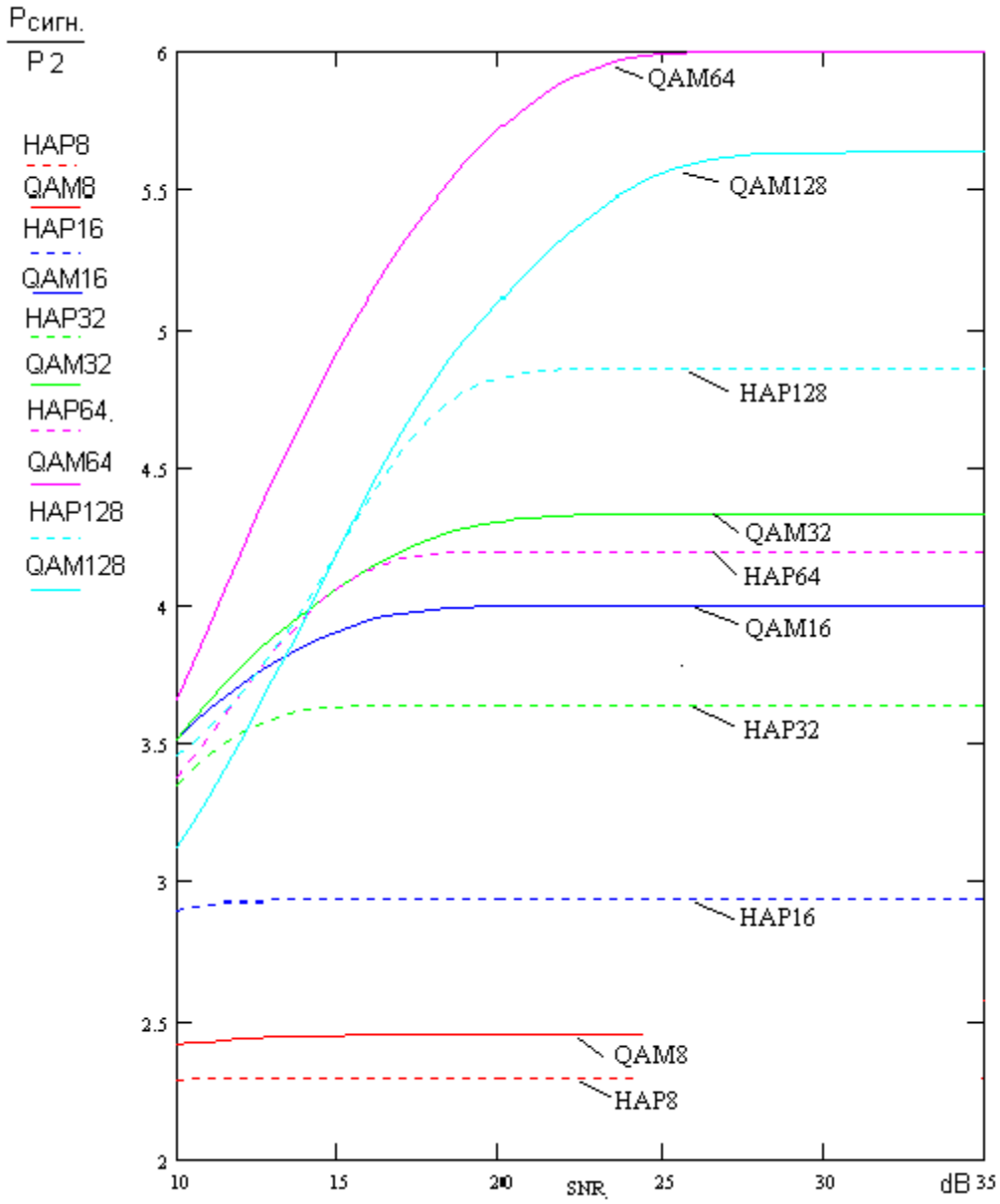


Рис.4.18 Ефективність маніпуляційного кодування

Графіки на рис.4.19 і рис.4.20 характеризують відносний виграш у завадостійкості сузір'їв *HAP* порівняно з сузір'ями *QAM*. На рис.4.19 зображено графіки відношень імовірностей помилок розрізнення сигналів сузір'їв *QAM* P_{cQAM} до ймовірностей помилок розрізнення сигналів сузір'їв *HAP* P_{sHAP}. На рис.4.20 зображено графіки відношень імовірностей помилок у двійкових розрядах сузір'їв *QAM BER_QAM* до ймовірностей помилок у двійкових розрядах сузір'їв *HAP BER_HAP*. Усі величини ймовірностей – усереднені за всіма значеннями у кожному сузір'ї .

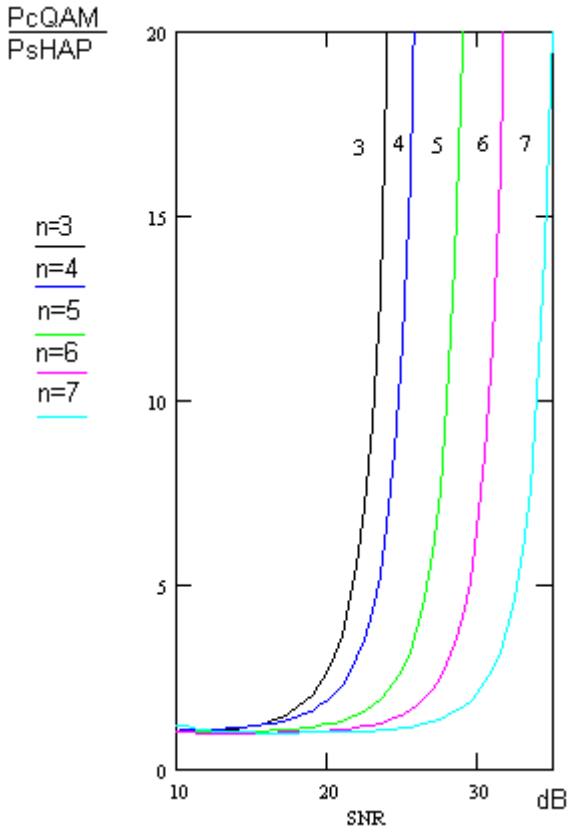


Рис.4.19 Виграш за ймовірністю помилки розрізнення сигналів

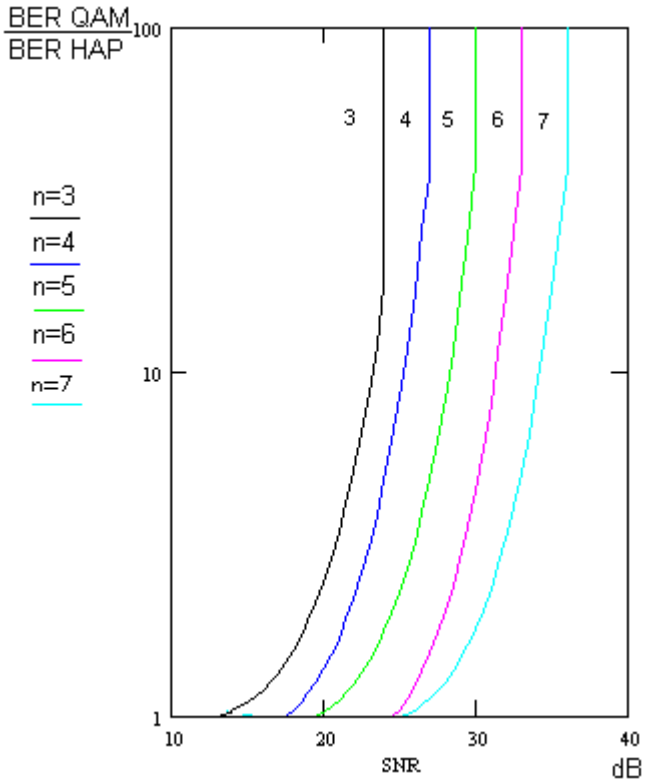


Рис.4.20 Відносний виграш у ймовірності помилки у двійковому розряді

Графіки залежностей середньої за сузір'ям імовірності помилки у двійковому розряді на виході напівнеперервного каналу зображені на рис.4.21.

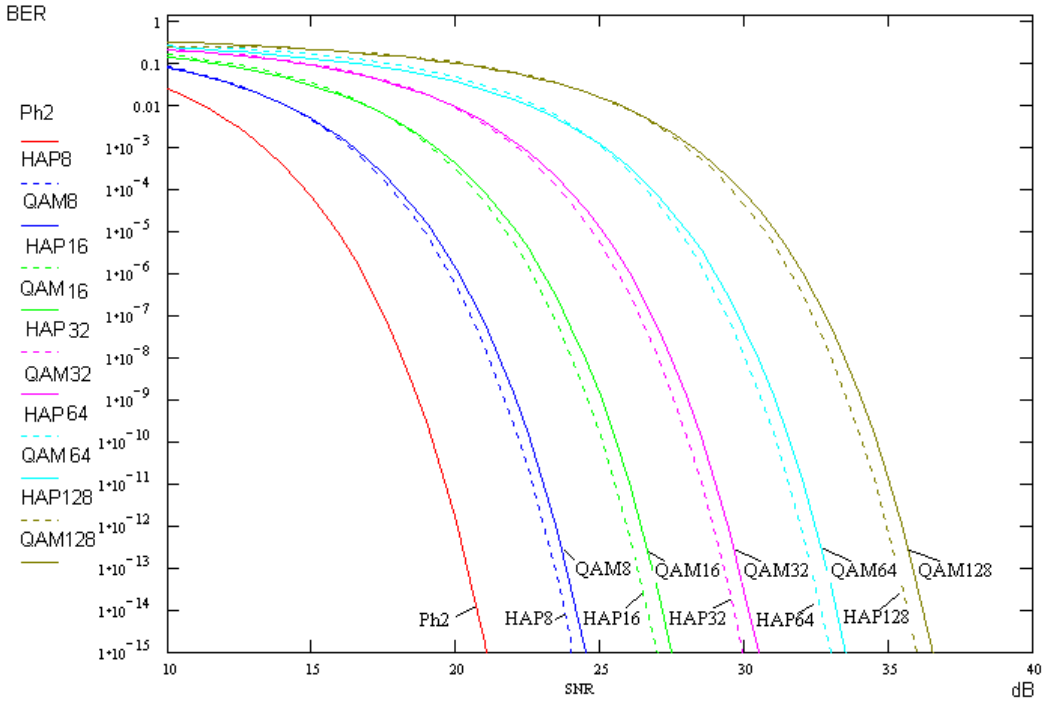


Рис.4.21 Абсолютний виграш у ймовірності помилки у двійковому розряді

4.7 Ефективність каналу передачі управляючої інформації

Відповідно до ієрархії ВВС канал передачі керуючої інформації складається з двох рівнів:

Фізичний рівень каналу забезпечує узгодження параметрів сигналу з такими характеристиками фізичного каналу, як АФЧХ і розподіл щільності потужності шуму, і повинний забезпечувати задану постійну складову затримки сигналу та ймовірність помилки у двійковому розряді переданого цифрового потоку що, у свою чергу, визначає вимоги до середньої швидкості передачі. Побудова підсистеми фізичного рівня є багатокритериальною оптимизаційною задачею, критеріями якої є:

- постійна величина затримки цифрового потоку в каналі ;
- значення ймовірності помилки в двійковому розряді цифрового потоку;

- середня швидкість передачі цифрового потоку;
- вартість підсистеми фізичного рівня.

Канальний рівень забезпечує об'єднання керуючих повідомлень у єдиний потік для передачі на фізичний рівень, контроль імовірності помилки отриманої інформації, контроль затримки повідомлення в каналі залежно від імовірності помилки, забезпечуваною підсистемою фізичного рівня і довжиною конкретного керуючого повідомлення, переданого верхніми рівнями системи управління мережі. Побудова підсистеми канального рівня також є багатокритериальною оптимизаційною задачею, критеріями якої є:

- дисперсія величини затримки керуючого повідомлення в каналі;
- значення ймовірності помилки в керуючому повідомленні;
- середня швидкість передачі потоку повідомлень;
- вартість підсистеми канального рівня.

Випадковий характер величини затримки повідомлень у каналі й імовірності помилки в повідомленні визначається нестаціонарністю характеристик завад у більшості фізичних каналів і, як наслідок, змінною величиною як самої ймовірності помилки в розряді, так і часу обробки повідомлення на канальному рівні. Існують можливості зменшити дисперсію ймовірності помилки в повідомленні та дисперсію затримки повідомлення в каналі шляхом:

- використання виправляючих кодів з великою корегуючою здатністю (наприклад, згорткових), що приводить до введення значної надмірності, що не буде ефективно використовуватися при зниженій імовірності помилки в розряді та значно зменшить пропускну здатність каналу;
- блоковому кодуванні при передачі надвеликими блоками, довжина яких настільки велика, щоб імовірність помилок у блоці дуже мало відрізнялася б від середньостатистичної. Широко відомі блокові коди з оптимальною надмірністю при заданій корегувальній здатності, однак кодування надвеликими блоками приводить до неприйнятного для керуючої інформації

значення постійної величини затримки цифрового потоку в каналі та підвищеній складності декодерів корегувального коду.

Компромісним рішенням у забезпеченні заданої вірності передачі при заданій величині затримки повідомлення в каналі та надмірності корегувального коду стали алгоритми зміни кількості надмірності, що вводиться, при зміні стану каналу. Оптимальної була б така надоперативна адаптація, при якій кількість надмірності, що вводиться, точно відповідала б стану каналу в даний момент часу. Але оскільки надмірність вводиться на передачі, а стан каналу оцінюється на прийомі, необхідний зворотній канал між приймачем і передавачем, за яким інформація про ситуацію в основному каналі ПД передавалася б від приймача до передавача в зворотньому напрямку. Такі канали одержали назву каналів зворотного зв'язку (ЗЗ). Процес передачі через канал ЗЗ оперативної інформації про поточний стан основного каналу дозволяє різко підвищити ефективність системи ПД. Однак, можливості таких систем обмежені наступними факторами: по-перше, передавач одержує з каналу ЗЗ відомості про основний канал, що відносяться до кодової комбінації , що вже була передана й оброблена приймачем, після чого ці відомості пройшли через канал ЗЗ. Оскільки, при використанні завадостійкого кодування вигідно використовувати довгі кодові комбінації, за час передачі в обидва кінці системи ситуація в основному каналі може змінитися. Отже, відомості про основний канал можуть утратити свою цінність, і навіть привести до негативного результату, якщо на їхній основі передавач прийме невірне рішення про зміну параметрів передачі. Безсумнівно, що це не стосується тих даних з каналу ЗЗ, що використовуються для виправлення отриманої приймачем кодової комбінації. З іншого боку, у переважній більшості випадків, основний канал піддається дії того ж джерела завад що і канал ЗЗ. Отже, при передачі з малими затримками важко домогтися високої оперативності адаптації системи через те, що сигнал про погіршення завадової обстановки, переданий каналом ЗЗ, попадає в часовий інтервал дії завади, яка привела до передачі цього самого сигналу. Імовірність правильного прийому сигналу ЗЗ про погіршення умов передачі в цей час різко падає,

внаслідок чого, збільшується час реакції системи аж до того, що сигнал про застосування заходів для підвищення завадостійкості приймається вже після закінчення дії завади.

Не зважаючи на ці обмеження, адаптивні властивості систем зі ЗЗ дають можливість оптимізувати кількість надмірності, внесеної коригувальним кодом, у залежності від імовірності помилок у двійковому розряді і приводять до значного економічного ефекту, що і забезпечило їх широке впровадження. Зокрема, у більшості систем передачі використовується алгоритм, відомий, як вирішальний зворотний зв'язок з поточним адресним перезапитом (ВЗЗап).

Задачі оптимізації кількості надмірності, що вводиться в канал коригувальним кодом при різних значеннях імовірності помилки в розряді приділено досить уваги в працях з теорії завадостійкого кодування, також досить докладно розглянуті питання впливу ЗЗ на затримку повідомлення в каналі. Однак, для передачі керуючої інформації становить інтерес взаємозв'язок дисперсії затримки повідомлення, імовірності помилки і пропускної здатності двійкового каналу зі ЗЗ і нестаціонарністю імовірності помилок.

При розгляді такого каналу задамося найбільш характерними для нього обмеженнями:

- керуюча інформація надходить на вхід каналу у виді повідомлень (блоків), довжиною, кратною 8 біт (у силу специфіки розвитку сучасної елементної бази);
- блоковий характер керуючої інформації визначає застосування блокових кодів (найчастіше кодів Боуза-Чоудхурі-Хоквингема (БЧХ) чи їхнього недвійкового підкласу – кодів Ріда-Соломона (РС));
- фізичний рівень забезпечує ймовірність помилки двійкового символу p. Але помилки групуються в пакети.

У якості загальної математичної моделі розглянутого каналу найбільше підходить модель Л.П.Пуртова. Моделі на основі ланцюгів Маркова вимагають більш точного опису вихідних параметрів і більшого обсягу обчислень, тому

більше підходять для опису конкретних каналів. Модель Л.П.Пуртова описує ймовірність помилки кратності, рівної чи більшої, ніж *t* у блоці довжиною n елементів у залежності від імовірності помилки в елементі *p* і коефіцієнта групування помилок *α*: $P_{>=t}(Nk)=(Nk/t)^{1-a}$ Застосування моделі обмежене значеннями кратності помилки $t<Nk/3$, характерними для більшості каналів. Коефіцієнт групування помилок лежить у межах 0.5...0.7 - для кабельних і 0.2...0.4 - для радіоканалів. Граничні значення: $\alpha=0$ - незалежні помилки і $\alpha=1$-всі помилки збираються в один пакет.

Як приклад розглянемо залежності *p(t)* імовірності помилок кратності *t* -від *t* у кодовій комбінації довжиною *Nk=256* розрядів при ймовірності незалежної помилки $p=10^{-3}$ і коефіцієнтах групування помилок $\alpha=0.3, 0.5$ і 0.7.

На рис.4.22 показані графіки цих залежностей.

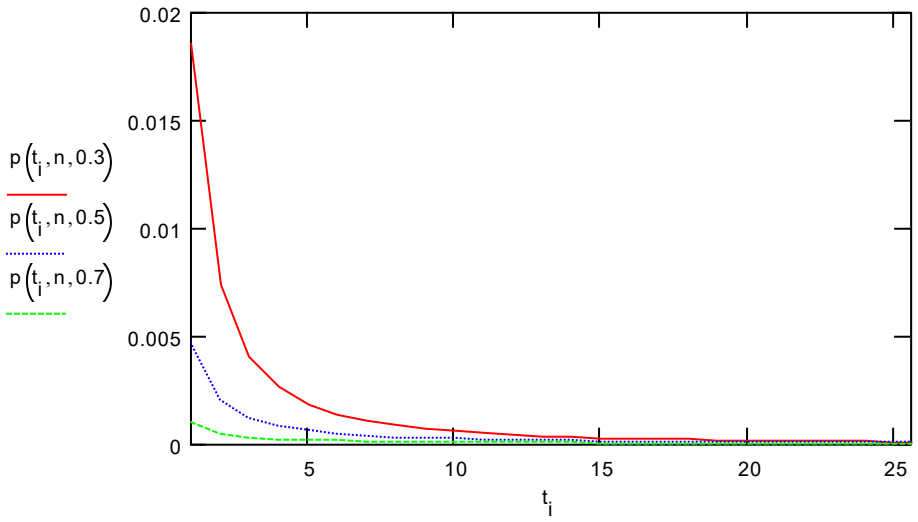


Рис 4.22 Залежності P(t) у каналах із групуванням помилок

З рис.4.22 видно, що, по-перше, при збільшенні коефіцієнта групування помилок за інших рівних умов зменшується ймовірність помилки кратності *t*, причому ця закономірність зберігається у всій області застосування моделі Л.П.Пуртова. В

других, імовірність помилки кратності t завжди більше ймовірності помилки кратності $t+1$. І нарешті, зі збільшенням коефіцієнта групування помилок, різниця між імовірностями помилок кратності t і $t+1$ зменшується.

Нескладний розрахунок показує, що середнє значення кратності помилки не на багато перевищує одиницю при будь-яких реальних значеннях p і α. Тому, у загальному випадку, вигідно використовувати код БЧХ у режимі виправлення однократної помилки.

Однак, у проводових каналах з коефіцієнтами групування помилок, що приймають значення від $\alpha=0.5$ до $\alpha=0.7$ (нижні криві на рис.4.22.), імовірність двократних, трикратних і т.д. помилок мало відрізняється від імовірності однократної помилки. Тобто, можна сказати, що, хоча найбільш імовірні однократні помилки, але необхідно приділити велику увагу помилкам малої кратності, відмінної від одиниці.

На практиці для виправлення двох-триразових помилок використовується перезапит усієї кодової комбінації. Такий підхід, на перший погляд, не позбавлений сенсу: для виправлення помилок кратності, що перевищує одиницю, необхідно вводити в кожну кодову комбінацію занадто велику надмірність і, оскільки трикратні помилки зустрічаються рідше однократних, ця надмірність у більшості кодових комбінацій не буде використана.

Але, з іншого боку, якщо, наприклад, для виправлення однократної помилки в кодовій комбінації довжиною $N_k=1024$ досить шістнадцяти перевірочних розрядів коду БЧХ, то для виправлення двократної помилки в тій же комбінації перезапитуються всі n розрядів, хоча відомо, що для виправлення кодом БЧХ дворазової помилки в такій комбінації досить тридцяти трьох розрядів.

Отже, у використовуваному методі корекції помилок виявляється недолік: для виправлення помилок малої кратності, що перевищує одиницю, застосовується чи відносно невелика кількість перевірочних розрядів коду БЧХ але в кожній кодовій

комбінації (незалежно від наявності помилки) чи перезапит тільки у випадку виявлення такої помилки, але дуже великого числа розрядів (уся кодова комбінація).

Обидва варіанти неефективно використовують надмірність: перший - за рахунок того, що перевірочна частина кодової комбінації використовується цілком (у режимі виправлення помилки) винятково рідко (у будь-якому випадку, ймовірність безпомилкового прийому буде набагато вище ймовірності однократної помилки і, тим більше, ймовірності помилок більшої кратності). Другий варіант не ефективний в наслідок того, що кількість переданих розрядів при перезапиті ніяк не зв'язана із кратністю помилки, а розрахована на виправлення всіх можливих помилок. Більш того, під час перезапиту комбінації, у якій виявлена помилка, яку не можна виправити за допомогою перевірочної частини, може виникнути ситуація, коли в перепитаній комбінації виникає невиявлена помилка, або помилка, що не виправляється, і веде до наступного перезапиту

Очевидним шляхом виправлення цього недоліку було б таке число перевірочних розрядів у кожній комбінації, що забезпечувало б виправлення найбільш імовірної помилки (однократної), а для виправлення помилок більшої кратності використання такої модифікації перезапиту, при якій кількість переданих розрядів була б оптимізована з урахуванням імовірностей помилок різної кратності.

Спробою такого удосконалення і є пропонований метод скороченого адресного перезапиту. Суть його полягає в наступному:

Так само, як і в існуючих системах, після інформаційної частини кодової комбінації передається перевірочна частина – результат кодування наприклад, кодом БЧХ. Призначення цієї частини те ж саме – виправлення помилок малої кратності і виявлення помилок непарної кратності, невеликих пакетів помилок і помилок кратності, що не перевищує подвоєну кратність помилки, що виправляється. При декодуванні на прийомі може виникнути одна з трьох ситуацій:

- прийом був безпомилковий чи кратність помилки перевищила виявляючі здібності коду. У такому випадку інформаційна частина кодової комбінації зберігається в накопичувачі прийому для наступної видачі одержувачу;
- у прийнятій комбінації виявлена помилка малої кратності, яку можна виправити за допомогою перевірочної частини цієї ж комбінації. - Помилка виправляється й інформаційна частина зберігається в накопичувачі прийому;
- у прийнятій комбінації виявляється помилка, що не може бути гарантовано виправлена за допомогою перевірочної частини. У таких випадках дії системи, що використовує скорочений перезапит, відрізняються від дій звичайної системи.

У першу чергу, відкидається перевірочна частина прийнятої комбінації – вона уже виконала свою задачу, а відкинувши її, система позбавляється і від помилок, що вона могла містити.

Потім система відправляє через канал ЗЗ сигнал перезапиту, а інформаційну частину зберігає в накопичувачі прийому.

Одержавши сигнал перезапиту, що містить ідентифікатор ушкодженої комбінації система, що передає, формує і відправляє перевірочну частину коду з більшою виправляючою здатністю (виправляючу послідовність), для чого використовує копію інформаційної частини ушкодженої комбінації, що зберігається в накопичувачі передачі.

Отримавши виправляючу послідовність, приймаюча система декодує код, яким вона була захищена і використовує її точно так само, як і перевірочну частину будь-якої кодової комбінації, тобто намагається виправити помилку в інформаційній частині ушкодженої кодової комбінації за допомогою виправляючої послідовності. Якщо кількість розрядів виправляючої послідовності недостатня для виправлення помилок в інформаційній частині, то формується повторний перезапит, і передається виправляюча послідовність з більшою виправляючою здатністю.

Розглянемо переваги і недоліки такого методу. На рис.4.23 зображено чотири графіки, що ілюструють можливість застосування алгоритму скороченого адресного перезапиту:

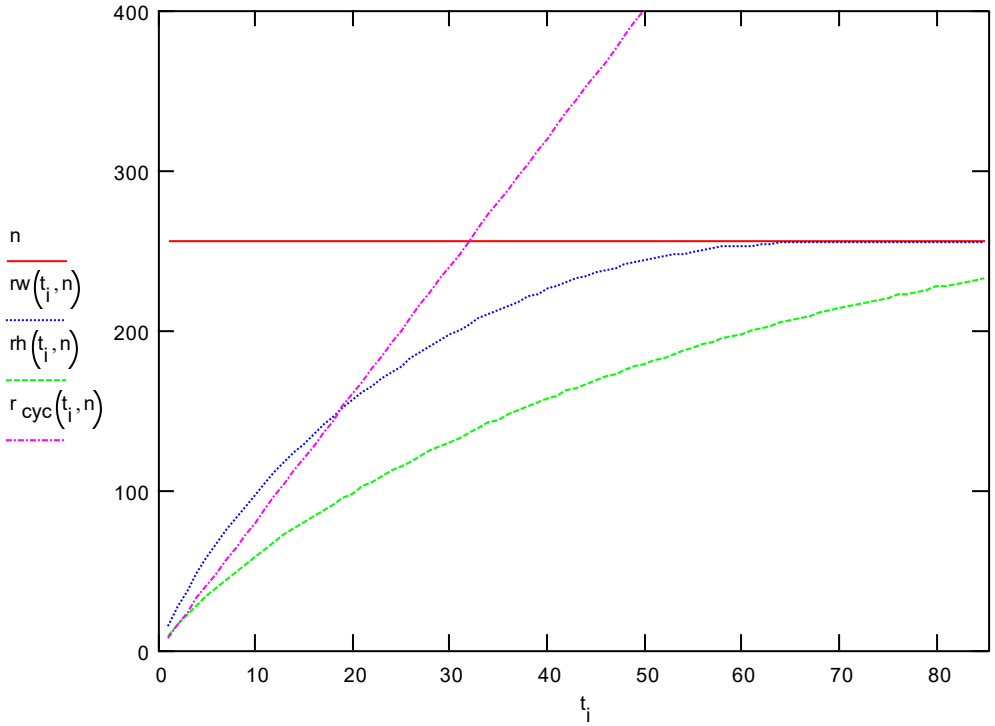


Рис.4.23 Мінімально необхідна кількість розрядів виправляючої послідовності

На осі ординат відкладена кількість розрядів виправляючої послідовності, яка є необхідною для гарантованого виправлення будь-якої помилки кратності *t* (*t* відкладена на осі абсцис) в кодовій комбінації довжиною N_k (у даному прикладі $N_k=256$). Горизонтальна пряма *n(t)=n=const* характеризує перезапит усієї кодової комбінації (далі - повний перезапит) у який незалежно від кратності помилки для виправлення використовується вся кодова комбінація довжиною *n*. Нижня крива (межа Хемінга – *rh(t)*) показує, що для виправлення помилки кратності *t*, необхідно не менш *rh* перевірочних розрядів (але межа Хемінга не гарантує існування кодів з

такими виправляючими властивостями). Друга крива, асимптотично наближається до «*n*» (межа Варшамова - *rw(t)*), показує, що існують коди, що містять не більш *rw* перевірочних розрядів, і здатні виправити будь-яку помилку кратності *t*. Тобто, те, що крива межі Варшамова розташована нижче $N_k=N_k(t)=const$, показує, що у всій області істинності моделі часткового опису дискретного каналу (для всіх $t<N_k/3$) існують коди, за допомогою яких можна виправити будь-яку помилку кратності *t*, перевірочна частина яких містить меншу кількість розрядів, чим використовується при повному перезапиті. А крива межі Хемінга показує, що не позбавлений сенсу пошук таких кодів, застосування яких при скороченому перезапиті було б вигідно і при помилках дуже великої кратності, що виходить за межі області істинності моделі Л.П.Пуртова.

Іншими словами, в області істинності моделі Л.П.Пуртова застосування скороченого перезапиту з виправляючою послідовністю, що має характеристики, не гірші ніж вимагаються за умовами Варшамова (а такі послідовності існують) дає безсумнівний виграш у кількості службових даних, переданих при перезапиті.

Остання крива на рис.4.23, що ще не була розглянута, характеризує широко поширені коди БЧХ. Як видно з графіка, застосування скороченого перезапиту з виправляючою послідовністю, що представляє собою перевірочну частину коду БЧХ, виправдане і дає найбільший виграш при помилках малої кратності. Але існують умови (при дуже великих *t*), у яких застосування повного перезапиту вигідніше застосування скороченого перезапиту з використанням коду БЧХ. Однак, зважаючи на те, що кратність помилки заздалегідь невідома й імовірність помилок великої кратності досить мала (див. рис.4.22) для оцінки можливості застосування перевірочної частини кодів БЧХ як виправляючою послідовністю у системі зі скороченим перезапитом необхідний додатковий розрахунок.

Приклад розрахунку результатів застосування скороченого перезапиту для коду *CRC-16*, імовірності помилки у двійковому розряді $p=10^{-3}$, коефіцієнту групування помилок $\alpha=0.5$, довжини кодової комбінації $N_k=1024$ і обмеженням числа перезапитів *L* до п’яти.

Функція розподілу ймовірностей помилок кратності t при вищезгаданих умовах :

$$P(t, N_k, p, \alpha) := p \cdot N_k^{1-\alpha} \cdot \left[\frac{1}{t^{1-\alpha}} - \frac{1}{(t+1)^{1-\alpha}} \right].$$

Імовірність помилки, виявленої кодом БЧХ $P_{вп}$, описується в [28] як сума ймовірностей помилок кратності t, коли:

- t-парне;
- t-непарне, але менше подвоєної максимальної кратності помилки, що може бути гарантовано виправлена;
- t-непарне, але менше половини ступеня утворюючого поліному.

Імовірність невиявленої помилки $P_{нп}$ є різницею помилки кратності, більшої за нуль та ймовірності виявленої помилки:

$$P_{НП}(t, N_k, p, \alpha) := p \cdot N_k^{1-\alpha} - P_{ВП}.$$

Тоді ймовірність h перезапитів при перезапиті всієї комбінації (режим повного перезапиту):

$$P_{ПП}(h) := P_{ПП}(h-1) \cdot P_{ВП} \left[1 - p \left(\frac{N_k}{t} \right)^{1-\alpha} \right], \qquad h = 1,2,...$$

А в режимі скороченого перезапиту:

$$P_{СП}(h) := P_{СП}(h-1) \left[p \left(\frac{N_k}{h} \right)^{1-\alpha} \right], \qquad h = 1,2,...$$

Для вищезгаданих умов ці ймовірності приймають значення, приведені в таблиці 4.11:

Таблиця 4.11

Імовірності *h* перезапитів при виправленні однієї кодової комбінації

число перезапитівh	Імовірність при повному перезапиті	Імовірність при скороченому перезапиті
1	0.02264	0.02264
2	$5.12631\ 10^{-04}$	$5.12315\ 10^{-04}$
3	$1.16066\ 10^{-05}$	$9.46513\ 10^{-06}$
4	$2.6279\ 10^{-07}$	$1.51442\ 10^{-07}$
5	$5.94992\ 10^{-09}$	$216726\ 10^{-09}$

З урахуванням перезапитів імовірність непоміченої помилки для режиму повного перезапиту описується, як проста сума ймовірностей непомічених помилок в першій комбінації та в комбінаціях перезапиту, зважених наведеними в таблиці 4.2 ймовірностями. Ймовірність непоміченої помилки для режиму скороченого перезапиту майже повністю визначається ймовірністю непоміченої помилки в кодовій комбінації, яка передавалась вперше. Отже абсолютний виграш в імовірності непоміченої помилки складає:

$$BP_{НП} := \sum_{h=1}^{L} P_{НП}(t, N_k, p, \alpha) \cdot P_{ПП}(h).$$

Виграш у пропускної здатності є різницею кількості інформації, що використовується для виправлення помилок, помноженою на швидкість модуляції *B*:

$$BC := B\left(\sum_{h=1}^{L} P_{ПП}(h) \cdot N_k - \sum_{h=1}^{L} P_{ПС}(h) \cdot r(h+1, N_k) \right),$$

де $r(t,N)$-довжина виправляючої послідовності, яка необхідна для виправлення помилки кратності t в кодовій комбінації N. Для коду БЧХ за [28]:

$$r(t,N)=int(1+t\cdot log2(N+1)) .$$

Для вказаних вище умов виграш в імовірності непоміченої помилки складає 0.2, а виграш у пропускної спроможності, нормований швидкістю модуляції -23.

Висновки до розділу 4

1. В розділі запропонована методика розробки автоматизованих систем експериментальних досліджень. Експериментальне дослідження комутатора визначило параметри біфуркації трафіку при яких забезпечується гарантовану достовірність передаваємої інформації.

2. В розділі проведено автоматизована система експериментальних досліджень, експериментальне дослідження комутатора, що забезпечує гарантовану достовірність передаваємої інформації в інфокомунікаційних мережах України.

3. Проведені в роботі експериментальні дослідження комутатора дозволили визначити параметри мережі SDN при різній кількості його абонентів.

ВИСНОВКИ

В рамках дисертаційного дослідження проведено аналіз та обґрунтовано актуальність важливого наукового завдання щодо розробки моделі, методів, алгоритмів та методик отримання вхідних та вихідних параметрів програмно-конфігуруємих мереж з гарантованою достовірністю управляючої інформації.

Отримано наступні результати, що відповідають меті та задачам дослідження:

1. Вперше розроблено модель конвергентної мережі, наукова новизна якої полягає в тому, що вона ґрунтується на теорії інформації та методах імітаційного моделювання, що дозволяє враховувати особливості роботи пристрою та оцінити основні характеристики трафіку, що передається через нього.

2. Вперше розроблено метод автоматизованого експериментального дослідження конвергентного програмно-конфігурованого обладнання, наукова новизна якого полягає в тому, що він ґрунтується на теорії систем та методах системного аналазі та дозволяє отримати імовірнісні та навантажувальні характеристики обладнання на довільних біфоркаціях трафіку та наборах конфігурації.

3. Вперше розроблено методику автоматизованої побудови гібридного програмно-конфігурованого обладнання, наукова новизна якої полягає в тому, що вона ґрунтується на методах об’єктно орієнтовного підходу та дозволяє автоматичному режимі виконувати побудову гібридного програмно-конфігурованого обладнання.

4. Вперше розроблено алгоритм оптимального прийому багатопозиційних сигналів з фазорізницевою модуляцією, наукова новизна якого полягає в тому, що він ґрунтується на максимально правдоподібній оцінці сигналу та дозволяє отримати максимальну достовірність передаваємої інформації при заданому співвідношенні сигнал/завада.

5. Результати дослідження можуть бути використані провайдерами телекомунікаційних послуг, телекомунікаційними компаніями, науковими та проектними організаціями. У ході подальших досліджень планується розробка методів, що дозволяють підвищити точність зняття характеристик обладнання за рахунок використання попередньо оптимізованих експериментально шаблонів для конкретних виробників.

6. Достовірність результатів забезпечується коректністю використання математичних положень термінів та прийомів щодо доведень і обґрунтувань тверджень та підтверджується результатами наведених експериментальних досліджень. Таким чином, наукове завдання в дисертації виконано, мета досягнута.

Список використаних джерел

1. Softw are-Defined Networking: The New Norm for Networks [Електронний ресурс] / ONF. – 2012. // – Режим доступу: https://www.opennetworking.org/images/stories/downloads/white-papers/wp-sdn-newnorm.pdf (05.04.2013 р.).

2. Banjar, A. Analysing the performance of the OpenFlow standard for software-defined networking using the OMNeT++ network simulator/ A. Banjar, P. Pupatwibul, R. Braun, B. Moulton //2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE). - IEEE, 2014. - С. 31-37.

3. Беркман Л. Н. Архітектурна концепція побудови, принцип реалізації,ефективність застосування інтелектуальної телекомунікаційної мережі /Л. Н. Беркман, С. В. Толюпа // Зб. наук. праць ВІТІ НТУУ —КПІ . – 2007. – No3. – С. 9-17.

4. Колченко В. О. Впровадження інтелекту в мережі наступного покоління (NGN) – перехід до мереж майбутнього покоління (FGN) /В. О. Колченко / Наукові записки УНДІЗ. – 2010. – No2(14). – С.80-85.

5. Беркман Л. Н. Проблеми створення сучасної конвергентної мережі на базі концепції FMC (Fixed-Mobile Convergence) / Л. Н. Беркман, О. І. Чумак, В. В. Григорович, П. Ю. Дещинський // Вісник УНДІЗ. – 2008. – No2. – С. 61-63.

6. Павлов В. В. Начала теории эргатических систем / В. В. Павлов. – Київ : Наук. думка, 1975. – 240с.

7. OpenFlow Switch Specification [Електронний ресурс] / Open network foundation.2013.URL // – Режим доступу: http://archive.openflow.org/documents//openflow-spec-v1.1.0.pdf (10.04.2014 р.).

8. OpenFlow Switch Specification Version 1.2 [Електронний ресурс] /ONF. – 2012. // – Режим доступу: https://www.opennetworking.org/images/stories//downloads/specification/openflow-spec-vl .2.pdf. (15.08.2014 р.).

9. Nadeau T. SDN: Software Defined Networks / T. Nadeau, K. Gray //O'Reilly. – 2013. – No3. – P. 70-95.

10. POX Wiki/POX.2014.URL [Електронний ресурс] // – Режим доступу: https://openflow.stanford.edu/display/ONL/POX+Wiki (10.04.2014 р.).

11. OpenFlow Switch Specification [Електронний ресурс] / Open network foundation.2013.URL. // – Режим доступу: http://archive.openflow.org/documents/ /openflow-spec-v1.1.0.pdf (10.04.2014 р.).

12. Shi X. et al. PABO: Mitigating congestion via packet bounce in data center networks //Computer Communications. - 2019. - T. 140. - C. 1-14.

13. Software-defined networking: the new norm for Networks [Електронний ресурс] / ONF White Paper. – April 13, 2012. // – Режим доступу: https://www.opennetworking.org/publications/whitepaper/13/04/2012/sdn-new_norm.pdf. (22.03.2014 р.).

14. OpenFlow Tutorial [Електронний ресурс] / OpenFlow.2013.UPL. // – Режим доступу: //http://archive.openflow.org/wk/index.php/OpenFlowTutorial (07.11.2014 р.).

15. ONF Specification [Електронний ресурс] / Open network foundation.2013.URL. // – Режим доступу: https://www.opennetworking.org/sdn-resources/onf-specifications (07.11.2013 р.).

16. Software-Defined Networking (SDN) Definition [Електронний ресурс] / ONF. – 2013. // – Режим доступу: https://www.open-networking.org/sdn-resources/sdn-definition (07.03.2014 р.).

17. Draft Formal Specification: Framework for Software-De-fined Networks (SDN) [Електронний ресурс] / IETF. – February 2013. // – Режим доступу: https://tools.ietf.org/html/draft-shin-sdn-formal-specification-03 (07.03.2014 р.).

18. Framework of Telecom SDN (Software-Defined Networking) // ITU-T Draft Recommendation Y.FNsdn. – February 2013.

19. Egawa T. SDN standardization Landscape from ITU-T Study Group 13 //ITU Workshop on SDN Geneva, Switzerland. – June 2013.

20. McKeown N. OpenFlow Enabling Innovation in Campus Networks /N. McKeown, T. Anderson // SIGCOMM. – 2008. – Vol. 38. – P. 69-74.

21. Gharbi A., Gharsellaoui H. and Ben Ahmed S. Multi-Agent control system. 2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA). 2014. pp. 117-124.

22. Toulis, Panos; Airoldi, Edoardo (2015). "Scalable estimation strategies based on stochastic approximations: classical results and new insights". Statistics and Computing. 25 (4): 781–795. doi:10.1007/s11222-015- 9560-y. PMC 4484776. PMID 26139959.

23. MiniNet documentation [Електронний ресурс] / MiniNet.2014.URL // – Режим доступу: https://github.com/mininet/wiki/Documentation (10.04.2014 р.).

24. Vesely, V. Multicast simulation and modeling in OMNeT++ / V. Vesely, P. Matousek, M. Sveda //SimuTools. - 2013. - С. 142-145.

25. Ushakova, M. Research of Productivity of Software Configurable Infrastructure in Vanet Networks on the Basis of Models of Hybrid Data Transmission Devices / M. Ushakova, Y. Ushakov, I. Bolodurina, D. Parfenov, L. Legashev, A. Shukhman // 2020 International Scientific and Technical Conference Modern Computer Network Technologies (MoNeTeC). - IEEE, 2020. - С. 1-9.

26. RedDropper: INET Framework 4.2 documentation [Електронний ресурс]. - Режим доступа: https://doc.OMNeTpp.org/inet/api-current/neddoc/inet.queueing.filter.RedDropper.html

27. Openflow for INET 4 [Електронний ресурс]. -2020. - Режим доступа: https://github.com/CrezZ/OpenFlow-OMNET-INET4

28. R.Horning. OpenFlow Module for OMNeT++ 5.4+ and INET 3.6 [Електронний ресурс]. -2017. - Режим доступа: https://github.com/inet-framework/openflow

29. Поповський В.В., Олійник В.Ф. Математичні основи управління і адаптації в телекомунікаційних системах: підручник. – Х .: ТОВ “Компанія СМІТ”, 2011. – 362 с

30. Ushakova, M.V. Model of the Distributed Self-Organizing Network of IoT Sensors / M. V. Ushakova, Y. A. Ushakov, P. N. Polezhaev, V. N. Tarasov // (MoNeTeC) 2018 : International Scientific and Technical Conference Modern Computer Network Technologies, 25-26 Oct. 2018, Moscow. - P. 1-6.

31. Телекомунікаційні системи та мережі. Структура й основні функції. Том 1. / Поповський В.В., Лемешко О.В., Ковальчук В.К., Плотніков 119 М.Д., Картушин Ю.П. та інші. [Електронний ресурс]. – Х.: ХНУРЕ. 2018. Режим доступу – http://www.znanius.com/3534.html.

32. Телекомунікаційні системи та мережі. Структура й основні функції. Том 1. / Поповський В.В., Лемешко О.В., Ковальчук В.К., Плотніков М.Д., Картушин Ю.П. та інші. [Електронний ресурс]. – Х.: ХНУРЕ. 2018. Режим доступу – http://www.znanius.com/3534.html.

33. Ushakov, Y. A. Service-oriented routing in distributed self-organizing software-defined networks / Y. A. Ushakov, M. V. Ushakova, A. E. Shukhman, P. N. Polezhaev, L. V. Legashev //2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT). - IEEE, 2018. - С. 1-5.

34. Boudali H., Dugan J.B. A continuous-time Bayesian network reliability modeling, and analysis framework. - IEEE Transactions on Reliability, vol. 55, No. 1, March 2006, pp. 86-97.

35. Barakabitze A. A. et al. 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges //Computer Networks. - 2020. - Т. 167. - С. 106984.

36. Veins: The open source vehicular network simulation framework [Електронний ресурс]. - 2020. - Режим доступа: https://veins.car2x.org/

37. Lei Y., Tai K. C. In-parameter-order: A test generation strategy for pairwise testing / Y. Lei, K. C. Tai //Proceedings Third IEEE International High-Assurance Systems Engineering Symposium (Cat. No. 98EX231). - IEEE, 1998. - С. 254-261.

38. Seroussi G. Vector sets for exhaustive testing of logic circuits / G. Seroussi, N. H. Bshouty //IEEE Transactions on Information Theory. - 1988. - Т. 34. - №. 3. - С. 513-522.

39. Nayyar, A. A Comprehensive Review of Simulation Tools for Wireless Sensor Networks (WSNs) / A. Nayyar, R. Singh // Journal of Wireless Networking and Communications . - 2015. - Т. 5. - №. 1. - С. 19-47.

40. IP Switching Cisco Express Forwarding Configuration Guide, Cisco IOS XE Fuji 16.7.x [Електронний ресурс] // Cisco Systems. - 2018. - Режим доступа: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipswitch_cef/configuration/xe-16-8/isw-cef-xe-16-8-book/isw-cef-overview.html.

41. (Manual: Packet Flow [Електронний ресурс] // MicroTic Documentation. - 2019. - Режим доступа: https://wiki.mikrotik.com/wiki/Manual:Packet_Flow

42. NE40E V800R010C10SPC500 Hardware Description [Електронний ресурс] // Huawei Technologies. - 2020. - Режим доступа: https://support.huawei.com/enterprise/en/doc/EDOC1100055058/ef36bd8/product-signal-flow-overview.

43. Богуш К.Ю., Богуш Ю.П., Шиян А.И. GPSS World Моделювання телекомунікаційних систем та мереж. Посібник для дипломного проектування . – ІСЗЗІ НТУУ КПІ, 2010

44. Teixeira, R. PacketScope: Monitoring the Packet Lifecycle Inside a Switch / R. Teixeira, R. Harrison, A. Gupta, J. Rexford //Proceedings of the Symposium on SDN Research. - 2020. - С. 76-82.

45. В.В. Клобуков, В.А. Рябоконь, С.О. Ермак, «Дослідження протоколів побудови маршрутів в мережах METRO ETHERNET». Матеріали X Міжнародної науково-технічної конференції «АВІА-2011», Том 1, С. 5.72- 5.76, 2011.

46. Kardas G. Model-driven development of multiagent systems: A survey and evaluation. The Knowledge Engineering Review. 2013. Vol. 28. P. 479–503.

47. Skobelev, P, 2011, ‘Multi-Agent Systems for Real Time Resource Allocation, Scheduling, Optimization and Controlling: Industrial Applications’ in fifth International Conference on Industrial Applications of Holonic and Multi Agent Systems, Toulouse, France, 2011, Springer, Berlin, vol. 6867, pp. 1-14.

48. Hegr T. et al. Measurement of switching latency in high data rate ethernet networks //Elektronika ir Elektrotechnika. - 2015. - Т. 21. - №. 3. - С. 73-78.

49. Oleh Kitura, Turovsky Oleksandr, Vadym Vlasenko, Nataliia Rudenko, Oleksandr Golubenko, Oleksandr Drobyk. Two-Time Procedure for Calculation of Carrier Frequency of Phasomodulated in Communication System, International Journal of Communication Networks and Information Security (IJCNIS),2021, Vol. 13, No. 3, pp. 382-387. DOI :10.54039/ijcnis.v13i3.5165.

50. Іванець О. Б., Морозова І. В. Назарчук М. А., Миколушко А. М. Іваницький Є. С., Синтез інформаційної системи діагностування складних об’єктів. Українській метрологічний журналу. 2020. №1 А. С. 50–51. DOI : 10.24027/2306-7039.1A.2020.193279

51. Ushakova, M. Research of Performance Parameters of Virtual Switches with OpenFlow Support / Margarita Ushakova, Yuriy Ushakov, Jie Cui, Leonid Legashev, Alexandr Shukhman, Aleksandr Bolodurin // VII Международная конференция «Инжиниринг & Телекоммуникации — En&T-2020».

52. Tarasov V. et al. Analysis of intervals between traffic packets on the Software Defined Networks depending on the TCP window size //2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T). - IEEE, 2016. - С. 15-17.

53. Beneta E. V., Sakharova M. A. & Kanayev A. K. Integrated mathematical operation process model of Carrier Ethernet intelligent control system. The 20th International conference on soft computing and measurements (SCM–2017). Book of reports. In 3 vol. Saint Petersburg, SPbGETU “LETI” Publ., 2017, vol. 2, pp. 282–285.

54. Wooldridge, Michael. (2004). An Introduction to Multi-Agent Systems. J. Artificial Societies and Social Simulation. 7. 10.1007/978-3-642-01904-3_2.

55. Kravari K., Bassiliades N. A survey of agent platforms. Journal of Artificial Societies and Social Simulation. 2015. Vol. 18. P. 11.

56. Wooldridge, Michael. (2004). An Introduction to Multi-Agent Systems. J. Artificial Societies and Social Simulation. 7. 10.1007/978-3-642-01904-3_2.

57. Кітура О. В., Беркман Л. Н., Дакова Л. В., Даков С. Ю., Блаженний Н. В., Полонський К. В. Дослідження когнітивних методів керування інфокомунікаціями

в умовах невизначеності. Звя'зок. №4(152), С. 7-13, 2021. DOI: 10.31673/2412-9070.2021.040713

58. Bolodurina, I. Method of Obtaining Generalized Statical Characteristics of Network Equipment for Creation of Simulation Models / Bolodurina I., Ushakova M., Ushakov Y. // 2020 International Russian Automation Conference (RusAutoCon). - IEEE, 2020. - С. 947-952.

59. Кітура О. В., Колодяженський Б. М., Тушич А. М., Інтеграція IoT у сферу логістики. Звя'зок. №2(150), С. 28-29, 2021. DOI: 10.31673/2412-1159070.2021.022829.

60. Ushakova, M. Creation of adequate simulation models to analyze performance parameters of a virtual fog computing infrastructure / Margarita Ushakova, Yuriy Ushakov, Irina Bolodurina, Denis Parfenov, Leonid Legashev, Alexandr Shukhman // 14th International Symposium “Intelligent Systems - 2020”.

61. System and Technologies of Digital Television: manual for graduate students. [Text]/ V.A. Loshakov, V.V . Popovsky , S.O. Saburova, I.S. Shostko, M.Y. Oshepkov, K.O. Popovskaya, L.I. Melnikova. Under the general editorship of Professor V.A. Loshakov – Kh: Company SMIT”, 2019. – 416 p.

62. Кітура О. В., Яковенко Н. Д., Сторчак К. П., Звенігородський О. С., Бєлкін Ю. О., Математичне моделювання динамічних зв’язаних процесів на основі скінченно-елементної методики. Звя'зок. №3(151), С. 35-38, 2021. DOI: 10.31673/2412-9070.2021.033538

63. М. Луцький, О. Іванець, В. Шавшина. Інформаційна система оцінювання ймовірності виникнення ризикової події при прийнятті біомедичних рішень. Наукоємні технології, №1 (53), 2022. С. 41 – 48.

64. Mekky, H. Network function virtualization enablement within SDN data plane / H. Mekky, F. Hao, S. Mukherjee, T. V. Lakshman, Z. Zhang // IEEE INFOCOM 2017-IEEE Conference on Computer Communications. - IEEE, 2017. - С. 1-9.

65. Odarchenko R. S. Modeling of SDN Overlay Networks and Their Main Characteristics Research / R. S. Odarchenko, S. Yu. Dakov, V. V. Polischuk, A. M. Tyrsenko // Knowledge-based technologies. – 2016. – № 3 (31). – P. 284.

66. Skobelev P., Paulo Leitão, Stamatis Karnouskos (Ed.). Multi- Agent Systems for Real Time Adaptive Resource Management. In Industrial Agents: Emerging Applications of Software Agents in Industry. Elsevier, 2015. P. 207– 230.

67. Ndiaye M. Software Defined Networking for Improved / M. Ndiaye, G.P. Hancke, A.M. Abu-Mahfouz // Wireless Sensor Network Managemen, A Survey. Sensors. – 2017. – 17. – P. 1031

68. Herrnleben, S. Model-Based Performance Predictions for SDN-Based Networks: A Case Study / S. Herrnleben, P. Rygielski, J. Grohmann, S. Eismann, T. Hobfeld, S. Kounev // International Conference on Measurement, Modelling and Evaluation of Computing Systems. - Springer, Cham, 2020. - С. 82-98.

69. Коноваленко О. Є., Брусенцев В. О. Мультиагентні системи управління та підтримки прийняття рішень. Вісник Національного технічного університету "ХПІ". Сер.: Машинознавство та САПР. 2019. № 1. С. 18-27.

70. Solomentsev, O., Zaliskyi, M., Shcherbyna, O., Kozhokhina, O. Sequential Procedure of Changepoint Analysis during Operational Data Processing. Proceedings of 2020 IEEE Workshop on Microwave Theory and Techniques in Wireless Communications, MTTW 2020this link is disabled, 2020, Pp. 168–171, 9245068.)

71. Дмитренко В.В. Імітаційно-аналітичні моделі гібридних програмно-конфігурованих пристроїв./ Дмитренко В. В., Марчук О. М. // Зв’язок. , №6(166), 2023, С. 37-43.

72. Дмитренко В.В. Проблеми синтезу моделей мережного обладнання та методи їх вирішення/ Дмитренко В.В., Брезіцький С.М. // Телекомунікаційні та інформаційні технології, № 4 (81) , 2023, С. 95-100.

73. Software-defined networking: the new norm for Networks [Електронний ресурс] / ONF White Paper. – April 13, 2012. // – Режим доступу: https://www.opennetworking.org/publications/whitepaper/13/04/2012/sdn-new_norm.pdf. (22.03.2014 р.).

74. Software-Defined Networking (SDN) Definition [Електронний ресурс] / ONF. – 2013. // – Режим доступу: https://www.open-networking.org/sdn-resources/sdn-definition (07.03.2014 р.).

75. Zhang J. SDN replaced deployment and real-time QoS provisioning based on network models // International Journal of Autonomous and Adaptive Communications Systems. - 2020. - Т. 13. - №. 1. - С. 11-22.

76. Yang Y. Y. et al. Implementation of network traffic monitor system with SDN / Y. Y. Yang, C. T. Yang, S. T. Chen, W. H. Cheng, F. C. Jiang, //2015 IEEE 39th annual computer software and applications conference. - IEEE, 2015. - Т. 3. - С. 631¬634.

77. Wong, M. D. Testing Compilers for Programmable Switches Through Switch Hardware Simulation / M. D. Wong, A. Varma, A. Sivaraman //arXiv preprint arXiv:2005.02310. - 2020.

78. Wang, S. Y. EstiNet openflow network simulator and emulator / S. Y. Wang, C. L. Chou, C. M. Yang //IEEE Communications Magazine. - 2013. - Т. 51. - №. 9. - С. 110-117.

79. Wang, G. Programming Your Network at Run-time for Big Data Applications / G. Wang, T.S. Eugene Ng, A. Shaikh // Hot Topics in Software Defined Networking (HotSDN). — Helsinki, Finland, 2012.

80. Wang S. Y. Comparison of SDN OpenFlow network simulator and emulators: EstiNet vs. Mininet //2014 IEEE Symposium on Computers and Communications (ISCC). - IEEE, 2014. - С. 1-6.

81. NetSim Network Simulator [Электронный ресурс] / Boson Holdings. - 2020. - Режим доступа: https://www.boson.com/netsim-cisco-network-simulator.

82. NetSim User Manual [Электронный ресурс]. -2019. -Режим доступа: https://www.tetcos.com/downloads/v11/NetSim_User_Manual.pdf

83. Salih, M. A. OpenFlow 1.3 extension for OMNeT++ / M. A. Salih, J. Cosmas, Y. Zhang //2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing. - IEEE, 2015. - С. 1632-1637.

84. Sato T. et al. Abstract model of sdn architectures enabling comprehensive performance comparisons //2015 11th International Conference on Network and Service Management (CNSM). - IEEE, 2015. - С. 99-107.

85. Shi X. et al. PABO: Mitigating congestion via packet bounce in data center networks //Computer Communications. - 2019. - T. 140. - C. 1-14.

86. Tang Y. et al. Automatic belief network modeling via policy inference for SDN fault localization //Journal of Internet Services and Applications. - 2016. - T. 7. - №. 1. - C. 1.

87. Tarasov V. et al. Analysis of intervals between traffic packets on the Software Defined Networks depending on the TCP window size //2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T). - IEEE, 2016. - C. 15-17.

88. Teixeira, R. PacketScope: Monitoring the Packet Lifecycle Inside a Switch / R. Teixeira, R. Harrison, A. Gupta, J. Rexford //Proceedings of the Symposium on SDN Research. - 2020. - C. 76-82.

89. Ushakov, Y. A. Service-oriented routing in distributed self-organizing software-defined networks / Y. A. Ushakov, M. V. Ushakova, A. E. Shukhman, P. N. Polezhaev, L. V. Legashev //2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT). - IEEE, 2018. - C. 1-5.

90. Kumar, D. Analysis of Impact of Network Topologies on Network Performance in SDN / D. Kumar, M. Sood //International Conference on Innovative Computing and Communications. - Springer, Singapore, 2020. - C. 357-369.

91. Kriege, J.,\ Simulating stochastic processes with OMNeT++ / J. Kriege, P. Buchholz //SimuTools. - 2011. - C. 367-374.

92. Kabir M. H. et al. Detail comparison of network simulators //International Journal of Scientific & Engineering Research. - 2014. - T. 5. - №. 10. - C. 203-218.

93. Ivey J. et al. Comparing a scalable SDN simulation framework built on ns-3 and DCE with existing SDN simulators and emulators //Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation. - 2016. - C. 153-164.

94. Bolodurina, I. Method of Obtaining Generalized Statical Characteristics of Network Equipment for Creation of Simulation Models / Bolodurina I., Ushakova M.,

Ushakov Y. // 2020 International Russian Automation Conference (RusAutoCon). - IEEE, 2020. - C. 947-952.

95. Banjar, A. Analysing the performance of the OpenFlow standard for software-defined networking using the OMNeT++ network simulator/ A. Banjar, P. Pupatwibul, R. Braun, B. Moulton //2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE). - IEEE, 2014. - C. 31-37.

96. Steinbach T. et al. An extension of the OMNeT++ INET framework for simulating real-time ethernet with high accuracy / T. Steinbach, H.D. Kenfack, F. Korf, T.C. Schmidt //Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques. - 2011. - C. 375-382.

97. System and Technologies of Digital Television: manual for graduate students. [Text]/ V.A. Loshakov, V.V . Popovsky , S.O. Saburova, I.S. Shostko, M.Y. Oshepkov, K.O. Popovskaya, L.I. Melnikova. Under the general editorship of Professor V.A. Loshakov – Kh: Company SMIT", 2019. – 416 p.

98. Oleh Kitura, Turovsky Oleksandr, Vadym Vlasenko, Nataliia Rudenko, Oleksandr Golubenko, Oleksandr Drobyk. Two-Time Procedure for Calculation of Carrier Frequency of Phasomodulated in Communication System, International Journal of Communication Networks and Information Security (IJCNIS),2021, Vol. 13, No. 3, pp. 382-387. DOI :10.54039/ijcnis.v13i3.5165.

99. Shirzadeh Chaleshtari, A, Shadrokh, Sh, 2012, 'A Branch and Bound Algorithm for Resource Constrained Project Scheduling Problem subject to Cumulative Resources', World Academy of Science, Engineering and Technology, vol. 6, pp. 23-28.

100. Beneta E. V. & Kanayev A. K. The formation of failure control algorithms in telecommunication network, built on the basis of Carrier Ethernet technology. Informatsionniye tekhnologii na transporte. Sbornyk materialov sektsii 118 "Informatsionniye tekhnologii na transporte" Mezhdunarodnoy konferentsii "Regionalnaya informatyka–2016" [Information technologies on transport. Coll. papers of "Information technologies on transport" section of International conference "Zone-based informatics–2016"]. Saint Petersburg, PGUPS Publ., 2016, pp. 95–101.

101. Xu Xiaofei Carrier SDN: Next-gen carrier Networking [Електронний ресурс] / Xiaofei Xu // Режим доступу: https://www.huawei.com/en/about-huawei/publications/communicate/hw-259729.htm (27.06.2015 р.).

102. Movahedi, Zahra & Bastanfard, Azam. (2021). Toward competitive multi-agents in Polo game based on reinforcement learning. Multimedia Tools and Applications. 80. 10.1007/s11042-021-10968-z.

103. Романов, О. І. ., Марінов, А. І. ., & Сколець, С. С. . (2021). Дослідження мереж sdn з використанням платформи mininet. Збірник матеріалів Міжнародної науково-технічної конференції «ПЕРСПЕКТИВИ ТЕЛЕКОМУНІКАЦІЙ», 98–101. вилучено із http://conferenc.its.kpi.ua/proc/article/view/230356.

104. Fei Chen and Wei Ren On the Control of Multi-Agent Systems: A Survey. Foundations and Trends® in Systems and Control. 2019. Vol. 6. No. 4, pp 339-499. http://dx.doi.org/10.1561/2600000019.