Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: " Модель надання мережевих сервісів у SDN з підтримкою Bring Your Own Control "

Виконав: студент VI курсу, групи СБм-61
Спеціальності:
125 Кібербезпека та захист інформації
(шифр і назва напряму підготовки, спеціальності)
Чепіль Ігор Михайлович
підпис (прізвище та ініціали)

Керівник Карпінський М. П.
підпис (прізвище та ініціали)

Нормоконтроль Стадник М. А.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент
підпис (прізвище та ініціали)

м. Тернопіль – 2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Чепілю Ігорю Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Модель надання мережевих сервісів у SDN з підтримкою Bring Your Own Control

Керівник роботи Карпінський Микола Петрович,
доктор технічних наук, професор кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 24 » 11 2025 року № 4/7-1024

2. Термін подання студентом завершеної роботи 12.12.2025

3. Вихідні дані до роботи Архітектура SDN, параметри функціонування сервісів VPN/IPS, дані про оброблення мережевого трафіку та події безпеки.

4. Зміст роботи (перелік питань, які потрібно розробити)
Аналіз архітектури програмно-визначених мереж, формування моделі розподілу управлінських повноважень за концепцією Bring Your Own Control, проєктування механізмів взаємодії між контролерами оператора й користувача, оцінювання ефективності запропонованої моделі з огляду на час реагування, адаптивність та рівень кіберзахисту.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Актуальність дослідження. Мета, об'єкт, предмет дослідження.
Наукова новизна та практичне значення.
Завдання дослідження.
Архітектура SDN. Концепція Bring Your Own Control.
Модель життєвого циклу сервісу з підтримкою BYOC. Місце BYOC в архітектурі SDN.
Модель BYOC для IPS у середовищі SDN. Реалізація BYOC для VPN з інтегрованим IPS.
Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент		
Безпека в надзвичайних ситуаціях	Теслюк В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 19.09.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	20.09 – 22.09	Виконано
2.	Підбір джерел для аналізу в галузі дослідження	25.09 – 10.10	Виконано
3.	Опрацювання джерел в галузі дослідження	11.10 – 15.10	Виконано
4.	Налаштування симуляційного середовища	16.10 – 25.10	Виконано
5.	Оформлення розділу «Теоретичні засади побудови sdn-інфраструктур та управління мережевими сервісами»	25.10 – 05.11	Виконано
6.	Оформлення розділу «Модель надання сервісів у SDN з підтримкою BYOC»	06.11 – 10.11	Виконано
7.	Оформлення розділу «Архітектура та реалізація BYOC-сервісу IPS у середовищі SDN»	11.11 – 25.11	Виконано
8.	Виконання завдання до підрозділу «Охорона праці та безпека в надзвичайних ситуаціях»	26.11-01.12	Виконано
9.	Оформлення кваліфікаційної роботи	02.12 – 10.12	Виконано
10.	Нормоконтроль	08.12 – 10.12	Виконано
11.	Перевірка на плагіат	12.12 – 14.12	Виконано
12.	Попередній захист кваліфікаційної роботи	15.12 – 20.12	Виконано
13.	Захист кваліфікаційної роботи	23.12.2025	

Студент _____ Чепіль І. М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Карпінський М. П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Модель надання мережевих сервісів у SDN з підтримкою Bring Your Own Control // ОР «Магістр» // Чепіль Ігор Михайлович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2025 // С. 75, рис. – 13, табл. – 1 , кресл. – 12, додат. – 2.

Ключові слова: BYOC, SDN, IPS, Guest Controller, XMPP, OpenDaylight, Cybersecurity.

У кваліфікаційній роботі магістра розроблено архітектуру сервісу мережевої безпеки за моделлю BYOC у середовищі SDN з інтегрованою IPS. Запропонована архітектура поєднує централізоване управління оператором сервісу з можливістю клієнта контролювати політику безпеки через власний Guest Controller із модулем Security Manager. У моделі кінцеві модулі IDS-end аналізують трафік на сайтах, Service Operator керує інфраструктурою через OpenFlow на південному інтерфейсі (SBI), а клієнт отримує події та надсилає рішення через XMPP на північному інтерфейсі (NBI). Можливість практичної реалізації продемонстровано на прикладі VPN з інтегрованою IPS. Показано, що інтеграція з відкритим SDN-контролером OpenDaylight дає змогу реалізувати XMPP як розширення північного інтерфейсу, що робить архітектуру сумісною зі стандартами й придатною для промислового використання.

Результати дослідження підтверджують доцільність використання моделі BYOC у сервісах кібербезпеки, демонструють можливість практичного впровадження та окреслюють напрями подальшого розвитку, зокрема автоматизацію політик, підтримку розподіленого IPS та масштабування на основі OpenDaylight.

ABSTRACT

A model of network service delivery in SDN with Bring Your Own Control support // Thesis of educational level "Master"// Ihor Chepil // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, group СБм-61 // Ternopil, 2025 // p. 75, figs. 13, tbls. - 1, drws. 12, apps. 2.

Keywords: BYOC, SDN, IPS, Guest Controller, XMPP, OpenDaylight, Cybersecurity.

In the master's thesis, the architecture of a network security service based on the BYOC model in an SDN environment with an integrated IPS was developed. The proposed architecture combines centralized management by the service operator with the ability of the client to control the security policy through their own Guest Controller with a Security Manager module. In the model, IDS-end modules analyze traffic at the sites, the Service Operator manages the infrastructure via OpenFlow on the southbound interface (SBI), and the client receives events and sends decisions via XMPP on the northbound interface (NBI). The possibility of practical implementation is demonstrated on the example of a VPN with integrated IPS. It is shown that integration with the open SDN controller OpenDaylight enables the implementation of XMPP as an extension of the northbound interface, making the architecture standards-compliant and suitable for industrial use.

The research results confirm the feasibility of using the BYOC model in cybersecurity services, demonstrate the potential for practical deployment, and outline directions for further development, including policy automation, support for distributed IPS, and scaling based on OpenDaylight.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SDN	—	Software-Defined Networking
BYOC	—	Bring Your Own Control
SBI	—	Southbound Interface
NBI	—	Northbound Interface
XMPP	—	Extensible Messaging and Presence Protocol
VPN	—	Virtual Private Network
NFV	—	Network Functions Virtualization
IPS	—	Intrusion Prevention System
SLA	—	Service Level Agreement
REST	—	Representational State Transfer
API	—	Application Programming Interface
SO	—	Service Orchestrator/Operator
GC	—	Guest Controller
DE	—	Decision Engine
SD	—	Service Dispatcher
NaaS	—	Network as a Service
IaaS	—	Infrastructure as a Service
SOAP	—	Simple Object Access Protocol
SRM	—	Service Request Manager
IDS	—	Intrusion Detection System
SM	—	Security Manager
XML	—	Extensible Markup Language

ВСТУП

Актуальність теми. У умовах зростання складності мережевих інфраструктур та інтенсивного впровадження цифрових сервісів, питання кібербезпеки стають критично важливими. Програмно-визначені мережі (SDN) пропонують гнучке та централізоване управління трафіком, що значно спрощує автоматизацію мережевих процесів. Водночас централізація створює потенційні точки відмови та вразливості, які можуть бути використані зловмисниками. Для підвищення захищеності та адаптивності мережевого середовища все більшого поширення набуває концепція BYOC. Реалізація моделей сервісного управління з підтримкою BYOC відкриває нові можливості для динамічної оркестрації, підвищення безпеки та персоналізації мережевих функцій, що особливо актуально в контексті мереж 5G, NFV та хмарних платформ.

Мета і задачі дослідження. Метою кваліфікаційної роботи є удосконалення моделі надання мережевих сервісів у SDN-мережі з підтримкою концепції BYOC, орієнтованої на підвищення рівня кібербезпеки.

Для досягнення цієї мети необхідно вирішити такі задачі:

- проаналізувати архітектурні підходи до побудови SDN-інфраструктур та моделей управління сервісами;
- визначити обмеження централізованого підходу до управління сервісами безпеки в SDN;
- дослідити можливості інтеграції концепції BYOC у процес життєвого циклу мережевих сервісів;
- розробити спрощену модель надання сервісів з розподіленим управлінням (оператор – клієнт);
- побудувати взаємодію між SDN-контролером і зовнішніми BYOC-застосунками через Northbound API;
- реалізувати експериментальну модель надання сервісу та перевірити її працездатність;
- оцінити переваги та обмеження запропонованого підходу.

Об’єкт дослідження. Процеси надання та управління мережевими сервісами в інфраструктурі програмно-визначених мереж.

Предмет дослідження. Архітектура, інтерфейси та життєвий цикл сервісів у SDN з реалізацією моделі BYOC.

Наукова новизна одержаних результатів кваліфікаційної роботи. У кваліфікаційній роботі удосконалено модель надання мережевих сервісів у SDN шляхом інтеграції концепції BYOC для децентралізованого управління життєвим циклом сервісів та гнучкого управління сервісами безпеки з боку користувача.

Практичне значення одержаних результатів. Розроблена модель може бути використана:

- при побудові прототипів SDN-систем з підтримкою BYOC у хмарних середовищах;
- для покращення безпеки та адаптивності сервісів, зокрема IPS або VPN, що розгортаються динамічно;
- в навчальному процесі для демонстрації реалізації децентралізованого керування в програмно-визначених мережах.

Апробація результатів магістерської роботи. Основні результати дослідження були представлені на XIV Міжнародній науково-практичній конференції молодих учених та студентів «Актуальні задачі сучасних технологій» (ТНТУ, Тернопіль, Україна, 11-12 грудня 2025 р).

Публікації. Основні результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А).

РОЗДІЛ 1 ТЕОРЕТИЧНІ ЗАСАДИ ПОБУДОВИ SDN-ІНФРАСТРУКТУР ТА УПРАВЛІННЯ МЕРЕЖЕВИМИ СЕРВІСАМИ

1.1 Загальна характеристика програмно-визначені мережі

1.1.1 Принципи SDN

Програмно-визначені мережі ґрунтуються на концепції відокремлення площини управління від площини передачі даних [1]. У традиційних мережах рішення про маршрутизацію та обробку трафіку приймаються безпосередньо кожним мережевим пристроєм, що ускладнює адміністрування та призводить до обмеженої гнучкості при впровадженні нових сервісів. На відміну від цього, у SDN усі функції управління зосереджуються у спеціальному контролері, який формує глобальне бачення стану мережі й визначає правила її роботи, тоді як мережеві комутатори та маршрутизатори лише виконують ці інструкції. Такий підхід забезпечує централізоване управління і дозволяє швидко змінювати політики маршрутизації, балансування навантаження чи безпеки без потреби вручну конфігурувати кожен пристрій.

Сутність принципів SDN полягає у створенні програмно конфігурованої мережевої інфраструктури, де логіка управління реалізується на рівні програмного забезпечення, а сама інфраструктура розглядається як абстрактний ресурс. Це відкриває можливість для автоматизації управління, оркестрації сервісів, масштабування ресурсів і впровадження інновацій без зміни фізичних компонентів. Ще однією важливою рисою є використання відкритих і стандартизованих інтерфейсів, що дає можливість сумісність обладнання та програмних платформ від різних виробників. Таким чином, SDN формує нову парадигму побудови мереж, у якій головним стає не апаратне забезпечення, а програмна логіка, що забезпечує гнучкість, адаптивність і підвищений рівень контролю, зокрема у сфері кібербезпеки.

1.1.2 Архітектура SDN

Архітектура SDN базується на концепції багаторівневого поділу функцій, що забезпечує чітке розмежування між передачею даних, прийняттям рішень та управлінням інфраструктурою. У класичних мережах усі ці завдання поєднані в одному пристрої, що призводить до надмірної складності та обмежує можливості масштабування. SDN пропонує інший підхід, у якому функції розподілені між трьома площинами. Архітектура програмно-визначених мереж побудована за принципом багаторівневої взаємодії між різними площинами, кожна з яких виконує свою специфічну функцію (див. рисунок 1.1) [2].

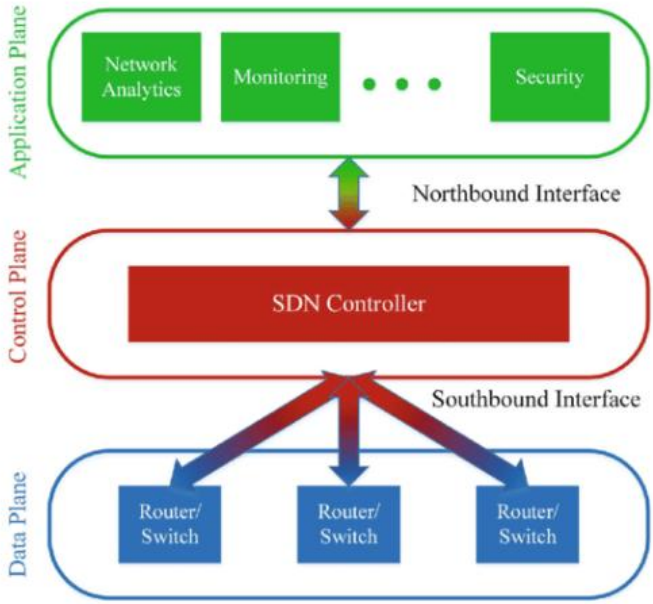

Рисунок 1.1 – Архітектура SDN

У нижній частині знаходиться площина даних, яка складається з фізичних мережевих пристроїв - маршрутизаторів і комутаторів. Їхнє завдання зводиться до пересилання пакетів відповідно до правил, що надходять від контролера. Вони не приймають власних рішень, а діють як виконавчий рівень у загальній архітектурі.

Над площиною даних розташована площина управління, де ключовим елементом виступає SDN-контролер. Саме він формує логіку роботи всієї мережі, забезпечує централізоване бачення топології та стану трафіку, приймає рішення щодо маршрутизації, застосування політик безпеки й управління

ресурсами. Контролер виступає центральною «точкою інтелекту» системи, через яку проходить уся інформація про стан мережі, а далі у вигляді правил надсилається на рівень пристроїв. Взаємодія між контролером і обладнанням здійснюється через південний інтерфейс, що забезпечує передачу інструкцій від рівня управління до рівня виконання.

Над контролером розміщена площина застосунків, яка відповідає за реалізацію бізнес-логіки, оркестрацію сервісів та забезпечення кібербезпеки. Тут функціонують модулі аналітики, моніторингу та безпеки, що використовують глобальну інформацію, яку надає контролер, і формують політики відповідно до потреб користувача чи оператора. Зв’язок між прикладною площиною та контролером відбувається через північний інтерфейс, що дозволяє прикладним сервісам задавати правила функціонування мережі у зручному програмному вигляді.

У структурі програмно-визначених мереж центральним елементом виступає контролер, який виконує роль «мозку» всієї системи. Він відповідає за централізоване управління трафіком, формування політик маршрутизації та безпеки, а також за координацію роботи різних пристроїв у площині даних. На відміну від класичних мереж, де кожен маршрутизатор чи комутатор самостійно приймає рішення, у SDN ці функції сконцентровані в контролері, що дозволяє досягти глобального бачення стану мережі, швидше реагувати на загрози та ефективніше використовувати ресурси. Контролер не лише зберігає інформацію про топологію, а й динамічно оновлює правила пересилання пакетів відповідно до поточних умов чи вимог користувача.

Взаємодія контролера з фізичними пристроями здійснюється за допомогою південного інтерфейсу, відомого як Southbound Interface. Цей інтерфейс є каналом зв’язку, через який контролер передає інструкції маршрутизаторам і комутаторам. Найбільш поширеним стандартом, що використовується для реалізації SBI, є протокол OpenFlow, який дозволяє створювати та змінювати таблиці потоків у пристроях нижнього рівня

Використання такого підходу забезпечує єдину логіку управління та виключає потребу у складних локальних конфігураціях. Таким чином, SBI є

механізмом трансляції рішень контролера в конкретні дії на рівні мережевого обладнання.

З іншого боку, контролер взаємодіє з прикладною площиною через північний інтерфейс, або Northbound Interface. Це набір програмних інтерфейсів, які дозволяють додаткам, таким як системи моніторингу, аналітики чи безпеки, отримувати доступ до інформації про стан мережі та формувати власні правила її функціонування. Найчастіше NBI реалізується у вигляді RESTful API, що робить можливим інтеграцію з будь-якими зовнішніми сервісами. Завдяки NBI прикладні рішення можуть впливати на поведінку мережі, налаштовувати сервіси, забезпечувати контроль доступу або реагувати на інциденти безпеки.

У результаті взаємодія між контролером, південним та північним інтерфейсами створює цілісну екосистему управління, де фізичні пристрої виконують роль виконавців, контролер виступає координатором, а прикладні сервіси формують логіку та політики, зокрема у сфері кіберзахисту. Така архітектура дозволяє реалізувати більш гнучкі й безпечні моделі надання мережевих сервісів, серед яких важливе місце займає концепція Bring Your Own Control.

1.2 Технології підтримки гнучкого керування в SDN

1.2.1 Протокол OpenFlow

Протокол OpenFlow є технологією, яка стала основою розвитку програмно-визначених мереж [3]. Його поява дозволила відокремити функції управління від функцій пересилання даних і створити стандартизований канал взаємодії між контролером та мережевими пристроями. У традиційних мережах кожен маршрутизатор або комутатор самостійно приймає рішення про обробку пакетів, що ускладнює централізоване управління. OpenFlow змінює цей підхід: він дає змогу пристроям нижнього рівня працювати як простим виконавцям, а логіка управління зосереджується в контролері (див. рисунок 1.2).

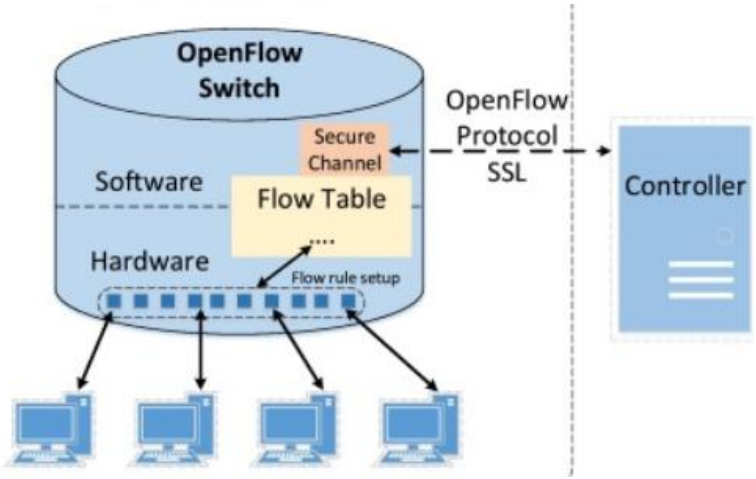

Рисунок 1.2 – Логіка роботи OpenFlow

Робота OpenFlow базується на концепції таблиць потоків, у яких зберігаються правила для обробки мережевого трафіку. Кожен потік визначається набором параметрів заголовка пакета, таких як MAC-адреси, IP-адреси, номери портів чи протоколи. Контролер формує ці правила та надсилає їх до комутатора за допомогою OpenFlow. Пристрій, отримавши правило, застосовує його до всіх пакетів, що відповідають умовам, без потреби повторного звернення до контролера. Це дозволяє одночасно досягти високої продуктивності пересилання та централізованого контролю.

Однією з важливих особливостей OpenFlow є можливість динамічного оновлення таблиць потоків у реальному часі. Коли в мережі виникають нові вимоги, наприклад зміна політик доступу, необхідність фільтрації трафіку чи перенаправлення потоків у разі атаки, контролер миттєво оновлює правила і передає їх до пристроїв. Така гнучкість робить OpenFlow ефективним інструментом для кібербезпеки, оскільки дозволяє швидко реагувати на інциденти, блокувати шкідливий трафік або перенаправляти підозрілі пакети на спеціальні вузли для аналізу. Крім того, OpenFlow став стандартом, який активно підтримується більшістю дослідницьких і комерційних платформ SDN. Його відкрита специфікація сприяє сумісності обладнання від різних виробників і забезпечує широкі можливості інтеграції з іншими технологіями. Завдяки цьому OpenFlow не лише створює основу для централізованого управління мережею, а й формує універсальне середовище для впровадження нових сервісів, оркестрації мережевих функцій і побудови захищених інфраструктур нового покоління.

1.2.2 Платформи контролерів

Платформи контролерів у програмно-визначених мережах є центральною ланкою, яка визначає функціональність усієї інфраструктури. Вони відповідають за управління потоками, реалізацію політик безпеки, моніторинг та координацію взаємодії між рівнями архітектури. Кожна платформа має свої особливості, проте всі вони орієнтовані на забезпечення гнучкості, масштабованості та сумісності мережевих рішень.

OpenDaylight є однією з наймасштабніших і найпопулярніших платформ контролерів із відкритим кодом [4]. Вона створена під егідою Linux Foundation і підтримує широкий набір протоколів, зокрема OpenFlow, NETCONF та BGP-LS. Архітектура OpenDaylight модульна, що дає можливість розробникам створювати власні додатки для управління трафіком, реалізації політик безпеки чи інтеграції з системами оркестрації на кшталт OpenStack. Завдяки своїй гнучкості OpenDaylight застосовується як у дослідницьких, так і в операторських мережах, де потрібна централізована координація та висока надійність.

Ryu є більш простим у використанні контролером, написаним мовою Python [5]. Його головна перевага полягає в доступності для швидкого прототипування та експериментів [6,7]. Дослідники та інженери часто використовують Ryu для перевірки нових алгоритмів маршрутизації, тестування моделей безпеки чи створення навчальних лабораторій. Простота інтеграції та добре задокументований API роблять його зручним інструментом для освітніх цілей і невеликих мережевих рішень.

ONOS (Open Network Operating System) орієнтований на масштабні операторські мережі та відрізняється розподіленою архітектурою, яка забезпечує високу відмовостійкість і продуктивність [8]. ONOS здатний обслуговувати великі дата-центри або телекомунікаційні середовища, підтримуючи при цьому механізми віртуалізації та сервісного ланцюжка. Його архітектура дозволяє горизонтальне масштабування, що робить платформу особливо цінною для телеком-операторів і провайдерів хмарних сервісів.

Крім зазначених рішень, існують й інші контролери, такі як Floodlight чи NOX. Floodlight розроблений на Java та підтримує OpenFlow, вирізняється простотою розгортання та використовується для невеликих SDN-мереж або як основа для навчання. NOX був одним із перших SDN-контролерів і став базою для розвитку багатьох сучасних рішень.

Таким чином, екосистема SDN-контролерів є різноманітною і дозволяє підібрати платформу залежно від потреб: OpenDaylight забезпечує масштабованість і розширюваність, Ryu надає простоту і швидкість прототипування, ONOS орієнтований на великі розподілені середовища, а Floodlight і NOX можуть слугувати у дослідницьких або навчальних цілях. Спільним для всіх контролерів є те, що вони формують основу для централізованого управління, створюють можливість програмного налаштування мереж і дозволяють реалізувати сучасні механізми кіберзахисту в SDN-інфраструктурах.

1.2.3 Інтерфейси та API

Інтерфейси та API у програмно-визначених мережах виконують важливу роль, оскільки саме через них здійснюється взаємодія між контролером і прикладними сервісами, а також інтеграція SDN з іншими системами управління. Northbound Interface забезпечує доступ прикладних програм до функцій контролера. Завдяки цьому розробники можуть створювати сервіси моніторингу, аналітики, безпеки чи оркестрації без необхідності занурюватися у низькорівневу роботу мережевого обладнання. Найбільш поширеним стандартом для реалізації північного інтерфейсу є REST API. Він ґрунтується на принципах HTTP-запитів і дозволяє додаткам уніфіковано звертатися до контролера, створюючи, змінюючи чи видаляючи об’єкти управління мережею. Використання REST робить взаємодію простою та зручною, оскільки вона сумісна з більшістю мов програмування і легко інтегрується у системи автоматизації. REST підходить для сценаріїв, коли потрібно забезпечити масштабованість і прозорість, наприклад для управління правилами

маршрутизації чи політиками доступу. У тих випадках, коли потрібна безперервна асинхронна комунікація, застосовується XMPP. Цей протокол спочатку був розроблений для обміну повідомленнями у реальному часі, однак завдяки своїй гнучкості знайшов застосування і в SDN. Його використання дозволяє організувати постійний двосторонній канал між контролером і додатками, що особливо важливо у середовищах, де швидкість реагування має критичне значення, наприклад у системах виявлення атак чи мобільних мережах. Ще однією сучасною технологією є gRPC - високопродуктивний фреймворк для віддалених викликів процедур, який базується на HTTP/2. На відміну від REST, що використовує текстові повідомлення у форматі JSON чи XML, gRPC передає дані у бінарному вигляді з використанням протоколу Protobuf. Це значно зменшує затримки і підвищує ефективність обміну даними, що робить його придатним для масштабних SDN-рішень у дата-центрах чи операторських мережах. Завдяки gRPC з’являється можливість швидко обробляти великий потік викликів і забезпечувати інтеграцію з мікросервісною архітектурою.

Крім цих технологій, у сфері SDN використовуються й інші інтерфейси, зокрема NETCONF або GraphQL, які дозволяють організувати більш складне управління конфігурацією та взаємодію із зовнішніми системами. NETCONF орієнтований на детальне керування параметрами пристроїв і часто застосовується у поєднанні з YANG-моделями даних. GraphQL, своєю чергою, надає більшу гнучкість у формуванні запитів до контролера, дозволяючи прикладним системам отримувати рівно той набір даних, який їм необхідний.

Усі ці інтерфейси й API разом формують екосистему, яка забезпечує відкритість, гнучкість і сумісність SDN із зовнішніми сервісами. Саме завдяки їм програмно-визначені мережі можуть динамічно змінювати свою поведінку, інтегрувати сервіси безпеки, підтримувати оркестрацію та реагувати на нові загрози в режимі реального часу.

1.3 Оркестрація мережевих сервісів

Оркестрація мережевих сервісів у SDN-інфраструктурі є ключовим механізмом, що забезпечує інтеграцію різних функцій у єдиний керований процес. Вона полягає у координації налаштування, розгортання та моніторингу сервісів у відповідності до вимог користувачів та операторів. На відміну від традиційних мереж, де управління кожним сервісом здійснюється вручну, у програмно-визначених мережах оркестрація відбувається програмно, що забезпечує автоматизацію та швидке реагування на зміни середовища.

Життєвий цикл мережевого сервісу охоплює всі етапи його існування: від створення і налаштування до експлуатації та завершення роботи. На початковому етапі визначаються вимоги до сервісу, описується його політика та функціональні параметри. Далі відбувається автоматизоване розгортання через SDN-контролер, який конфігурує пристрої мережі відповідно до заданих правил. Під час експлуатації сервіс постійно моніториться, а у разі змінних умов або загроз безпеці може бути динамічно перебудований. На завершальному етапі сервіс виводиться з експлуатації, а ресурси повертаються у спільний пул для подальшого використання. Такий підхід дозволяє реалізувати повну автоматизацію та оптимізацію управління життєвим циклом із урахуванням якості обслуговування та вимог до безпеки.

Мережеві сервіси безпеки займають особливе місце в архітектурі SDN, оскільки вони захищають дані від перехоплення, несанкціонованого доступу та атак. У випадку VPN оркестрація забезпечує автоматичне створення захищених тунелів між вузлами, вибір протоколів шифрування та динамічне оновлення ключів без необхідності ручного втручання адміністратора. Інтеграція IPS [9-11] у SDN дозволяє виявляти і блокувати атаки в режимі реального часу, застосовуючи політики безпеки, сформовані контролером. У поєднанні з централізованим управлінням оркестрація таких сервісів дозволяє швидко реагувати на загрози, змінювати конфігурації та масштабувати захист відповідно до потреб мережі. Це особливо важливо у сучасних середовищах, де атаки стають дедалі складнішими, а трафік - дедалі динамічнішим.

Оркестрація сервісів у SDN може здійснюватися у двох основних формах: централізованій та децентралізованій. Централізована модель базується на єдиному контролері, який визначає політики для всієї мережі. Такий підхід забезпечує глобальне бачення, уніфіковані правила та простоту управління, проте він створює ризик єдиної точки відмови та обмежує гнучкість у великих і розподілених мережах. Децентралізована оркестрація передбачає залучення клієнтських або локальних елементів управління, які взаємодіють із центральним контролером, але можуть автономно ухвалювати рішення в межах своїх доменів. Це дозволяє підвищити надійність і зменшити час реакції на локальні події, зокрема загрози безпеці. У сучасних дослідженнях все більшої уваги набувають гібридні моделі, які поєднують централізовані механізми координації з децентралізованими можливостями користувача, що відповідає концепції Bring Your Own Control і відкриває нові перспективи для захищених мереж.

1.4 Концепція Bring Your Own Control

Концепція BYOC з’явилася як відповідь на обмеження традиційної централізованої моделі управління у програмно-визначених мережах. Її сутність полягає в тому, що частина функцій управління передається безпосередньо клієнту або кінцевому користувачу. Якщо у класичному SDN усі рішення приймаються контролером, то в архітектурі з підтримкою BYOC користувач має можливість самостійно впливати на життєвий цикл сервісу: ініціювати його створення, змінювати параметри безпеки, контролювати доступ і навіть формувати власні політики. Це дозволяє поєднати централізоване бачення оператора з гнучкістю локального управління. Технічно BYOC реалізується через NBI контролера, який відкриває API для сторонніх застосунків, або за допомогою проміжних рішень на кшталт гостьових контролерів, що взаємодіють із центральною системою.

Впровадження BYOC у SDN відкриває нові можливості у сфері кіберзахисту. Завдяки децентралізації користувач отримує можливість

оперативно реагувати на загрози у своїй мережі без очікування централізованих змін від оператора. Наприклад, клієнт може самостійно створити правило блокування небезпечного з’єднання, підключити додатковий IPS-модуль або змінити політику шифрування VPN-тунелю. Це суттєво скорочує час від виявлення інциденту до його нейтралізації. Крім того, BYOC підвищує гнучкість у реалізації персоналізованих заходів безпеки: різні користувачі чи організації можуть впроваджувати власні стратегії захисту, адаптовані до їхніх специфічних потреб. Таким чином, BYOC знижує ризик єдиної точки відмови, властивий централізованим системам, і створює багаторівневу модель безпеки, де оператор відповідає за загальну цілісність інфраструктури, а клієнт - за індивідуальний захист.

Ідея BYOC активно досліджується у науковій спільноті та поступово знаходить своє відображення у практичних реалізаціях. Серед перших рішень можна відзначити інтеграцію BYOC у середовища з підтримкою OpenDaylight, де було розроблено механізми доступу клієнтів до північних інтерфейсів для самостійного управління сервісами VPN. Подібні підходи використовувались і в ONOS, де концепція BYOC дозволяла клієнтам взаємодіяти з контролером через REST API для створення власних політик безпеки. Наукові дослідження, проведені у сфері оркестрації NFV, демонструють можливість розширення BYOC на віртуалізовані середовища, де користувачі можуть управляти окремими VNF, наприклад системами IPS чи брандмауерами.

У сучасних публікаціях також відзначається перспективність BYOC для 5G-мереж та концепції network slicing. Тут BYOC дозволяє користувачам налаштовувати параметри виділеного сегмента мережі відповідно до своїх вимог, не порушуючи глобальної політики оператора. Це відкриває новий рівень персоналізації сервісів і робить мережі більш гнучкими й орієнтованими на кінцевого споживача. Таким чином, BYOC уже сьогодні розглядається не лише як теоретична концепція, а й як практичний інструмент підвищення безпеки та адаптивності мережевих інфраструктур.

1.1 Висновки до розділу 1

В першому розділі було узагальнено теоретичні засади побудови програмно-визначених мереж і підходи до управління мережевими сервісами. Показано, що ключовою ідеєю SDN є розділення площини управління та площини передачі даних, що переводить прийняття мережевих рішень із рівня окремих пристроїв на рівень централізованого контролера. Такий підхід забезпечує програмну конфігурованість, глобальне бачення топології та стану мережі, прискорює впровадження інновацій і створює передумови для підвищення кіберстійкості.

Розкрито архітектуру SDN як багаторівневу модель, у якій площина даних виконує пересилання трафіку, площина управління (контролер) формує політики й маршрути, а площина керування/менеджменту забезпечує моніторинг і інтеграцію з зовнішніми системами. Окремо акцентовано роль SBI для трансляції рішень контролера до мережевих пристроїв і NBI для взаємодії контролера з прикладними сервісами. У цьому контексті обґрунтовано значення відкритих інтерфейсів і стандартів для взаємодії рішень різних виробників. Показано, що протокол OpenFlow є базовим механізмом програмування площини даних, оскільки дозволяє керувати таблицями потоків у режимі реального часу та оперативно застосовувати політики доступу, фільтрації й перенаправлення трафіку. Окреслено роль API й інтерфейсів у побудові відкритої екосистеми SDN, що забезпечує масштабовану інтеграцію сервісів моніторингу, аналітики та безпеки.

Сформовано цілісне уявлення про оркестрацію мережевих сервісів як про процес керування їхнім життєвим циклом - від проєктування й автоматизованого розгортання до експлуатації та виведення з роботи. Показано специфіку оркестрації сервісів безпеки (VPN, IPS), де вимоги до швидкості реакції, динамічної конфігурації та персоналізації особливо високі.

Окрему увагу приділено концепції BYOC як логічному еволюційному кроку в межах SDN, де частина функцій управління передається клієнтові без порушення цілісності політик оператора. Обґрунтовано, що BYOC посилює

кібербезпеку завдяки можливості оперативного, контекстно-залежного втручання користувача у роботу сервісів (наприклад, у VPN або IPS-сценаріях), підтримує персоналізацію засобів захисту та органічно поєднується з 5G-мережами і підходом network slicing.

Перший розділ сформував теоретичне підґрунтя для подальшої розробки моделі надання мережевих сервісів у SDN з підтримкою BYOC.

РОЗДІЛ 2 МОДЕЛЬ НАДАННЯ СЕРВІСІВ У SDN З ПІДТРИМКОЮ BYOC

2.1 Архітектурні підходи до реалізації BYOC

2.1.1 Вимоги до BYOC-моделі

Концепція BYOC у середовищі SDN потребує ретельно продуманої архітектури, оскільки вона змінює баланс між централізованим управлінням оператором та децентралізованим контролем клієнтів. Успішна реалізація BYOC неможлива без формалізації низки вимог, які стосуються безпеки [12], масштабованості, ізоляції користувачів та узгодженості політик. Передусім BYOC-модель повинна гарантувати цілісність інфраструктури. Це означає, що делегування частини функцій управління гостьовим контролерам не повинно створювати загрозу для глобальної стабільності мережі. Кожне рішення, яке надходить від зовнішнього застосунку, має перевірятися на відповідність загальним політикам оператора. У цьому контексті важливою вимогою стає наявність механізмів валідації та узгодження правил, які унеможливлюють виникнення конфліктів між локальними налаштуваннями клієнта та глобальними обмеженнями. Другою фундаментальною вимогою є забезпечення безпеки каналів взаємодії. Оскільки BYOC передбачає активний обмін повідомленнями між сервісним оркестратором, SDN-контролером і гостьовим контролером клієнта, необхідно застосовувати шифрування (TLS/SSL), механізми автентифікації (наприклад, SASL) та багаторівневу авторизацію. Кожен клієнтський застосунок повинен мати унікальні облікові дані, а рівень доступу обмежуватися виключно його власними ресурсами. Особливої уваги вимагає реалізація multi-tenancy, коли через один і той самий північний інтерфейс можуть взаємодіяти десятки чи сотні клієнтів. У цьому випадку потрібні чіткі механізми ізоляції, що гарантують неможливість несанкціонованого доступу до чужих сервісів. Ще однією вимогою є масштабованість системи. BYOC має функціонувати не лише в лабораторних

середовищах чи малих корпоративних мережах, але й у масштабних операторських інфраструктурах із тисячами клієнтів і десятками тисяч активних сервісів. Це означає, що архітектура повинна підтримувати горизонтальне масштабування контролерів, балансування навантаження між ними та гнучку розподілену обробку подій. Використання асинхронних протоколів обміну повідомленнями (наприклад, XMPP чи WebSockets) стає важливою умовою, адже синхронна модель запит–відповідь може не витримати високих обсягів трафіку. Крім того, BYOC вимагає прозорих механізмів моніторингу та аудиту. Оператор має мати можливість відстежувати всі дії, що виконуються гостьовими контролерами, включно з модифікацією політик, створенням або видаленням сервісів, а також обробкою подій безпеки. Для цього необхідні журнали подій (logs), системи збору телеметрії та аналітичні модулі, що дозволяють виявляти аномалії у поведінці клієнтських застосунків. Моніторинг є не лише технічною вимогою, а й важливим інструментом для формування довіри між оператором та користувачем. Ще однією вимогою є адаптивність BYOC-моделі до різних типів сервісів. Вона повинна підтримувати інтеграцію як із традиційними сервісами безпеки (VPN, IPS, брандмауери), так і з новітніми рішеннями, що базуються на NFV або хмарних платформах. Для цього NBI має бути достатньо універсальним, аби дозволяти стороннім застосункам управляти як окремими мережевими функціями, так і комплексними сервісними ланцюжками. Нарешті, BYOC-модель повинна бути економічно та організаційно доцільною. Вона має вписуватися у бізнес-моделі операторів, наприклад у формат «Control-as-a-Service», де клієнт отримує додаткові можливості управління. У такій схемі оператор зберігає контроль над критичними функціями, але водночас відкриває частину можливостей клієнту, що створює новий рівень персоналізації та підвищує конкурентоспроможність сервісів.

Отже, вимоги до BYOC-моделі можна узагальнити як поєднання технічних, безпекових, масштабованих та організаційних аспектів. Лише при комплексному врахуванні цих факторів концепція BYOC здатна реалізувати свій потенціал і стати надійним механізмом децентралізованого управління сервісами у SDN.

2.1.2 Структура взаємодії

Модель BYOC ґрунтується на взаємодії трьох ключових сторін: оператора, який керує інфраструктурою; користувача, що отримує доступ до сервісів; та SDN-контролера, який виконує роль посередника і координатора. Така тристороння структура є принципово новою для класичних мереж, оскільки розподіляє повноваження управління та надає клієнтові можливість брати участь у процесі контролю. У традиційних SDN усі рішення ухвалюються центральним контролером, що працює під керівництвом оператора. Клієнт може лише користуватися сервісами, але не має реальних інструментів впливу на їхню конфігурацію. У BYOC-контексті ця парадигма змінюється. Оператор залишається власником інфраструктури й визначає глобальні політики, однак частину функцій управління він делегує гостьовим контролерам користувачів. Така взаємодія можлива завдяки використанню відкритих інтерфейсів - насамперед NBI, який дозволяє клієнтським застосункам підключатися до сервісного оркестратора.

Оператор у цій системі виконує кілька ключових функцій. По-перше, він визначає глобальні політики безпеки, які є обов’язковими для всіх користувачів. По-друге, він забезпечує цілісність і стабільність роботи мережі, здійснюючи моніторинг і контроль за діями гостьових контролерів. По-третє, оператор виконує роль постачальника сервісів, надаючи клієнтам API та інструменти для самостійного управління. Таким чином, оператор виступає гарантом стабільності, безпеки та узгодженості роботи всієї інфраструктури.

Користувач, у свою чергу, отримує значно ширші повноваження, ніж у класичних моделях. Через власний гостьовий контролер він може створювати, змінювати та видаляти сервіси, керувати політиками доступу, налаштовувати VPN-тунелі, інтегрувати IPS чи інші функції безпеки [13-15]. У більш складних випадках користувач може реалізувати власні алгоритми оптимізації або захисту і застосувати їх у своїй частині мережі. Це забезпечує персоналізацію послуг і підвищує рівень кіберзахисту, оскільки клієнт може оперативно реагувати на локальні загрози.

SDN-контролер є центральним елементом, який координує взаємодію між оператором і користувачем. З одного боку, він реалізує глобальну політику, визначену оператором, а з іншого приймає інструкції від гостьових контролерів і транслює їх у правила для площини даних. Контролер забезпечує узгодження конфігурацій, уникнення конфліктів між різними клієнтами та контроль відповідності локальних дій глобальним обмеженням. Фактично він виступає одночасно «арбітром» і «транслятором» політик, підтримуючи баланс між централізованим та децентралізованим управлінням.

Завдяки такій тристоронній структурі BYOC дозволяє побудувати більш гнучку модель управління, у якій кожна сторона виконує чітко визначену роль. Оператор гарантує стабільність та цілісність, користувач отримує свободу персоналізованого контролю, а SDN-контролер координує і технічно реалізує взаємодію. У результаті формується модель спільного управління, що поєднує переваги централізації з гнучкістю децентралізованого підходу.

2.1.3 Розмежування повноважень управління

Одним із найважливіших завдань у побудові моделі BYOC є чітке визначення кордонів між тим, що контролює оператор, і тим, що передається у розпорядження користувача. Якщо у традиційній архітектурі SDN оператор фактично володіє монополією на управління всіма аспектами мережі, то BYOC змінює цей баланс, делегуючи частину повноважень клієнту. Такий підхід відкриває нові можливості для персоналізації сервісів, але водночас створює ризики, пов’язані з безпекою та узгодженістю рішень. Саме тому розмежування управління є фундаментальною умовою ефективності моделі.

Оператор зберігає контроль над ключовими елементами інфраструктури, які визначають цілісність і стабільність мережі. До таких аспектів належить управління фізичними ресурсами, глобальними правилами безпеки, базовими механізмами маршрутизації та розподілом пропускної здатності. Оператор також відповідає за моніторинг і аудит усіх дій, що виконуються гостьовими контролерами, та має право блокувати ті з них, які суперечать узгодженим

політикам. Його роль полягає у збереженні «каркасу» мережі, який не можна порушити навіть у разі децентралізованих змін.

Користувач, у свою чергу, отримує свободу в управлінні власними сервісами. Він може визначати параметри VPN-з'єднань, задавати політики доступу, підключати або відключати IPS-модулі, встановлювати правила шифрування чи створювати додаткові механізми моніторингу. У більш складних сценаріях клієнту може бути делеговано управління сервісними ланцюжками (Service Chains), де він самостійно обирає порядок та логіку застосування віртуальних функцій безпеки. Водночас кожне з цих рішень перевіряється центральною системою, щоб уникнути конфліктів з іншими клієнтами або з глобальною політикою оператора.

SDN-контролер у цій схемі виступає посередником, який забезпечує узгодження локальних і глобальних змін. Він валідує команди, отримані від гостьових контролерів, транслює їх у правила для площини даних і гарантує, що жодна дія користувача не може загрожувати роботі мережі в цілому. Якщо виникає конфлікт, пріоритет завжди залишається за глобальними політиками оператора, проте у більшості випадків контролер забезпечує «мирне співіснування» локальних налаштувань із глобальною стратегією.

Таким чином, розмежування повноважень управління у BYOC можна охарактеризувати як симбіоз централізованого та децентралізованого підходів. Оператор гарантує стабільність і безпеку, клієнт отримує гнучкість і персоналізацію, а контролер координує їхню взаємодію. У результаті формується модель спільної відповідальності, де успішність системи залежить від чіткості визначення прав і обов'язків кожної сторони, а також від наявності ефективних механізмів контролю та узгодження рішень.

2.2 Модель життєвого циклу сервісу з підтримкою BYOC

Концепція BYOC змінює класичне розуміння життєвого циклу мережевих сервісів у SDN. Якщо у традиційній моделі всі стадії контролювалися виключно оператором, то інтеграція BYOC дозволяє децентралізувати управління,

створюючи дворівневу систему, у якій взаємодіють оператор і користувач. Такий підхід забезпечує баланс між стабільністю інфраструктури та гнучкістю персоналізованих сервісів. Життєвий цикл у BYOC розглядається як безперервний процес, що включає створення, конфігурацію, моніторинг, модифікацію, контроль безпеки та завершення роботи сервісу, причому окремі функції виконуються на різних рівнях.

2.2.1 Операторський рівень життєвого циклу

Операторський рівень життєвого циклу сервісу у моделі SDN з підтримкою BYOC відповідає за глобальне управління ресурсами, узгодження політик та забезпечення стабільності інфраструктури. Він визначає «каркас» функціонування всієї мережі, на основі якого реалізуються клієнтські сервіси. Цей процес складається з кількох ключових етапів, що дозволяють організувати повний життєвий цикл послуги - від створення до її виведення з експлуатації. Логіку роботи такого життєвого циклу ілюструє рисунок 2.1.

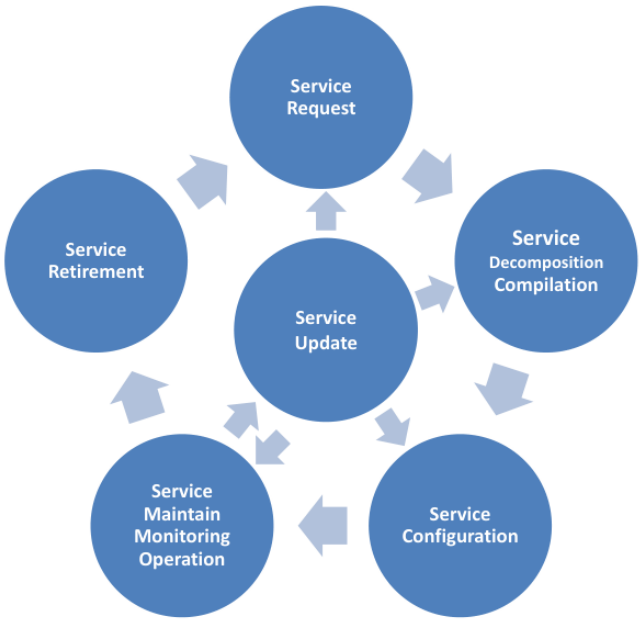

Рисунок 2.1 – Операторський рівень життєвого циклу сервісу

Першим кроком є отримання та обробка запиту на сервіс. Запит на створення чи модифікацію надходить від користувацького порталу через NBI.

Менеджер запитів узгоджує угоду про SLA та високорівневу специфікацію сервісу. На цьому етапі оператор зобов’язаний перевірити наявність необхідних ресурсів у момент запланованого розгортання. Якщо ресурсів недостатньо, запит ставиться у чергу до моменту їх вивільнення. Далі відбувається декомпозиція та компіляція сервісу. Високорівнева модель розкладається на кілька елементарних підмоделей, що описують складові частини сервісу. Ці підмоделі надходять до сервісного компілятора, який генерує набір конфігурацій мережевих ресурсів. Таким чином, складний сервіс перетворюється на сукупність конфігурацій, які можна безпосередньо застосувати до інфраструктури. Наступним етапом є конфігурація сервісу. На основі створених компілятором конфігурацій ініціалізуються та резервуються віртуальні ресурси. Після цього на них розгортаються задані параметри, і сервіс фактично починає функціонувати. Варто підкреслити, що у середовищі BYOC ця фаза може частково делегуватися клієнту, але остаточна перевірка та запуск відбуваються під контролем оператора. Після введення сервісу в експлуатацію важливим етапом є підтримка, моніторинг та експлуатація. Автоматизовані механізми відстежують продуктивність, доступність і якість сервісу, а також ведуть журнали подій. Система моніторингу повинна своєчасно виявляти відхилення у роботі та надсилати відповідні сповіщення. Це забезпечує не лише стабільність, але й підвищений рівень кіберзахисту, оскільки оператор має змогу оперативно реагувати на загрози або аномалії у мережевому трафіку. Згодом може виникнути потреба у оновленні сервісу. Зміни інфраструктури або технічна еволюція призводять до того, що певні параметри слід модифікувати. Це може відбуватися прозоро для користувача або ж вимагати часткового повторного проходження початкових етапів життєвого циклу (декомпозиції, компіляції чи конфігурації). У BYOC-контексті клієнтський контролер може ініціювати такі зміни, але їх застосування завжди перевіряється оператором. Завершальним етапом є виведення сервісу з експлуатації. Коли надходить запит на завершення роботи, конфігурації сервісу поступово вивільняються з інфраструктури, а зарезервовані ресурси повертаються у спільний пул. Це дозволяє оптимізувати

використання ресурсів і забезпечує можливість швидкого їх повторного розподілу для нових запитів.

Таким чином, життєвий цикл на операторському рівні охоплює всі критично важливі етапи управління сервісами від створення до завершення і виступає фундаментом для роботи клієнтських механізмів BYOC. Він є достатньо універсальним, щоб підтримувати різні типи додатків (наприклад, VPN, IPS чи сервіси маршрутизації), при цьому зберігаючи баланс між централізованою стабільністю та децентралізованою гнучкістю.

2.2.2 Клієнтський рівень життєвого циклу

Клієнтський рівень у моделі BYOC забезпечує можливість активного управління життєвим циклом сервісів з боку кінцевих користувачів. Цей рівень охоплює різні сценарії взаємодії, що залежать від типу клієнтських застосунків, і формує персоналізовану частину життєвого циклу сервісу. Життєвий цикл сервісу на стороні клієнта можна умовно поділити на кілька рівнів складності, залежно від функціональності застосунків. У найпростішому випадку клієнтські застосунки створюють мережевий сервіс через NBI, узгоджують SLA із системою та визначають необхідні характеристики сервісу. Після завершення терміну дії SLA відбувається завершення сервісу. Такий життєвий цикл складається лише з двох фаз Service creation та Service retirement. Більш складний сценарій передбачає, що клієнтські застосунки не лише створюють сервіс, а й здійснюють його моніторинг. Вони отримують події та статистику від SDN-контролера через NBI, аналізують параметри продуктивності, доступності та безпеки. Таким чином, життєвий цикл доповнюється етапом Service monitoring, що дозволяє клієнту контролювати SLA в реальному часі. У найбільш просунутому випадку клієнт отримує ще й можливість модифікувати сервіс на основі даних моніторингу. Події, які надходять від SDN-контролера, можуть активувати алгоритми всередині клієнтського застосунку, які здійснюють автоматичне переналаштування мережевих ресурсів. Це створює зворотний зв’язок між моніторингом і конфігурацією та дозволяє оперативно реагувати на

зміни у середовищі. Такий життєвий цикл складається з чотирьох фаз: Service creation, Service monitoring, Service modification, Service retirement.

Узагальнена модель клієнтського життєвого циклу представлена на рисунку 2.2.

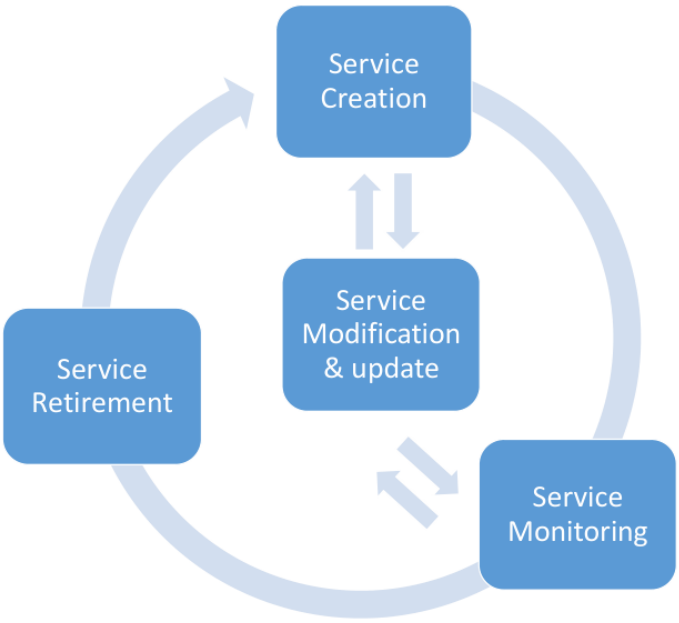

Рисунок 2.2 – Узагальнена модель клієнтського життєвого циклу

Вона включає всі описані вище кроки, а також додаткову фазу Service modification and update, що виникає у випадках, коли операторська інфраструктура потребує змін у процесі роботи сервісу. Це може бути викликано технічними проблемами, модернізацією обладнання чи зміною політик управління. Таким чином, клієнтський рівень інтегрується з операторським, створюючи динамічну двошарову модель управління сервісом.

Важливо підкреслити, що клієнтський рівень у BYOC не обмежується лише базовими сценаріями. У контексті кібербезпеки користувачі можуть інтегрувати власні алгоритми виявлення та блокування атак, застосовувати індивідуальні IPS-рішення чи додаткові механізми шифрування. Це забезпечує гнучкість і персоналізацію, але водночас створює нові виклики для оператора, який зобов’язаний валідувати кожну дію клієнта, щоб уникнути загроз для всієї мережевої інфраструктури.

2.2.3 Двошарова інтеграція

Двошарова інтеграція у моделі BYOC відображає синергію між операторським та клієнтським рівнями управління сервісами. Вона є фундаментом архітектури, що поєднує централізовану стабільність із децентралізованою гнучкістю. На практиці це означає, що оператор відповідає за загальну політику, ресурси та безпеку, тоді як клієнт отримує свободу у межах власних сервісів, але обов'язково узгоджує свої дії з інфраструктурою оператора.

Двошарова інтеграція забезпечує безперервність життєвого циклу сервісу. Створення сервісу ініціюється клієнтом через гостьовий контролер, проте остаточне виділення ресурсів та запуск відбуваються під контролем оператора. Моніторинг здійснюється на обох рівнях: клієнт відстежує параметри, що стосуються його SLA [16], тоді як оператор контролює глобальні показники продуктивності та безпеки. У випадку модифікацій клієнт може ініціювати зміни, але вони завжди проходять перевірку на відповідність загальним політикам. Якщо ж необхідні зміни викликані технічними проблемами або еволюцією інфраструктури, то оновлення відбуваються з боку оператора, який інформує клієнта через NBI. Завершення сервісу також має двошаровий характер. Клієнт може подати запит на припинення, але остаточне виведення ресурсу з експлуатації виконується оператором, який вивільняє відповідні ресурси.

Ключовим інструментом двошарової інтеграції виступає NBI. Саме через нього відбувається взаємодія між клієнтськими додатками та сервісним оркестратором оператора. Використання REST API [17], XMPP [18] або інших механізмів дозволяє стандартизувати цей обмін і водночас забезпечити безпеку та ізоляцію клієнтів у багатокористувацькому середовищі. У BYOC-контексті NBI не є лише технічним протоколом. Він перетворюється на адміністративний кордон, який чітко визначає, які функції залишаються під контролем оператора, а які делегуються клієнту. Особливе значення має інтеграція у сценаріях динамічної оркестрації сервісів. Наприклад, у мережах 5G клієнти можуть замовляти «network slices» зі специфічними параметрами безпеки, затримки чи

пропускної здатності. Оператор створює базову інфраструктуру зрізу, а клієнт, використовуючи гостьовий контролер, конфігурує його відповідно до власних потреб. Завдяки двошаровій інтеграції забезпечується поєднання масштабованості операторських ресурсів та персоналізації клієнтських сервісів.

Важливим елементом є також узгодження рішень у сфері кібербезпеки. Оператор визначає глобальні політики безпеки, наприклад, фільтрацію шкідливого трафіку або застосування IPS для всієї інфраструктури. Водночас клієнт може додатково реалізувати власні механізми захисту, такі як локальні IDS чи VPN, що діють виключно у межах його сервісу. Така інтеграція дозволяє уникнути надмірної централізації та водночас зберегти контроль над цілісністю системи.

Двошарова інтеграція формує модель співуправління, де оператор і клієнт виступають рівноправними учасниками процесу, проте з різними зонами відповідальності. Для оператора головним завданням є стабільність, узгодженість і безпека інфраструктури, тоді як для клієнта гнучкість, адаптивність і можливість персоналізованого управління сервісами. Взаємодія цих двох рівнів створює середовище, здатне до швидкого реагування на зміни, підтримки високих стандартів безпеки та одночасного розвитку нових бізнес-моделей, зокрема «Control-as-a-Service».

2.3 Основи інтеграції концепції BYOC у SDN

У класичній архітектурі програмно-визначених мереж ключову роль відіграє NBI, який забезпечує взаємодію між контролером і прикладними застосунками. Саме через нього прикладний рівень отримує абстрактоване уявлення про мережу і може впливати на її поведінку без знання деталей фізичної інфраструктури. Це дозволяє реалізувати механізми програмованості, розширюваності та відкритості мереж, у тому числі залучення сторонніх застосунків для управління. Завдяки NBI оркестратор, керований оператором, отримує можливість контролювати ступінь відкритості мережі та надавати клієнтам доступ до певних функцій. Така гнучкість формує адміністративний

кордон між внутрішнім керуванням операторської інфраструктури та зовнішніми клієнтами. Концепція BYOC логічно випливає з цієї архітектури і передбачає делегування частини або навіть усієї функції управління сервісом сторонньому застосунку, відомому як гостьовий контролер (Guest Controller, GC), який належить зовнішньому клієнту (див. рисунок 2.3).

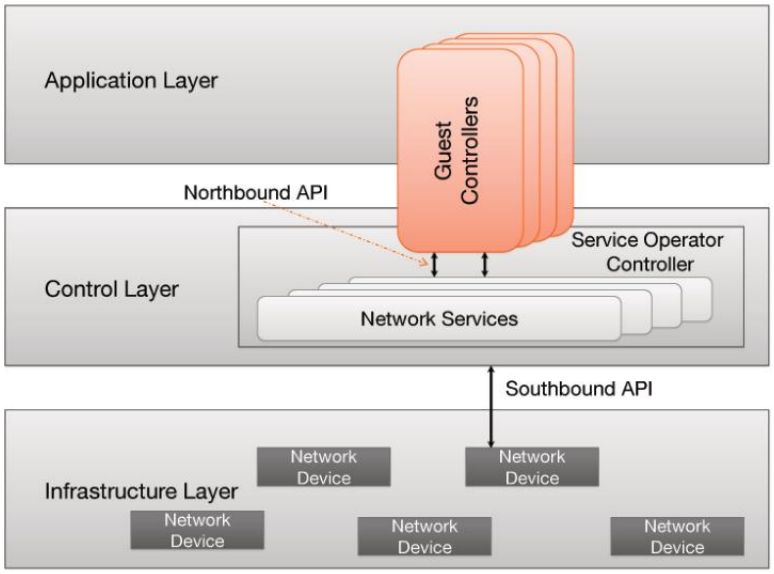

Рисунок 2.3 – Місце BYOC в архітектурі SDN

На рівні логічної моделі GC займає проміжне положення між площиною контролю та прикладним рівнем, взаємодіючи з SDN-контролером через відкриті API. Використання BYOC має низку важливих переваг. По-перше, це дозволяє зменшити навантаження на центральний контролер, оскільки частина рішень та обробки переноситься на зовнішні застосунки. У традиційних SDN-архітектурах централізація логіки управління може призводити до перевантаження єдиного вузла, особливо коли одночасно функціонує велика кількість сервісів. Децентралізація завдяки BYOC допомагає вирішити цю проблему. По-друге, розподіл контролю знижує ризики, пов’язані з конфіденційністю даних. У випадку класичної моделі вся інформація про трафік проходить через один контролер, що може створювати проблеми з безпекою та довірою. Використання GC дозволяє користувачу самостійно впроваджувати алгоритми захисту, контролювати політики шифрування чи блокування та залишати частину даних локально. Окрім технічних аспектів, BYOC створює нові можливості для бізнес-

моделей. Делегування функцій управління дозволяє оператору надавати клієнтам спеціалізовані API для інтеграції власних алгоритмів. У такій моделі оператор фактично виступає посередником і забезпечує безпечну платформу для впровадження стороннього інтелектуального функціоналу. Водночас впровадження BYOC породжує нові виклики. Найважливішим із них є перевірка коректності рішень, прийнятих гостьовим контролером. Політики, які генерує зовнішній застосунок, мають бути узгоджені з глобальними правилами оператора, інакше існує ризик порушення цілісності та безпеки всієї мережі. Тому інтеграція BYOC потребує механізмів валідації, контролю відповідності політик і дотримання узгоджених стандартів безпеки. Цей аспект залишається відкритим напрямом для подальших досліджень і вдосконалень.

BYOC у поєднанні з можливостями NBI створює умови для формування більш гнучкої, масштабованої та безпечної архітектури SDN. Делегування частини функцій управління клієнтам дозволяє зменшити централізоване навантаження, покращити конфіденційність і надати користувачам новий рівень персоналізації сервісів. При цьому подальший розвиток концепції пов’язаний із пошуком балансів між відкритістю системи, дотриманням політик оператора та необхідністю забезпечення високого рівня кіберзахисту.

2.4 Взаємодія BYOC-застосунку з SDN-контролером через Northbound Interface

Ефективне впровадження концепції BYOC у середовищі SDN безпосередньо залежить від функціональності NBI, який забезпечує комунікацію між сервісним оркестратором (SO) та гостьовим контролером (GC). Саме цей інтерфейс формує адміністративний кордон між оператором, що керує інфраструктурою, та клієнтом, який може отримати частковий чи повний контроль над власними сервісами. Основні функції NBI охоплюють створення, конфігурацію, модифікацію й моніторинг сервісів, а також управління специфічними подіями, що передаються до GC. Для цього реалізується набір програмних пакетів, які дозволяють оператору надавати клієнтам сервісний рівень абстракції без необхідності доступу до деталей фізичної мережі. Такий

підхід гарантує масштабованість, багатокористувацьку підтримку та безпеку при спільному використанні інтерфейсу різними клієнтами. Однією з ключових характеристик NBI є підтримка як синхронних, так і асинхронних взаємодій. У синхронному режимі клієнтський застосунок формує запит і одразу отримує відповідь, що зручно для створення або модифікації сервісу. У асинхронному режимі взаємодія будується на механізмі підписки та повідомлень. Коли у мережі відбувається подія, оркестратор створює нотифікацію та розсилає її всім клієнтам, що підписалися на відповідний сервіс. Такий підхід особливо цінний для оперативного управління безпекою, коли гостьовий контролер має швидко отримати інформацію про інциденти, наприклад OpenFlow PacketIn, і сформувати відповідну реакцію.

Комунікація між GC і SO реалізується за принципом Push/Pull. Ця модель запозичена з вебтехнологій і адаптована для управління сервісами у SDN. Клієнт підписується на конкретний сервіс, а оркестратор через спеціальні модулі Decision Engine та Service Dispatcher визначає, які повідомлення повинні бути оброблені всередині SO, а які передані до гостьового контролера. DE приймає події від мережевих елементів і ухвалює рішення про їх подальшу обробку. Якщо подія має бути передана зовнішньому GC, DE надсилає її до SD, який розповсюджує повідомлення всім підписаним контролерам. Така структура дозволяє поєднати централізований контроль із можливістю динамічного делегування окремих функцій користувачам. Логіку такої взаємодії ілюструє рисунок 2.4.

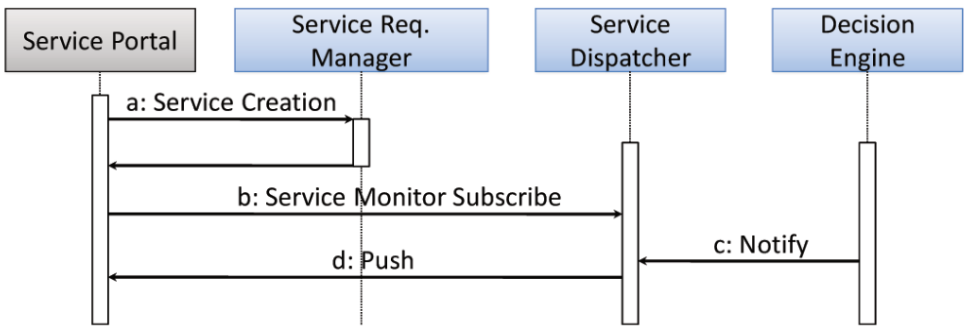

Рисунок 2.4 – Комунікація між компонентами

Ще одним важливим аспектом є підтримка багатокористувацького доступу. Оскільки один і той самий NBI використовується різними клієнтами для керування власними сервісами, інтерфейс повинен забезпечувати ізоляцію доступу та захист даних. Для цього впроваджується система автентифікації та авторизації, яка гарантує, що кожен клієнт має доступ лише до своїх ресурсів і подій. Це дозволяє реалізувати модель multi-tenant, яка є необхідною умовою для масштабованих хмарних і операторських середовищ.

Таким чином, NBI у поєднанні з концепцією BYOC виступає не лише технічним інтерфейсом для інтеграції клієнтських застосунків, але й стратегічним інструментом, що підвищує гнучкість, масштабованість і безпеку SDN-мереж. Завдяки йому можливо делегувати управління окремими функціями стороннім контролерам без шкоди для цілісності інфраструктури, водночас забезпечуючи високий рівень кіберзахисту й персоналізації сервісів.

2.5 Реалізації північного інтерфейсу у SDN

Сучасні SDN-контролери активно впроваджують механізми NBI, орієнтовані на надання відкритих сервісів прикладному рівню. Реалізація NBI зазвичай базується на веб-технологіях, що використовують HTTP як транспортний протокол та XML- або JSON-повідомлення для обміну даними. Такий підхід відомий як вебсервіси і розглядається як розподілена проміжна технологія, яка забезпечує уніфіковану взаємодію між прикладними системами та контролером.

Найпоширенішим стандартом, на основі якого реалізуються веб-сервіси у SDN, є REST. Використання REST API дозволяє формалізувати звернення прикладних програм до функцій, які надає контролер, і робить процес інтеграції простим і масштабованим. У межах такого інтерфейсу прикладний рівень може викликати функції для обчислення оптимального шляху, уникнення петель, налаштування політик маршрутизації та безпеки. Це суттєво підвищує гнучкість системи, оскільки розробники можуть створювати власні додатки, які керують мережею відповідно до специфічних вимог користувача чи бізнес-моделі

оператора. Як альтернативу REST дослідники пропонують розглядати XMPP. Завдяки своїй зрілості та простоті XMPP здатен забезпечити асинхронну обробку повідомлень, що робить його привабливим кандидатом для впровадження у сценаріях BYOC, де критично важливим є своєчасне отримання повідомлень від контролера. Ще одним потенційним напрямом розвитку є використання WebSockets, які дозволяють встановлювати постійне двостороннє з'єднання між клієнтом і сервером, проте ця технологія поки що перебуває на етапі активного вдосконалення.

Використання NBI поширюється і на системи оркестрації. Зокрема, платформи OpenStack Neutron або VMware vCloud Director можуть взаємодіяти з SDN-контролером через відкриті API для керування сервісами у хмарному середовищі. Такий підхід відкриває можливість реалізації концепцій NaaS та IaaS, коли мережеві та обчислювальні ресурси надаються динамічно і програмно керуються через єдиний оркестраційний шар. Це створює основу для побудови гнучких, масштабованих і захищених інфраструктур, де клієнти отримують доступ до мережевих функцій через стандартизовані API, не залежачи від специфіки фізичного обладнання.

2.5.1 REST у реалізації NBI

REST є архітектурним стилем вебсервісів, який набув найбільшого поширення для реалізації NBI у SDN. Він базується на ресурсному підході, де кожен ресурс має унікальний ідентифікатор у вигляді URL, а доступ до нього здійснюється за допомогою базових HTTP-операцій, таких як GET, PUT, POST та DELETE. Ці операції забезпечують єдиний інтерфейс взаємодії, у якому ресурси представляють як стан, так і функціональність застосунків. Особливістю REST є те, що він не обмежений конкретним форматом даних. Хоча XML історично застосовувався для обміну інформацією у вебсервісах, REST дозволяє використовувати й інші формати, включно з JSON, CSV, plain text, RSS чи навіть HTML. Така гнучкість робить REST універсальним і простим у реалізації.

Популярність REST у світі SDN пояснюється його простотою, продуктивністю та масштабованістю. Він легший у використанні, ніж традиційні вебсервіси на базі SOAP, які раніше застосовувались для специфікації NBI, але швидко були відкинуті через складність та надмірність. REST не потребує дорогих або спеціалізованих інструментів для інтеграції, що дозволяє операторам і розробникам швидко створювати клієнтські застосунки для управління мережею. У контексті SDN це означає можливість легкої реалізації функцій маршрутизації, обчислення оптимальних шляхів, управління безпекою чи моніторингу.

Разом із тим REST має і певні обмеження. Найважливішим недоліком є відсутність вбудованої підтримки асинхронних можливостей та проблеми з безпечним багатокористувацьким доступом. Оскільки REST здебільшого використовує модель «запит–відповідь», асинхронна взаємодія за його допомогою реалізується або через періодичні запити клієнта (polling), що створює додаткове навантаження на мережу, або через допоміжні технології, наприклад AJAX. Останні також базуються на ініціативі з боку клієнта, що не завжди відповідає вимогам динамічного управління у середовищі SDN.

2.5.2 XMPP у реалізації NBI

Одним із перспективних підходів до реалізації північного інтерфейсу у SDN є використання протоколу XMPP, також відомого як Jabber [19]. Первинно XMPP був створений для сервісів миттєвого обміну повідомленнями, проте згодом його функціональність значно розширилася. Протокол базується на потоковій технології XML і дозволяє здійснювати обмін структурованими повідомленнями між будь-якими вузлами мережі. Кожен вузол ідентифікується унікальним Jabber ID (JID), який має формат node@domain/resource. Така структура дозволяє реалізувати гнучку систему ідентифікації клієнтів, контролерів та сервісних елементів у межах одного середовища. Гнучкість XMPP підтверджується наявністю численних розширень (XMPP Extension Protocols), що дають змогу адаптувати його під різні сценарії використання - від простих застосунків

миттєвого обміну повідомленнями до складних систем для хмарних обчислень і віддаленого керування. Однією з ключових переваг XMPP є підтримка асинхронного обміну повідомленнями, що усуває необхідність періодичного опитування (polling) клієнтом сервера. Завдяки цьому XMPP забезпечує push-механізм, коли повідомлення або нотифікації надсилаються клієнту відразу після їх появи. Це робить його особливо ефективним для середовищ BYOC, де необхідно швидко передавати сигнали управління від сервісного оркестратора до гостьових контролерів.

Архітектура XMPP-базованого NBI передбачає, що модулі GC, SRM і SD містять вбудованих XMPP-клієнтів, кожен із власним JID. Користувач ініціює створення або модифікацію сервісу через XMPP-повідомлення до SRM, яке передається через сервісний портал. У процесі взаємодії може бути укладена угода про рівень обслуговування (SLA), що визначає характеристики сервісу та параметри контролю, які передаються зовнішньому GC. Після цього сервіс розгортається у мережі, а клієнт отримує підтвердження із зазначенням JID модуля SD. Саме через цей JID гостьовий контролер підписується на повідомлення моніторингу та керування, які надходять у режимі реального часу. Таким чином, XMPP забезпечує асинхронну модель Publish-Subscribe, що може реалізовуватися як у базовому протоколі, так і через розширення XEP-0060. Логічну схему архітектури XMPP-базованого NBI наведено на рисунку 2.5.

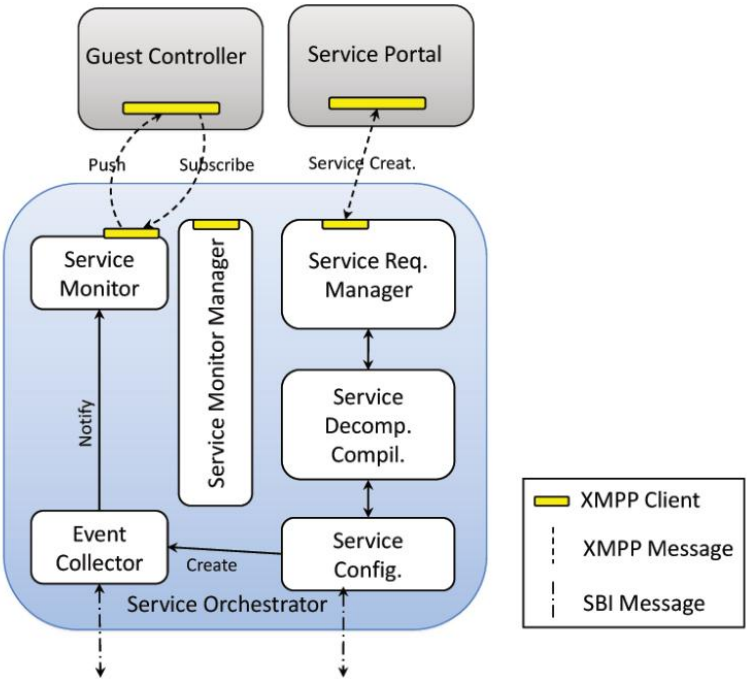

Рисунок 2.5 – Логічна схема архітектури XMPP-базованого NBI

Протокол XMPP також добре підходить для побудови безпечного NBI. Він має вбудовану підтримку шифрування каналів зв’язку за допомогою TLS, а також механізми автентифікації та авторизації через SASL. Це дозволяє обмежувати доступ лише авторизованим клієнтам та унеможливлювати перехоплення керуючих повідомлень. Важливо, що більшість бібліотек XMPP реалізують ці механізми за замовчуванням, що спрощує інтеграцію.

Для зменшення складності опису сервісів передбачено використання моделі даних YANG. Ця мова дозволяє описувати параметри сервісів на високому рівні абстракції, які потім можуть бути перетворені у XML-подання YIN та передані через XMPP. Такий підхід забезпечує уніфікацію опису сервісів, повторне використання моделей і просту інтеграцію із вже існуючими інструментами управління.

Використання XMPP як основи для реалізації північного інтерфейсу у SDN забезпечує низку переваг: асинхронну взаємодію у режимі реального часу, вбудовану безпеку та масштабованість. Це робить XMPP ефективним кандидатом для впровадження у середовищах з підтримкою BYOC, де критично важливими є швидкість реакції на події, багатокористувацька підтримка і гарантії конфіденційності даних.

2.6 Висновки до розділу 2

В другому розділі було обґрунтовано модель надання мережевих сервісів у SDN з підтримкою BYOC та показано, як децентралізація керування доповнює класичну централізацію, не руйнуючи цілісності інфраструктури. Розкрито архітектурні засади BYOC. Показано, що оператор формує глобальну політику, безпеку й управління ресурсами, тоді як користувач через гостьовий контролер персоналізує власні сервіси, включно з безпековими функціями (VPN, IDS/IPS, QoS), а SDN-контролер виступає «арбітром» узгодження конфігурацій.

Запропонована модель життєвого циклу сервісів із підтримкою BYOC представлена у двох площинах. На операторському рівні формалізовано послідовність кроків від приймання запиту і узгодження SLA до декомпозиції та компіляції сервісу, конфігурації, моніторингу, експлуатації, оновлення й виведення з експлуатації. На клієнтському рівні деталізовано три типи сценаріїв. Показано, що саме двошарова інтеграція цих рівнів забезпечує безперервність життєвого циклу, швидку реакцію на події та збалансоване співуправління.

Окремо проаналізовано NBI як ключовий механізм зв’язку між оркестратором і гостьовий контролером. Обґрунтовано доцільність комбінування синхронних і асинхронних взаємодій. Порівняно реалізації NBI. Зроблено висновок, що гібридний підхід до NBI є практично доцільним у операторських і хмарних середовищах.

У підсумку розділ встановив теоретичну й прикладну основу BYOC для SDN. Визначено архітектурні вимоги, життєві цикли й інтеграційні механізми, що дозволяють одночасно підвищити гнучкість сервісів, швидкість реагування на кіберзагрози та масштабованість платформи, зберігаючи контроль оператора над політиками та безпекою.

РОЗДІЛ 3 АРХІТЕКТУРА ТА РЕАЛІЗАЦІЯ BYOC-СЕРВІСУ IPS У СЕРЕДОВИЩІ SDN

3.1 Проєктування BYOC для сервісу кібербезпеки

Концепція BYOC у сфері сервісів кібербезпеки виникла як відповідь на обмеження класичної моделі управління мережею, де всі контрольні функції централізовано зосереджені у провайдера послуг. У традиційних підходах користувач отримує доступ лише до готового сервісу (наприклад, IPS), не маючи впливу на внутрішні механізми його роботи, зокрема на засоби забезпечення безпеки. Це створює низку проблем: обмежену гнучкість, залежність від політик оператора та неможливість швидко реагувати на нові типи атак.

Модель BYOC долає ці недоліки завдяки частковій передачі повноважень від SO до самого клієнта. Для цього у клієнтському домені розгортається окремий GC, що виконує функції контролю за визначеними елементами сервісу. Саме GC стає для користувача «точкою входу» до управління, забезпечуючи прямий контроль над безпековими компонентами та узгодження його дій із глобальною мережею, яку адмініструє SO. У контексті сервісів кібербезпеки ключовим прикладом є інтеграція IPS у сервіс VPN. У класичній архітектурі з використанням IDS [20] передбачено два основних компоненти (див. рисунок 3.1).

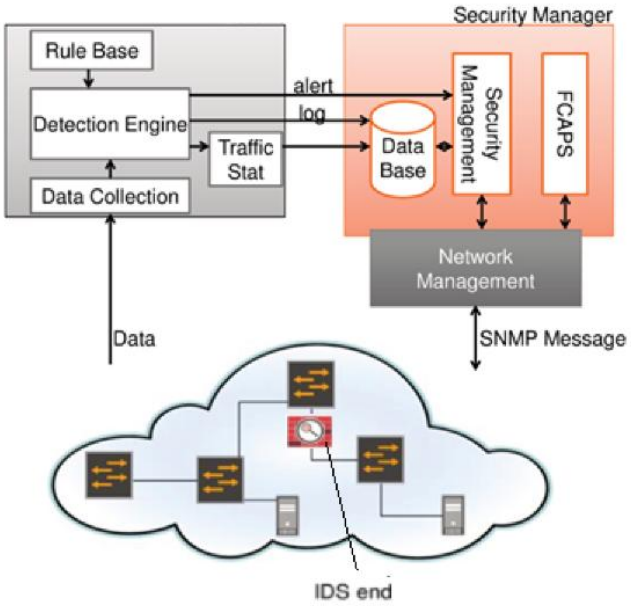

Рисунок 3.1 – Класична модель управління мережею SDN

IDS -end, що розміщується у стратегічних точках мережі (наприклад, шлюзах між локальною мережею та Інтернетом). Він аналізує вхідний і вихідний трафік у режимі реального часу та виявляє аномалії. Security Manager - центральний модуль управління, який отримує від IDS -end інформацію про підозрілі події, виконує аналіз і ухвалює рішення щодо відповідних дій (блокування IP, зміна маршрутизації, активація фільтрів тощо).

У класичних рішеннях, SM завжди перебуває під контролем оператора. У результаті користувач обмежується лише сповіщеннями про атаки, але не має змоги самостійно керувати IDS. Інтеграція BYOC змінює цей баланс. У запропонованій моделі GC клієнта містить власний SM, тоді як IDS -end продовжують функціонувати на рівні інфраструктури, що належить оператору. Таким чином, модель розподіляє контрольні функції. Оператор (SO) відповідає за інфраструктурну частину - створення, налаштування та підтримку VPN-з’єднання, розгортання IDS-end на вузлах доступу, маршрутизацію та моніторинг ресурсів. Клієнт (GC) отримує контроль над логікою IPS, я саме обробку повідомлень log та alert, вибір політики реагування, формування рішень для блокування чи перенаправлення трафіку.

3.2 Розширення архітектури IPS засобами SDN

Класичні системи передбачають прямий канал взаємодії між кінцевими вузлами IDS-end та центральним менеджером безпеки SM. Така модель ефективна для централізованого контролю, однак вона обмежує можливості розширення та інтеграції додаткових сервісів, зокрема у випадку, коли частина управлінських функцій має бути передана клієнтові за концепцією BYOC.

Головна ідея нашої схеми полягає у відокремленні функцій управління від рівня пересилання даних, що дозволяє створити більш гнучку, модульну та масштабовану архітектуру. У межах інтегрованої моделі роль посередника між IDS-end та SM бере на себе SO. Саме SO тепер відповідає не лише за життєвий цикл сервісу, а й за організацію обміну повідомленнями між модулями. Для цього він використовує два типи інтерфейсів. NBI відкриває канали взаємодії між SO та GC клієнта. Завдяки цьому NBI клієнт може отримувати повідомлення від IDS-end та реалізовувати власну логіку безпеки через SM. SBI забезпечує управління мережевими пристроями на рівні інфраструктури (наприклад, OpenFlow Switch), дозволяючи SO застосовувати рішення, що були сформовані або локально в SO, або зовнішнім GC. Завдяки такій інтеграції SO стає центральною керуючою ланкою, яка відокремлює рівень аналізу від рівня виконання. Він підтримує як внутрішню логіку управління, так і відкриває доступ до BYOC-інтерфейсу для клієнтів.

Архітектурна схема рішення, де чітко відображені SO, NBI та SBI, наведена на рисунку 3.2.

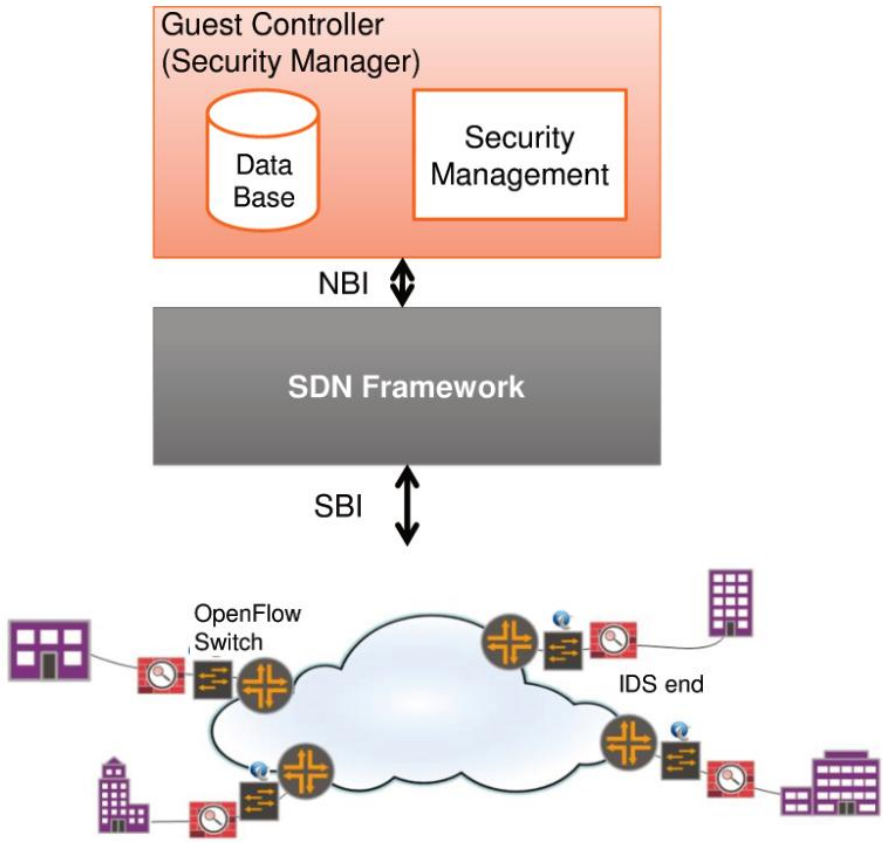

Рисунок 3.2 – Розроблена модель управління мережею SDN

Вона демонструє, що рішення дозволяє не лише зберегти централізований контроль на рівні оператора, а й водночас делегувати частину функцій клієнтові, створюючи гнучку багаторівневу модель IPS у середовищі SDN.

У запропонованій архітектурі безпосередньо на сайтах клієнта розгортаються IDS-end, які виконують роль сенсорів мережевого трафіку та формують повідомлення log і alert. Однак самі по собі IDS-end не реалізують повної функціональності IPS, оскільки відповідають лише за виявлення аномалій. Перетворення цієї системи на повноцінний IPS відбувається завдяки поєднанню кількох компонентів. IDS-end забезпечує моніторинг, SM, розміщений у складі GC, ухвалює рішення щодо реагування, а SO застосовує це рішення до інфраструктури за допомогою OpenFlow Switch. Таким чином, IPS у середовищі SDN постає як інтегрований сервіс, де функції виявлення, аналізу та блокування розподілені між різними модулями, що забезпечує гнучкість і підтримує концепцію BYOC.

3.3 Реалізація BYOC для VPN з інтегрованим IPS

У рамках дослідження запропоновано приклад використання концепції BYOC для побудови сервісу, де класичний VPN інтегрується з механізмами IPS. Основна ідея полягає у тому, що клієнт отримує не лише доступ до зашифрованого тунельного з'єднання між своїми сайтами, але й можливість самостійно управляти логікою захисту трафіку, тоді як оператор сервісу (SO) забезпечує інфраструктурну частину й життєвий цикл розгорнутого сервісу. Це створює багаторівневу архітектуру, де інтереси обох сторін (оператора та клієнта) збалансовані завдяки чіткому розмежуванню відповідальності.

Фізична топологія захищеного VPN з IPS наведена на рисунку 3.3

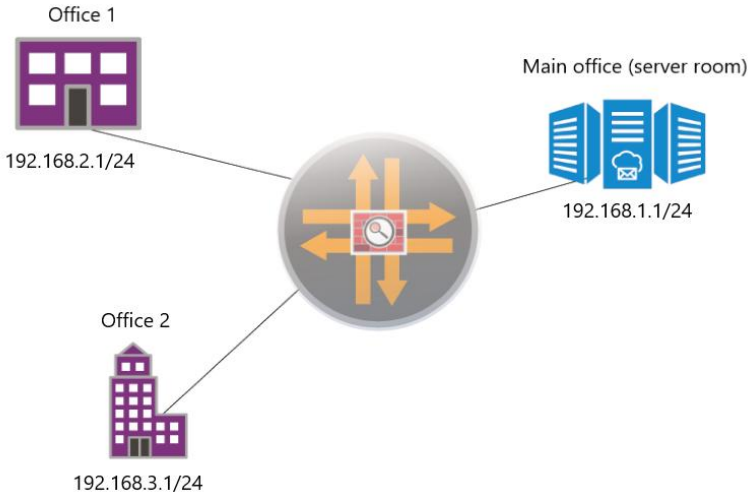

Рисунок 3.3 – Топологія захищеного VPN з IPS

У цьому прикладі клієнт має три географічно віддалені майданчики офіс1, офіс 2 та головний офіс. Всі вони з'єднані за допомогою VPN, що гарантує конфіденційність і цілісність переданих даних. Проте відмінністю від класичних сценаріїв є те, що кожен шлюз обладнаний власним IDS-end - кінцевий вузол системи виявлення вторгнень), який інтегровано з OpenFlow Switch (комутатором із підтримкою протоколу OpenFlow). Завдяки такому поєднанню OpenFlow Switch не лише виконує базові функції маршрутизатора та фаєрвола, а й забезпечує аналіз усього трафіку через IDS-end. Це дає змогу виявляти

підозрілі пакети безпосередньо на рівні мережевого вузла, перш ніж вони потрапляють у внутрішню інфраструктуру клієнта.

Важливою особливістю цього підходу є те, що опис сервісу виконується у вигляді моделі на мові XML, як показано на в Додатку Б. У цій моделі фіксуються ключові параметри, які визначають поведінку системи. Зокрема, атрибут control_type дозволяє задати, хто саме відповідає за управління IPS: оператор (значення internal) чи клієнт через власний GC у режимі BYOC. У даному прикладі вибрано саме другий варіант, що демонструє можливість винесення функцій контролю за межі оператора. Параметр моделі distribution описує спосіб розгортання сервісу. Для досліджуваного кейсу обрано значення distributed, яке означає, що IDS-end встановлюються у кожному клієнтському сайті, створюючи тим самим розподілену систему виявлення та запобігання атак. Крім того, XML-модель включає специфікацію інтерфейсів interface_1, interface_2 та interface_3, що відповідають за VPN-з'єднання між сайтами, визначаючи IP-адреси, зони та інші параметри. Нарешті, у моделі відображено й окремий сервіс IPS, контроль над яким покладається саме на GC, що відображає ідеологію BYOC.

У внутрішній архітектура IDS-end, яка розгортається на кожному майданчику клієнта, створюються два віртуальні ресурси: IDS-end і OpenFlow Switch. Після ініціалізації обох елементів за допомогою SO, світч конфігурується так, щоб увесь вхідний трафік обов'язково проходив через IDS-end. Це означає, що IDS-end виступає не просто як пасивний сенсор, а як активний компонент шляху пересилання, здатний відстежувати, аналізувати та маркувати трафік, генеруючи повідомлення типу log і alert у випадку виявлення підозрілої активності. З огляду на інтеграцію з BYOC, ці повідомлення далі передаються у SO, який за допомогою своїх внутрішніх механізмів (Decision Engine та Service Dispatcher) пересилає їх у GC клієнта. Саме клієнт через власний SM, реалізований у GC, аналізує повідомлення та визначає подальші дії, наприклад, блокування певної IP-адреси.

3.4 Реалізація інтерфейсу взаємодії з IPS

Після завершення етапу розгортання сервісу наступним кроком є реалізація інтерфейсу керування для системи запобігання вторгненням. У цій архітектурі клієнтський GC, що містить модуль SM, підключається до інфраструктури оператора (SO) за допомогою протоколу XMPP. Кожен GC має власний ідентифікатор у цій системі, який позначається як JID (Jabber ID - унікальний ідентифікатор клієнта XMPP), завдяки чому досягається чітка ізоляція та автентифікація клієнтських контролерів. Цей механізм відкриття каналу взаємодії є критично важливим, оскільки саме він створює міст між мережею оператора й користувацькими засобами управління безпекою, що відповідає принципам BYOC і дозволяє делегувати частину функцій контролю без компрометації цілісності інфраструктури.

Основними джерелами інформації для SM у GC є повідомлення, які формуються кінцевими вузлами виявлення атак IDS-end. Кожен IDS-end виконує аналіз трафіку, що проходить через нього та у разі виявлення підозрілої активності генерує спеціальні службові повідомлення. У даній архітектурі передбачено два типи таких повідомлень: log_ message, які відповідають за передачу інформації до журналу подій, та alert_ message, що сигналізують про критичні загрози та потребу негайного реагування. Для забезпечення сумісності з механізмами SDN та уніфікації обміну даними ці повідомлення кодуються у вигляді PacketIn (OpenFlow PacketIn message), які підтримуються стандартом OpenFlow. Таким чином, усі події безпеки від IDS-end доставляються у SO у єдиному форматі.

Щоб забезпечити можливість розрізнення типів повідомлень, у полі IPProto цих PacketIn використовується спеціальна маркувальна схема. Значенням 220 позначаються повідомлення типу log_message, тоді як значенням 230 ідентифікуються повідомлення alert_message. Такий підхід дозволяє оператору не змішувати різні класи повідомлень та створює можливість їхньої подальшої селекції для різних GC. При цьому SO виконує роль центрального вузла

маршрутизації службових повідомлень, що надходять від IDS-end, і визначає кінцевого одержувача завдяки використанню спеціальної бази правил.

Ця база відома як Decision Base (DB) і представлена на рисунку 3.4.

№	DataPathID	InPort	Src	Dst	TCP	IPProto	Action
1	00:00:00:00:AA:01	5	Any	Any	Any	220	SM1
2	00:00:00:00:AA:01	5	Any	Any	Any	230	SM1
3	00:00:00:00:BB:02	8	Any	Any	Any	220	SM2
4	00:00:00:00:BB:02	8	Any	Any	Any	230	SM2

Рисунок 3.4 – Decision Base для розподілу повідомлень IDS-end

У DB зберігаються правила, які визначають, до якого саме GC потрібно передати певний клас повідомлень, що надійшли від IDS-end. Таким чином, SO, проаналізувавши значення поля IPProto та інші атрибути PacketIn (наприклад, Data Path ID або InPort), може однозначно ідентифікувати GC, до якого адресоване повідомлення. Це дозволяє забезпечити коректний розподіл інформаційних потоків навіть у тому випадку, якщо в системі одночасно функціонує кілька GC, що відповідають за різні домени безпеки.

Сам процес прийняття рішень щодо маршрутизації повідомлень реалізується у спеціальному модулі DE. Саме DE є ключовим компонентом SO, який дозволяє інтегрувати BYOC-підхід у традиційну архітектуру SDN. На вході DE отримує PacketIn-повідомлення від IDS-end, звіряє їх із правилами DB і на цій основі визначає, чи слід обробити повідомлення локально у SO (наприклад, у разі технічного журналювання), чи переслати його до певного GC для подальшого аналізу й ухвалення рішень. Логіка DE дозволяє також відслідковувати унікальні профілі потоків, що надходять з IDS-end, і, використовуючи Decision Base, забезпечує прозоре делегування контрольних функцій клієнтові.

3.5 Механізми маршрутизації та доставки повідомлень від IDS-end

У межах архітектури BYOC важливим етапом після реалізації інтерфейсу керування IPS є організація механізмів маршрутизації та доставки повідомлень від IDS-end до відповідних гостьових контролерів. Для цього в системі передбачено два компоненти - DB та SD, які працюють у тісній взаємодії, забезпечуючи як правильний вибір адресата повідомлення, так і його гарантовану доставку. DB виконує функцію таблиці правил, подібної до тієї, що використовується у фаєрволах, однак замість рішень на рівні доступу до портів чи IP-адрес у ній зберігається відповідність між спеціально визначеними значеннями поля IPProto та ідентифікатором гостьового контролера. Як показано на рисунку 3.4, значення IPProto використовуються для кодування типів повідомлень: наприклад, 220 для log_message і 230 для alert_message. Завдяки цьому SO може відрізнити повідомлення різних категорій та спрямувати їх саме до того GC, який відповідає за їхню обробку. Кожен запис у DB фактично є правилом маршрутизації службового трафіку безпеки, і подібно до фаєрвола, ці правила можуть бути гнучко налаштовані оператором залежно від кількості клієнтів, кількості підключених IDS-end і вимог до сегментації потоків даних.

У свою чергу, після того як DB визначає, якому GC має бути передане повідомлення, в роботу вступає SD, що забезпечує його транспортування від SO до відповідного GC. Архітектурно SD виконує роль модуля комунікації, який підтримує кілька одночасних сесій і відповідає за ізоляцію потоків клієнтів. Для цього використовується протокол XMPP, що дає змогу будувати незалежні канали взаємодії для кожного клієнта. Як і в системах обміну повідомленнями, кожен GC ідентифікується у мережі за допомогою унікального ідентифікатора JID, який виступає своєрідною адресою доставки службових повідомлень. Це рішення дозволяє гарантувати, що навіть у випадку, коли в системі функціонує велика кількість клієнтів, повідомлення з IDS-end одного з них ніколи не потраплять до іншого, адже ізоляція забезпечується як на рівні DB, так і на рівні транспортного протоколу SD.

Як результат, взаємодія DB і SD утворює повноцінний механізм делегування функцій контролю клієнтові. DB визначає, до якого GC слід передати повідомлення на основі атрибутів PacketIn, а SD виконує фізичну доставку цього повідомлення у зашифрованому та ізольованому каналі XMPP. Така архітектура дозволяє клієнтові безпосередньо отримувати повідомлення log чи alert від IDS-end, обробляти їх у власному Security Manager, а далі формувати рішення щодо блокування або перенаправлення трафіку. У свою чергу, SO залишається центральним оператором інфраструктури, який контролює коректність роботи каналів і застосовує зміни у мережевих пристроях. Таким чином, DB і SD виступають двома критично важливими складовими архітектури, що забезпечують практичну реалізацію BYOC і створюють умови для побудови безпечного та масштабованого VPN-сервісу з інтегрованим IPS.

3.6 Розподілений контроль IPS

У межах концепції BYOC одним із найбільш перспективних напрямів розвитку є реалізація розподіленого контролю системи запобігання вторгненням. Такий підхід дозволяє не обмежувати весь процес аналізу та прийняття рішень одним єдиним SM, а ділити функції між кількома гостьовими контролерами (GC), кожен з яких може спеціалізуватися на певному аспекті обробки трафіку або працювати у власному домені мережі. Як наслідок, система IPS стає масштабованою, стійкішою до збоїв та більш адаптивною до потреб різних клієнтів. Одним із прикладів є використання спільної бази сигнатур атак, так званої Attack Signature Database, яка синхронізується між усіма GC і гарантує, що незалежно від того, який саме GC обробляє повідомлення від IDS-end, у нього завжди є актуальна інформація про відомі атаки. При цьому кожен GC може виконувати власний аналіз отриманих log або alert повідомлень, що надходять від SO через механізм SD, і приймати локальні рішення відповідно до своєї спеціалізації.

У такій архітектурі кожен GC має власний набір політик і алгоритмів для аналізу трафіку, однак завдяки єдиній базі сигнатур досягається узгодженість у

виявленні загроз. Наприклад, один GC може відповідати за первинну фільтрацію повідомлень log_message, тоді як інший концентрується на обробці критичних alert_message. Це дає змогу розподіляти навантаження між кількома центрами обробки й уникати ситуацій, коли єдиний SM стає «вузьким місцем» системи. Крім того, розподілений контроль дозволяє залучати декілька організаційних підрозділів або навіть зовнішніх постачальників послуг безпеки до спільної роботи над одним сервісом, при цьому кожен GC керує своєю частиною IPS.

Архітектурно така модель представлена на рисунку 3.5, де показано розподіл функцій IPS між кількома контролерами.

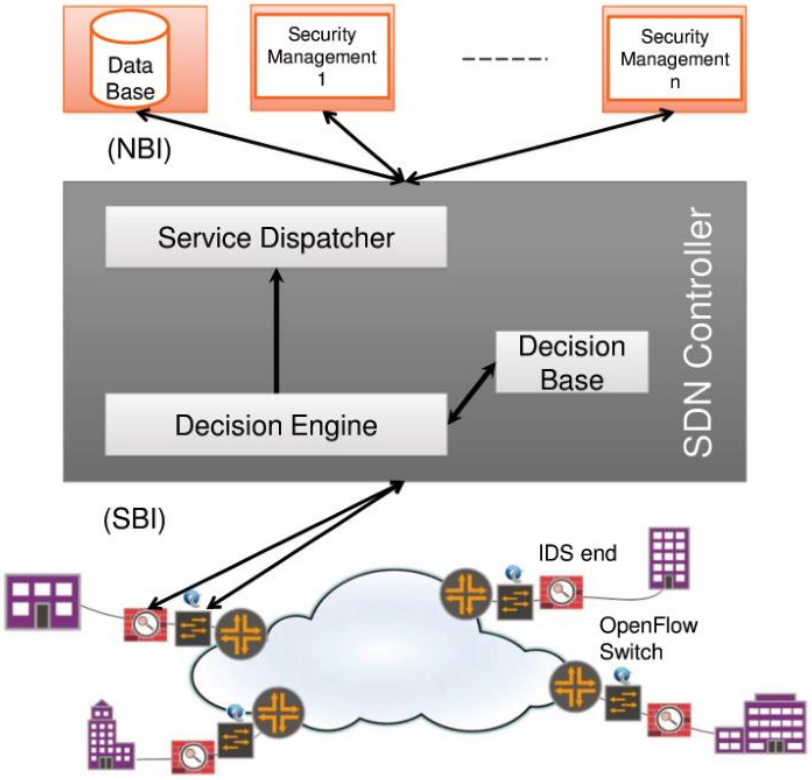

Рисунок 3.5 – Модель розподіленого контролю IPS

На цьому рисунку видно, що IPS-engine та база сигнатур можуть бути логічно відокремлені від конкретного GC, а взаємодія організована через SO, який зберігає роль центрального координатора. Саме SO, використовуючи Decision Engine та Decision Base, визначає, яке повідомлення слід передати якому GC, а далі Service Dispatcher забезпечує їх доставку за допомогою XMPP та JID-ідентифікаторів. Завдяки цьому можлива одночасна робота кількох незалежних

GC, які можуть навіть належати різним організаціям, але використовують спільну Attack Signature Database.

Такий розподілений контроль створює новий рівень маштадованості, оскільки дозволяє обирати оптимальну стратегію розподілу навантаження. Наприклад, у випадку атаки типу DDoS, одразу кілька GC можуть брати участь у прийнятті рішень, координуючи між собою кроки з блокування чи перенаправлення трафіку. Це особливо актуально для мультидоменних середовищ, де різні GC відповідають за різні сегменти мережі клієнта. При цьому клієнт отримує більше контролю над безпекою власної інфраструктури, зберігаючи водночас централізовану підтримку з боку SO, який гарантує цілісність та стабільність роботи всієї мережі.

3.7 Типи запитів за протоколом XMPP

У запропонованій архітектурі BYOC для організації взаємодії між SO та GC використовується XMPP, що забезпечує обмін повідомленнями у режимі publish/subscribe та підтримує унікальну ідентифікацію клієнтів за допомогою JID. На відміну від REST, де комунікація базується на HTTP-запитах, XMPP працює на основі потоків XML і дозволяє формувати як синхронні, так і асинхронні повідомлення. Це особливо важливо у випадку BYOC, де повідомлення можуть мати різну природу: журнальні записи (log), тривожні сигнали (alert), а також службові пакети, пов’язані з оновленням політик безпеки чи передачею рішень від SM до SO.

Таблиця 1.1 узагальнює типи XMPP-запитів та їх використання у BYOC-архітектурі для керування IPS.

Таблиця 1.1 – Типи XMPP-запитів

Тип XMPP-запиту	Приклад	Використання у BYOC-IPS
`<message type="chat"/>`	`log_message: Suspicious traffic on port 80`	Передача подій від IDS-end через SO до конкретного GC (*лог або тривога*).
`<iq type="get"/>`	`<query xmlns="byoc:ips:status"/>`	GC запитує стан IPS (активні IDS-end, правила у DB).
`<iq type="set"/>`	`<block ip="192.168.2.50" action="drop"/>`	GC надсилає рішення SO для оновлення політики (наприклад, блокування IP).
`<presence/>`	`<presence><status>online </status></presence>`	GC повідомляє про свою доступність у системі; SO реєструє активність клієнта.
`<message type="groupchat"/>`	`alert_message: Possible DDoS detected`	Широкомовна розсилка тривог до кількох GC у сценарії розподіленого IPS

Протокол XMPP у контексті архітектури BYOC підтримує кілька базових типів запитів, які дозволяють організувати повноцінну взаємодію між SO та GC. Одним із найважливіших механізмів є елемент `<message/>`, який використовується для однонаправленої передачі повідомлень. Наприклад, коли IDS-end генерує PacketIn зі значенням IPProto=220, SO визначає у базі рішень DB, що цей пакет належить GC1, і формує XMPP-повідомлення у форматі `<message from="so@domain" to="gc1@domain" type="chat"><body>log_message: Suspicious traffic detected on port 80</body></message>`. Це відповідає логіці пересилання log-повідомлень і демонструє, як події журналу можуть бути доставлені клієнту.

Інший тип запиту `<iq type="get"/>`, який використовується тоді, коли GC потребує отримати інформацію про стан сервісу IPS. GC може сформувати запит у вигляді `<iq from="gc1@domain" to="so@domain" type="get" id="1234"><query xmlns="byoc:ips:status"/></iq>`, після чого SO поверне йому список активних IDS-end і поточні правила, що зберігаються у DB. Такий механізм дозволяє клієнтові отримувати актуальний стан системи безпеки у будь-який момент.

Коли ж потрібно внести зміни до політики безпеки, використовується тип `<iq type="set"/>`. У цьому випадку GC може передати SO команду, наприклад,

для блокування IP-адреси. Запит виглядатиме так: `<iq from="gc1@domain" to="so@domain" type="set" id="5678"><command xmlns="byoc:ips:control"><block ip="192.168.2.50" action="drop"/></command></iq>`. Це відповідає етапу застосування рішень GC у SO, що показано на архітектурній схемі внутрішнього розгортання IDS-end.

Не менш важливим є й використання елементу `<presence/>`, який сигналізує про наявність або відсутність GC у мережі. Наприклад, при активації контролера GC може надіслати повідомлення `<presence from="gc1@domain" to="so@domain"><status>online</status></presence>`. Таким чином SO знає, що GC доступний і готовий до обробки вхідних подій, а також може підтримувати синхронізацію станів у реальному часі.

Протокол XMPP дозволяє використовувати багатоточкові повідомлення у форматі `<message/>` з атрибутом groupchat. Це необхідно у випадках, коли одне повідомлення має бути доставлене кільком GC, наприклад у розподіленому середовищі IPS. Прикладом такого повідомлення є `<message from="so@domain" to="ips_group@conference.domain" type="groupchat"><body>alert_message: Possible DDoS detected, apply mitigation</body></message>`. У цьому випадку SO надсилає сигнал тривоги одразу кільком контролерам, що дозволяє координувати їхні дії під час реагування на загрозу. Саме цей підхід демонструється у моделі розподіленого IPS-контролю, поданій на рисунку 3.5.

Таким чином, XMPP у моделі BYOC виступає універсальним транспортним протоколом, що одночасно забезпечує доставку подій безпеки, обмін службовими даними, оновлення політик та підтримку присутності клієнтських контролерів.

3.8 SDN контролер OpenDaylight

Серед різних SDN-контролерів, які можуть застосовуватися для побудови сервісів із підтримкою моделі BYOC, одним із найперспективніших є OpenDaylight (ODL) - відкритий SDN-контролер, що розвивається під егідою Linux Foundation. Його ключова перевага полягає у модульній архітектурі на

основі OSGi та підтримці широкого спектра південних інтерфейсів (Southbound Interface, SBI), зокрема OpenFlow, NETCONF та BGP-LS (див. рисунок 3.6).

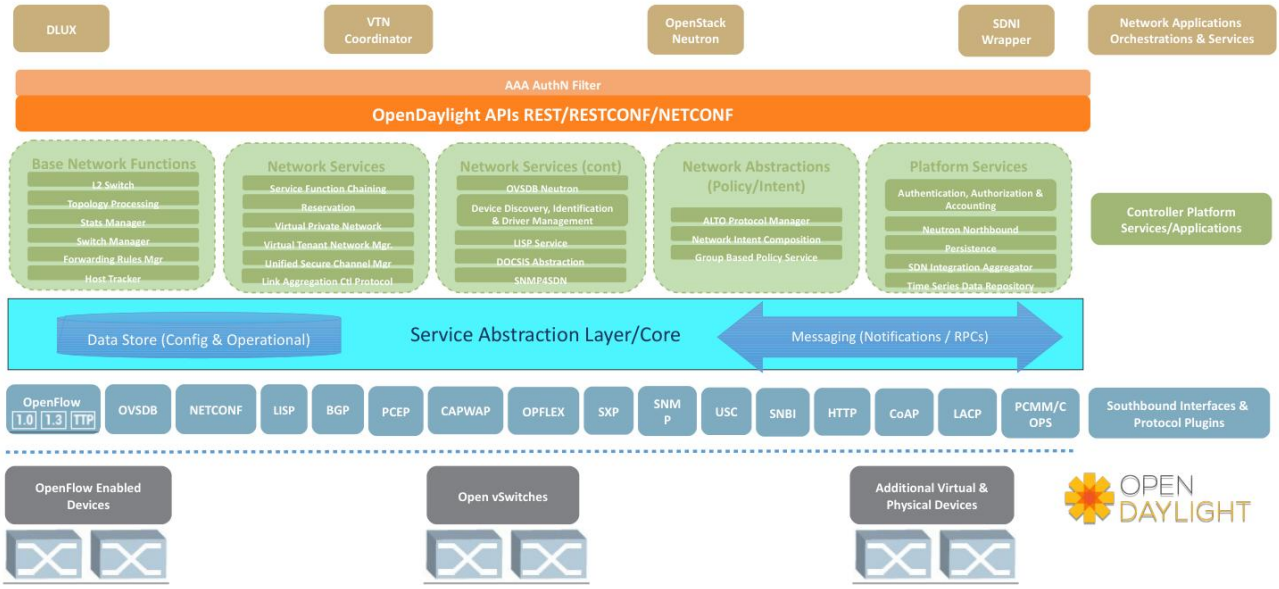

Рисунок 3.6 – Архітектура OpenDaylight SDN контролера

У базовій конфігурації OpenDaylight орієнтований на централізоване управління мережею через Northbound API, яке зазвичай реалізується на основі RESTCONF. Однак для реалізації BYOC, де клієнт отримує можливість безпосереднього контролю над певними функціями безпеки, необхідним є впровадження додаткового протоколу, який забезпечує двосторонній асинхронний обмін повідомленнями між клієнтським GC і SO.

У цьому контексті протокол XMPP виявляється доцільним рішенням, оскільки він дозволяє будувати publish/subscribe-механізми та підтримує унікальну ідентифікацію клієнтів через JID. Інтеграція XMPP у OpenDaylight можлива шляхом розробки додаткового модуля, що функціонує як розширення NBI. У цьому випадку OpenDaylight зберігає свої традиційні RESTCONF API для адміністратора та сервісного оператора, але водночас відкриває XMPP-канал для зовнішніх клієнтів. Такий канал дозволяє GC отримувати повідомлення log і alert, які надходять від IDS-end у форматі PacketIn, та надсилати власні рішення у вигляді команд блокування або перенаправлення трафіку.

Архітектурно це виглядає наступним чином. OpenDaylight виконує роль SO, що координує інфраструктуру через OpenFlow, водночас інтегрований XMPP-

модуль забезпечує комунікацію із GC. Повідомлення, які отримує SO від IDS-end, аналізуються через DB, а потім пересилаються через XMPP до відповідного GC. Далі SD у межах OpenDaylight розподіляє ці повідомлення між кількома GC, використовуючи їхні JID для ізоляції потоків. Такий підхід дозволяє досягти ситуації, коли один і той самий OpenDaylight-контролер обслуговує різні клієнтські домени, але логіка обробки інцидентів безпеки винесена на бік клієнта.

Перевагою використання OpenDaylight як основи є підтримка стандартів ONF і широка спільнота розробників, що дозволяє легко масштабувати рішення і адаптувати його до різних сценаріїв. Зокрема, у випадку розподіленого IPS-контролю, OpenDaylight може виступати в ролі центрального оператора, який зберігає глобальну консистентність системи, тоді як XMPP-розширення створює канали для незалежних GC, що обробляють log та alert повідомлення. Це робить архітектуру гнучкою, модульною та сумісною з вимогами BYOC.

Таким чином, інтеграція OpenDaylight з XMPP відкриває можливість практичної реалізації концепції BYOC у сервісах кібербезпеки. Вона дозволяє поєднати переваги централізованого контролю SDN із гнучкістю делегування функцій безпеки клієнту, забезпечуючи масштабованість, ізоляцію та ефективну реакцію на загрози у багатодоменних VPN-середовищах.

3.9 Висновки до розділу 3

В третьому розділі було обґрунтовано та детально описано архітектуру і реалізацію BYOC-сервісу IPS у середовищі SDN, показано, як частина контрольних функцій безпеки переноситься від оператора до клієнта через його Guest Controller. Продемонстровано, що поєднання IDS-end як сенсорів трафіку, клієнтського Security Manager у складі GC та інфраструктурних можливостей оператора (OpenFlow-комутатори, керовані через SBI) перетворює традиційну модель виявлення на повноцінний сервіс запобігання вторгненням, у якому функції виявлення, аналізу й блокування чітко розмежовані та керуються узгоджено.

Сформовано принципи розширення класичної IPS засобами SDN. Service Operator слугує посередником між площиною даних і клієнтською площиною контролю, відкриваючи NBI та зберігаючи централізоване застосування рішень у мережі. Запропоновано механізм взаємодії SO та GC на основі XMPP з JID-ідентифікацією, а також внутрішніх модулів Decision Base і Service Dispatcher у зв’язці з Decision Engine. Показано, що така побудова дає змогу клієнтові оперативно впливати на політику безпеки у власному домені, не порушуючи цілісності глобальної інфраструктури.

Окремо продемонстровано практичну життєздатність підходу на прикладі захищеного VPN, де кожен сайт обладнано IDS-end та OpenFlow-комутатором, а опис сервісу уніфіковано в XML-моделі з параметрами control_type і distribution. Показано типи XMPP-запитів, що підтримують як асинхронні події, так і синхронні запити. Показано можливість інтеграції з промисловим SDN-контролером OpenDaylight: RESTCONF лишається для адміністративних задач, тоді як XMPP-розширення забезпечує клієнтський канал керування безпекою, зберігаючи сумісність зі стандартними SBI (OpenFlow, NETCONF, BGP-LS).

РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

Метою кваліфікаційної роботи є розробка та дослідження моделі надання мережевих сервісів у SDN-мережі з підтримкою концепції BYOC, орієнтованої на підвищення рівня кібербезпеки. Оскільки, такі роботи передбачають використання комп’ютерної техніки, зокрема ПК та серверного обладнання, то обов’язком виконавця такого процесу є дотримання вимог охорони праці і техніки безпеки.

Роботодавець зобов'язаний згідно Закону України «Про охорону праці» стаття 13 «Управління охороною праці та обов’язки роботодавця» створити на робочому місці в кожному структурному підрозділі умови праці відповідно до нормативно-правових актів, а також забезпечити додержання вимог законодавства щодо прав працівників у галузі охорони праці [21].

Із цією метою роботодавець забезпечує функціонування системи управління охороною праці, а саме:

- створює відповідні служби і призначає посадових осіб, які забезпечують вирішення конкретних питань охорони праці, затверджує інструкції про їхні обов'язки, права та відповідальність за виконання покладених на них функцій, а також контролює їх додержання;

- розробляє за участю сторін колективного договору і реалізує комплексні заходи для досягнення встановлених нормативів та підвищення існуючого рівня охорони праці;

- забезпечує виконання необхідних профілактичних заходів відповідно до обставин, що змінюються;

- впроваджує прогресивні технології, досягнення науки і техніки, засоби механізації та автоматизації виробництва, вимоги ергономіки, позитивний досвід з охорони праці тощо;

- забезпечує належне утримання будівель та споруд, виробничого обладнання та устаткування, моніторинг за їх технічним станом;

- забезпечує усунення причин, що призводять до нещасних випадків, професійних захворювань, та здійснення профілактичних заходів, визначених комісіями за підсумками розслідування цих причин;

- організовує проведення аудиту охорони праці, лабораторних досліджень умов праці, оцінку технічного стану виробничого обладнання та устаткування, атестацій робочих місць на відповідність нормативно-правовим актам з охорони праці в порядку і строки, що визначаються законодавством, та за їх підсумками вживає заходів з усунення небезпечних і шкідливих для здоров'я виробничих факторів;

- розробляє і затверджує положення, інструкції, інші акти з охорони праці, що діють у межах підприємства та встановлюють правила виконання робіт і поведінки працівників на території підприємства, у виробничих приміщеннях, на будівельних майданчиках, робочих місцях відповідно до нормативно-правових актів з охорони праці, забезпечує безоплатно працівників нормативно-правовими актами підприємства з охорони праці;

- здійснює контроль за дотриманням працівником технологічних процесів, правил поведінки з машинами, механізмами, устаткуванням та іншими засобами виробництва, використанням засобів колективного та індивідуального захисту, виконанням робіт відповідно до вимог з охорони праці;

- організовує пропаганду безпечних методів праці та співробітництво з працівниками у галузі охорони праці.

Роботодавець несе безпосередню відповідальність за порушення нормативно-правових актів з охорони праці. Для забезпечення оптимальних умов праці працівників при розробці, встановлення та налаштуванні систем високої доступності на основі кластерів необхідно передбачити відповідність мікроклімату у приміщеннях згідно вимог ДСН 3.3.6.042-99. Категорія робіт при виконанні даного виду завдань належить до легкої – Іб. Для того щоб визначити, чи відповідає повітря приміщення встановленим нормам, необхідно кількісно оцінити кожний з його параметрів. Оптимальні показники мікроклімату, які

необхідно забезпечити у приміщеннях, де експлуатуються ПК у теплу пору року повинні становити: температура – +22 оС - +24 оС, відносна вологість – 40-60%, швидкість руху повітря 0,1 м/с. Окрім, забезпечення оптимальних показників мікроклімату, необхідно передбачити ще й оптимальні показники шуму та вібрації на робочих місцях. Граничні величини шуму на робочих місцях регламентуються ДСН 3.3.6.037 – 99 „Державні санітарні норми виробничого шуму, ультразвуку та інфразвуку”. В ньому закладено принцип встановлення певних параметрів шуму, виходячи з класифікації приміщень та їх використання для трудової діяльності. Окрім цього, на робочих місцях працівників необхідно забезпечити дотримання вимог НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями».

Під час організації робочих місць операторів необхідно забезпечити достатні просторові та безпекові умови для виконання професійних обов’язків. Приміщення повинні мати достатню площу й об’єм, щоб працівники могли вільно розміщувати обладнання, змінювати робочу позу та безпечно пересуватися в межах робочої зони без ризику травмування чи перевантаження.

Окрім цього, приміщення, у яких постійно перебувають оператори, повинні бути оснащені засобами раннього виявлення пожежі, зокрема системою автоматичної пожежної сигналізації. Дотримання зазначених вимог є необхідною умовою під час створення та експлуатації систем високої доступності на основі кластерних рішень із використанням серверного обладнання та персональних комп’ютерів, оскільки дозволяє не лише знизити ризики для здоров’я працівників, а й підвищити надійність та безперервність функціонування інформаційної інфраструктури в цілому.

4.2 Джерела, зони дії та рівні забруднення навколишнього середовища у разі аварій на хімічно і радіаційно небезпечних об’єктах

Джерела та причини аварій на хімічно та радіаційно небезпечних об'єктах різноманітні та часто пов'язані з комплексом чинників, включаючи людський

фактор, технічні несправності, недостатність безпекових заходів та навіть природні катастрофи. Розглянемо приклади, які демонструють ці аварії.

Аварія на Чорнобильській АЕС (1986 рік) – одна з найбільш катастрофічних ядерних аварій у світовій історії. Ця аварія сталася через людські помилки та недоліків у справності реактора. Під час проведення експерименту з безпеки, який включав вимкнення деяких систем безпеки, стався вибух, що призвів до викиду радіаційних матеріалів у атмосферу [22].

Витік метилізоціанату в Бхопалі, Індія (1984 рік) – цей інцидент став однією з найбільших хімічних катастроф у світі. Витік стався внаслідок води, яка потрапила у резервуар з метилізоціанатом, що призвело до хімічної реакції та викиду токсичного газу. Причиною інциденту стали недоліки в управлінні безпекою та технічному обслуговуванні на заводі [23].

Аварія на АЕС Фукусіма-Даїчі (2011 рік) – приклад аварії, викликаної природним катаклізмом. Масштабне землетрус та наступний цунамі призвели до втрати зовнішнього живлення та відмови систем охолодження реакторів, що спричинило серйозні пошкодження та радіоактивні викиди [24].

Вибух на хімічному заводі в Тяньцзіні, Китай (2015 рік) – стався внаслідок неправильного зберігання великої кількості небезпечних хімічних речовин. Вибух спричинив значні руйнування та забруднення навколишнього середовища [25].

При аваріях на атомних електростанціях, таких як Чорнобиль або Фукусіма, радіаційне забруднення може поширюватися на сотні та навіть тисячі кілометрів від місця інциденту. Радіоактивні частинки, підняті у повітря, можуть бути рознесені вітром на великі відстані, що призводить до широкомасштабного забруднення атмосфери, ґрунтів, водойм та живих організмів. Характер такого розповсюдження залежить від багатьох факторів, включаючи інтенсивність викидів, погодні умови та топографію місцевості [26].

Випадок хімічного забруднення, як у Бхопалі, Індії, показує, як витік токсичних речовин може мати негайні та руйнівні наслідки для навколишнього середовища та здоров'я людей. В таких випадках, забруднювачі можуть швидко розповсюджуватися повітрям та водою, впливаючи на великі території. Хімічні

речовини можуть контамінувати ґрунт, підземні та поверхневі води, що призводить до довгострокових екологічних наслідків. Зона дії забруднення також залежить від швидкості реагування на аварію. Ефективні заходи реагування, такі як ізоляція вогнища аварії, можуть обмежити розповсюдження забруднювачів. Проте, в багатьох випадках, особливо при великих аваріях або недостатньому реагуванні, забруднення може поширитися на значні території, ускладнюючи процес відновлення та очищення.

Рівні забруднення на небезпечних об'єктах класифікуються за серйозністю впливу на довкілля та здоров'я людей. Ця класифікація допомагає у визначенні необхідних заходів реагування та стратегій відновлення.

Низький рівень забруднення:

– характеризується мінімальними концентраціями забруднюючих речовин, які не перевищують встановлені норми безпеки;

– вплив на довкілля та здоров'я людей є мінімальним або відсутнім;

– зазвичай не вимагає спеціальних заходів реагування, але потребує моніторингу для виявлення можливих змін.

Середній рівень забруднення:

– рівень забруднення перевищує норми безпеки, але не досягає критично небезпечного рівня;

– може мати короткотермінові негативні наслідки для довкілля, такі як зниження якості води, забруднення ґрунтів, шкоду для диких тварин;

– для людей можуть виникнути тимчасові проблеми зі здоров'ям, особливо в разі прямого контакту з забрудненими речовинами;

– вимагає заходів по локалізації забруднення та відновленню забруднених територій.

Високий рівень забруднення:

– характеризується значним перевищенням норм безпеки;

– має серйозні негативні наслідки для довкілля, такі як масова загибель тварин та рослин, забруднення водойм, руйнування екосистем;

– для людей високий ризик розвитку серйозних захворювань, включаючи отруєння, хронічні захворювання, збільшення випадків раку в разі радіаційного забруднення;

– потребує негайних заходів по евакуації населення, локалізації забруднення та тривалого відновлення екосистем.

Критичний рівень забруднення:

– найбільш небезпечний рівень, що включає екстремально високі концентрації токсичних або радіоактивних речовин;

– зона забруднення може бути оголошена непридатною для життя на тривалий час;

– довкілля може зазнати незворотних змін, з надзвичайно довгим періодом відновлення;

– для людей критичний рівень забруднення несе безпосередню загрозу життю та здоров'ю;

– вимагає довгострокових стратегій очищення та реабілітації, а також масштабних екологічних та охоронних проектів [27].

У даному розділі було проведено аналіз факторів ризику, зон впливу та рівнів забруднення, які виникають у випадку аварій на хімічно і радіаційно небезпечних об’єктах. Розглянуто різні сценарії аварій, які включають як людські помилки, так і природні катастрофи, та їх вплив на довкілля та здоров’я людей. Забруднення, викликане такими аваріями, може масштабні наслідки, поширюючись на значні території та призводячи до серйозних екологічних та проблем зі здоров’ям.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи магістра було спроєктовано архітектуру сервісу мережевої безпеки за моделлю BYOC у середовищі SDN із інтегрованою IPS. Результатом стала узгоджена багаторівнева схема, у якій функції виявлення, аналізу та блокування загроз розмежовані між кінцевими модулями IDS, клієнтським контролером та оператором сервісу з підтримкою протоколу OpenFlow на південному інтерфейсі і протоколу XMPP на північному інтерфейсі.

У першому розділі було здійснено огляд предметної області, сформульовано проблематику централізованих моделей управління безпекою та обґрунтовано доцільність переходу до моделі BYOC. Проаналізовано класичні архітектури систем виявлення та запобігання вторгненням, визначено вимоги до сервісу у середовищі VPN, сформовано критерії оцінювання масштабованості й стійкості.

У другому розділі було описано концептуальну модель і механізми інтеграції. Детально розглянуто відокремлення площини керування від площини даних у SDN, визначено роль оператора сервісу як посередника, охарактеризовано північний і південний інтерфейси, а також логіку взаємодії клієнтського контролера та оператора сервісу через протокол XMPP. Подано модель селекції подій через базу рішень та їх маршрутизації диспетчером сервісів, визначено формат подій і маркування типів повідомлень для подальшої адресації до клієнтського контролера.

У третьому розділі було реалізовано й досліджено сервіс запобігання вторгненням за моделлю BYOC у сценарії віртуальної приватної мережі. Представлено фізичну топологію з кінцевими модулями IDS та комутаторами з підтримкою протоколу OpenFlow на клієнтських сайтах, розроблено модель сервісу у форматі XML з параметрами control_type та distribution, описано механізм взаємодії Decision Engine → Decision Base → Service Dispatcher для доставки журналів та тривожних повідомлень до клієнтського контролера та зворотного застосування рішень у мережевій інфраструктурі. Наведено

результати тестування функціональності, безпеки та доступності, виконано порівняння з централізованою моделлю, продемонстровано можливості розподіленого контролю із використанням кількох клієнтських контролерів та спільної бази сигнатур атак, а також описано інтеграцію з контролером OpenDaylight, розширеним протоколом XMPP.

У підсумку показано, що архітектура BYOC забезпечує вищу гнучкість і швидкість реакції клієнта без втрати керованості інфраструктури оператором сервісу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Software Defined Networking (SDN). URL: https://learn.microsoft.com/uk-ua/windows-server/networking/sdn/ (date of access: 04.12.2025).
2. Software-Defined Networking - Architecture. Free Tutorials on Technical and Non Technical Subjects. URL: https://www.tutorialspoint.com/software-defined-networking/software-defined-networking-architecture.htm (date of access: 04.12.2025).
3. OpenFlow™ Protocol Feature Overview and Configuration Guide. Allied Telesis. URL: https://www.alliedtelesis.com/us/en/documents/openflow-feature-overview-and-configuration-guide (date of access: 04.12.2025).
4. OpenDaylight. URL: https://www.opendaylight.org/ (date of access: 04.12.2025). What is an intrusion prevention system (IPS)? | fortinet. Fortinet. URL: https://www.fortinet.com/resources/cyberglossary/what-is-an-ips (date of access: 04.11.2024).
5. Ryu SDN Framework. URL: https://ryu-sdn.org/ (date of access: 04.12.2025).
6. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56. https://doi.org/10.31891/2219-9365-2024-79-7
7. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220. https://doi.org/10.31891/2219-9365-2024-80-26
8. Open Network Operating System (ONOS) SDN Controller for SDN/NFV Solutions. Open Networking Foundation. URL: https://opennetworking.org/onos/ (date of access: 04.12.2025).
9. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.

10. Tymoshchuk, V., Mykhailovskyi, O., Dolinskyi, A., Orlovska, A., & Tymoshchuk, D. (2024). OPTIMISING IPS RULES FOR EFFECTIVE DETECTION OF MULTI-VECTOR DDOS ATTACKS. Матеріали конференцій МЦНД, (22.11. 2024; Біла Церква, Україна), 295-300.
11. Tymoshchuk, V., Vantsa, V., Karnaukhov, A., Orlovska, A., & Tymoshchuk, D. (2024). COMPARATIVE ANALYSIS OF INTRUSION DETECTION APPROACHES BASED ON SIGNATURES AND ANOMALIES. Матеріали конференцій МЦНД, (29.11. 2024; Житомир, Україна), 328-332.
12. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.
13. What is an intrusion prevention system (IPS)? | fortinet. (n.d.). Fortinet. https://www.fortinet.com/resources/cyberglossary/what-is-an-ips
14. What is intrusion detection systems (IDS)? How does it work? | fortinet. (n.d.). Fortinet. https://www.fortinet.com/resources/cyberglossary/intrusion-detection-system
15. Tymoshchuk, V., Karnaukhov, A., & Tymoshchuk, D. (2024). USING VPN TECHNOLOGY TO CREATE SECURE CORPORATE NETWORKS. Collection of scientific papers «ΛΌГОΣ», (June 21, 2024; Seoul, South Korea), 166-170.
16. Goodwin M. What Is an SLA (service level agreement)? | IBM. IBM. URL: https://www.ibm.com/think/topics/service-level-agreement (date of access: 04.12.2025).
17. IBM. (n.d.). What Is a REST API (RESTful API)? | IBM. https://www.ibm.com/think/topics/rest-apis
18. XMPP - The universal messaging standard. (n.d.). XMPP - The universal messaging standard | The universal messaging standard. https://xmpp.org/
19. The Jabber Project | XMPP - The universal messaging standard. (n.d.). XMPP - The universal messaging standard | The universal messaging standard. https://xmpp.org/about/the-jabber-project/

20. Tymoshchuk, V., Pakhoda, V., Dolinskyi, A., Karnaukhov, A., & Tymoshchuk, D. (2024). MODELLING CYBER THREATS AND EVALUATING THE PERFORMANCE OF INTRUSION DETECTION SYSTEMS. Grail of Science, (46), 636–641. https://doi.org/10.36074/grail-of-science.29.11.2024.081
21. Про охорону праці. (n.d.). Офіційний вебпортал парламенту України. https://zakon.rada.gov.ua/go/2694-12
22. Барановська Н. П. Соціальні та економічні наслідки Чорнобильської катастрофи. К. 2001. 20 с.
23. Bhopal 1984 Disaster: A Gas Leak Tragedy and its Effects on the People. URL: https://www.jetir.org/papers/JETIREW06090.pdf (дата звернення: 16.11.2025).
24. Fukushima disaster: What happened at the nuclear plant? - BBC News. URL: https://www.bbc.com/news/world-asia-56252695 (дата звернення: 16.11.2025).
25. Case study of the Tianjin accident: Application of barrier and systems analysis to understand challenges to industry loss prevention in emerging economies. URL: https://minerva.jrc.ec.europa.eu/en/shorturl/minerva/acceptedmanuscriptcase_ study _of_the_tianjin_accident_20190817pdf (дата звернення: 16.11.2025).
26. Nedzelky, D., Derkach, M., Skarga-Bandurova, I., Shumova, L., Safonova, S., & Kardashuk, V. (2021, September). A Load Factor and its Impact on the Performance of a Multicore System with Shared Memory. In 2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Vol. 1, pp. 499-503). IEEE.
27. Стручок В.С. Техноекологія та цивільна безпека. Частина «Цивільна безпека». Навчальний посібник. Тернопіль: ТНТУ. 2022. 150 с.
28. Exposure to outdoor air pollution and its human health outcomes: A scoping review. URL: https://journals.plos.org/plosone/article?id=10.1371/ journal.pone. 0216550 (дата звернення: 16.11.2025).

Додаток А Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний технічний університет імені Івана Пулюя
Маріборский університет (Словенія)
Технічний університет у Кошице (Словаччина)
Вільнюський технічний університет ім. Гедимінаса (Литва)
Краківський економічний університет (Польща)
Вроцлавський економічний університет (Польща)
Університет «Опольська Політехніка» (Польща)
Національний університет «Полтавська політехніка імені Юрія Кондратюка»
Вінницький національний аграрний університет
Львівський національний університет ім. І. Франка
Головне управління Пенсійного фонду в Тернопільській області
Наукове товариство ім. Шевченка
Тернопільський обласний комунальний інститут післядипломної педагогічної освіти
Сумський державний педагогічний університет
Запорізький національний університет

АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ

Збірник

тез доповідей

XIV Міжнародної науково-технічної конференції молодих учених та студентів

11-12 грудня 2025 року

УКРАЇНА
ТЕРНОПІЛЬ – 2025

УДК 004.056: 004.738.5
І. М. Чепіль, В.Д. Тимощук
(Тернопільський національний технічний університет імені Івана Пулюя)

МОДЕЛЬ НАДАННЯ МЕРЕЖЕВИХ СЕРВІСІВ У SDN З ПІДТРИМКОЮ BRING YOUR OWN CONTROL

I. Chepil, V. Tymoshchuk

A NETWORK SERVICE PROVISIONING MODEL IN SDN WITH BRING YOUR OWN CONTROL

Програмно-визначені мережі (SDN) дедалі частіше стикаються з обмеженнями традиційної моделі централізованого керування, яка ускладнює швидку адаптацію мережевих сервісів безпеки до динамічних загроз та індивідуальних потреб

користувачів [1]. У ситуаціях, коли політики безпеки повинні модифікуватися у реальному часі, а впровадження складних сервісів, таких як VPN чи IPS, залежить від рішень оператора, відсутність гнучкості та можливості клієнта впливати на процес керування призводить до затримок, зниження ефективності реагування та перевантаження площини керування.

У цьому контексті запропоновано застосування концепції Bring Your Own Control (BYOC), що забезпечує розподіл управлінських повноважень між оператором мережі та користувачем шляхом впровадження гостьового контролера (GC), який взаємодіє з контролером оператора (SO). Оператор зберігає контроль над інфраструктурою, глобальними політиками безпеки та ресурсами, тоді як клієнт через GC отримує можливість самостійно керувати сервісами безпеки у власному домені. Ключову роль відіграє північний інтерфейс (NBI), що реалізується як гібрид REST/XMPP [2]. REST використовується для синхронних операцій управління сервісом, тоді як XMPP забезпечує асинхронну доставку подій безпеки, публікацію та підписку на повідомлення IDS/IPS, а також ідентифікацію GC за допомогою ID.

Архітектура BYOC-сервісу кібербезпеки проілюстрована на прикладі захищеного VPN з інтегрованою IPS [3]. На кожному клієнтському майданчику розгортається IDS-end, інтегрований з OpenFlow-комутатором [4], який перенаправляє весь трафік через сенсор. Події безпеки формуються у вигляді OpenFlow PacketIn із маркуванням у полі IPProto для розрізнення log та alert-повідомлень. У Service Operator вони обробляються модулем Decision Engine, який, спираючись на базу рішень (Decision Base), визначає відповідний GC і передає події через Service Dispatcher за протоколом XMPP. У свою чергу Security Manager у складі GC аналізує події та генерує рішення щодо блокування, перенаправлення чи додаткового моніторингу трафіку, які надалі застосовуються до інфраструктури через SDN-контролер (OpenDaylight з OpenFlow-підтримкою [5]).

Розроблена модель демонструє можливість побудови двошарової системи керування, у якій поєднуються централізоване управління ресурсами з боку оператора та децентралізоване управління політиками безпеки з боку клієнта. Це скорочує час реакції на інциденти, підвищує адаптивність сервісів та спрощує інтеграцію сторонніх рішень кіберзахисту.

Література

1. Software Defined Networking (SDN). Microsoft Learn. URL: https://learn.microsoft.com/uk-ua/windows-server/networking/sdn/ (date of access: 30.11.2025).

2. Standards Process | XMPP - The universal messaging standard. URL: https://xmpp.org/about/standards-process/ (date of access: 30.11.2025).

3. Tymoshchuk, V., Vantsa, V., Karnaukhov, A., Orlovska, A., & Tymoshchuk, D. (2024). COMPARATIVE ANALYSIS OF INTRUSION DETECTION APPROACHES BASED ON SIGNATURES AND ANOMALIES. Матеріали конференцій МЦНД, (29.11. 2024; Житомир, Україна), 328-332. https://doi.org/10.62731/mcnd-29.11.2024.004

4. OpenFlow – protocol overview - EXATEL. URL: https://exatel.pl/en/knowledge/blog/articles/openflow-protocol-overview/ (date of access: 30.11.2025).

5. OpenDaylight. URL: https://www.opendaylight.org/ (date of access: 30.11.2025).

Додаток Б XML-опис розподіленого VPN-сервісу з IPS-захистом

```
<!-- Головний офіс (Main office / server room) -->
    <interface_1>
      <service_type>Connectivity</service_type>
      <name>Main_Office</name>
      <zone>Headquarters</zone>
      <configuration>
        <ip_address>192.168.1.1</ip_address>
        <netmask>255.255.255.0</netmask>
        <role>Server_Room</role>
      </configuration>
    </interface_1>

    <!-- Офіс 1 -->
    <interface_2>
      <service_type>Connectivity</service_type>
      <name>Office_1</name>
      <zone>Ternopil</zone>
      <configuration>
        <ip_address>192.168.2.1</ip_address>
        <netmask>255.255.255.0</netmask>
        <role>Branch_Office</role>
      </configuration>
    </interface_2>

    <!-- Офіс 2 -->
    <interface_3>
      <service_type>Connectivity</service_type>
      <name>Office_2</name>
      <zone>Lviv</zone>
      <configuration>
        <ip_address>192.168.3.1</ip_address>
        <netmask>255.255.255.0</netmask>
        <role>Branch_Office</role>
      </configuration>
    </interface_3>
  </service>

  <!-- Додатковий сервіс безпеки IPS -->
  <service>
    <name>IPS</name>
    <control_type>byoc</control_type>
    <distribution>distributed</distribution>
    <description>
      Intrusion Prevention System (IPS) розгортається у кожному офісі
      разом з OpenFlow Switch, що перенаправляє трафік через IDS-end
      для аналізу. Управління IPS делеговане Guest Controller (GC) клієнта.
    </description>
  </service>

</services>
```