Міністерство освіти і науки України
Західноукраїнський національний університет
Факультет комп’ютерних інформаційних технологій
Кафедра спеціалізованих комп’ютерних систем

ВОЗНЯК ВІКТОРІЯ СЕРГІЇВНА

АВТОМАТИЗОВАНА СИСТЕМА КЕРУВАННЯ ОБЛАДНАННЯМ КОМП’ЮТЕРНИХ МЕРЕЖ/ AUTOMATED COMPUTER NETWORK EQUIPMENT MANAGEMENT SYSTEM

спеціальність: 174 — Автоматизація, комп’ютерно-інтегровані технології та робототехніка

освітньо-професійна програма - Автоматизація та програма комп’ютерно-інтегровані технології

Магістерська робота

Виконав студент групи АКІТм-21
В.С. Возняк
_____
Науковий керівник:
к.т.н., О.М. Заставний
_____

Магістерську роботу допущено до захисту:
"____"_____2025р.
Завідувач кафедри
_____ А.І. Сегін

Тернопіль 2025

ЗМІСТ

ВСТУП

Актуальність теми. В сучасних умовах все більшої популярності набувають засоби, які допомагають керувати обладнанням комп'ютерних мереж. Особливу увагу приділяють автоматизованим системам, які не тільки виступають як помічники і інструменти, у роботі мережевих операторів, адміністраторів і інженерів. А такі системи подекуди заміняють рутинну роботу даних фахівців в автоматичному режимі.

На ринку є багато продуктів які виконують функції моніторингу, систематизації, збору журналів та формування різного роду статистики, але серед них мало продуктів, які можуть повноцінно керувати різними типами обладнання та продукцією різних вендорів.

Тому актуальним в наш час є створення власних систем і підсистем, які могли б робити повноцінне розкриття топологій мереж, їх періодичне сканування, збір різного роду даних, аналіз цих даних та журналів роботи пристроїв, керували конфігураціями, моніторили стани та у разі потреби відновлювали їх працездатність.

— **Мета кваліфікаційної роботи** полягає в розробці автоматизованої системи керування різноманітним обладнанням корпоративної комп'ютерної мережі (ККМ). Для досягнення мети потрібно виконати такі завдання:

— класифікувати мережеве обладнання по типу функціонування на різних рівнях взаємодії;

— здійснити актуальні пристрої комп'ютерних мереж провідних компаній світу;

— зробити аналіз способів контролю стану мережевих пристроїв;

— проаналізувати існуючу корпоративну комп'ютерну мережу та збої її мережевого обладнання;

— на основі досліджень спроектувати модуль моніторингу збоїв комп'ютерних мереж;

— побудувати функціональну схему програмного забезпечення.

Предметом дослідження є корпоративна комп'ютерна мережа ЗУНУ.

Об'єктом дослідження є автоматизована система керування обладнанням комп'ютернорї мережі.

Методи дослідження – аналіз факторів та причин, що впливають на надійність компютерних мереж. Дослідження систем моніторингу та керування мережевим обладнанням.

Наукова новизна. Запропоновано систему моніторингу, при якій відбувається сканування тільки деяких комутаторів у центрі мережі із кінцевих вузлів, що виявляє будь-які збої на всіх сегментах, не потребуючи сканування всіх наявних комутаторів.

Практичне значення отриманих результатів. Запропонована АС дозволить зменшити рутинну роботу мережевих спеціалістів по моніторингу та керуванню мережевим обладнанням ККМ.

Апробація. 1. Стафін В., Возняк В. Обгрунтування доцільності проектування та автоматизації абонентської мережі з використанням технології ADSL. // Матеріали проблемно-наукової міжгалузевої конференції АКІТ – 2022, Тернопіль. 21-23 лютого, 2022 – С.38-43.

2. Кореляційні моделі в полярній системі координат / А.І. Сегін, Ю.І. Попик, В.С. Возняк [та ін.] // Матеріали проблемно-наукової міжгалузевої конференції ISCM – 2023, 2023. – С.188-191.

3. N. Vozna, A. Dunets, V. Shevchuk, V. Vozniak, I. Mamuzić Automation of technological processes in the oil and gas industry. Metallurgy and related tropics: Proceedings of 18th symposium „Materials and metallurgy". SHMD´2025, Croatia (published in: Metalurgija 64 (2025) 1-2). – P. 233-242. https://hrcak.srce.hr/file/462264

1. ОБЛАДНАННЯ КОМПЮТЕРНИХ МЕРЕЖ ТА СПОСОБИ КЕРУВАННЯ НИМИ

1.1 Класифікація мережевого обладнання

Мережеве обладнання становить основу будь-якої комп'ютерної мережі, оскільки саме воно забезпечує фізичне та логічне з'єднання між пристроями, передавання даних, маршрутизацію трафіку, керування потоками інформації, а також безпеку під час обміну. Без правильно підібраного мережевого обладнання неможливо забезпечити стабільність, надійність і продуктивність мережевої інфраструктури. Тому питання класифікації мережевих пристроїв є одним із ключових під час побудови й дослідження сучасних комп'ютерних мереж.

Класифікація мережевого обладнання базується на кількох основних критеріях. Найчастіше застосовують поділ за рівнем функціонування в моделі OSI (Open Systems Interconnection), за функціональним призначенням та за масштабом використання. Такий підхід дозволяє системно розглянути всі типи пристроїв, визначити їхнє місце в структурі мережі та зрозуміти, яку роль вони відіграють у процесі передавання даних. Модель OSI (рис.1.1) є фундаментальною концепцією в галузі мережевих технологій, що надає структурований та ієрархічний підхід до опису процесів взаємодії відкритих систем. Вона складається із семи рівнів, кожен з яких виконує строго визначений набір функцій, забезпечуючи взаємозамінність, стандартизацію та модульність мережевих комунікацій. Саме завдяки такій архітектурі, пристрої та програмне забезпечення від різних виробників можуть безперешкодно обмінюватися даними.

Відповідно до моделі OSI, мережеві пристрої розподіляються за рівнями, на яких вони працюють.

Рисунок 1.1 - Організація моделі OSI

На фізичному рівні функціонують повторювачі та концентратори. Вони передають електричні сигнали між сегментами мережі, не аналізуючи їх вмісту, і забезпечують базове посилення сигналу на великих відстанях. Такі пристрої зазвичай використовуються в простих або малих локальних мережах, де важлива лише передача даних, а не їхнє фільтрування чи маршрутизація.

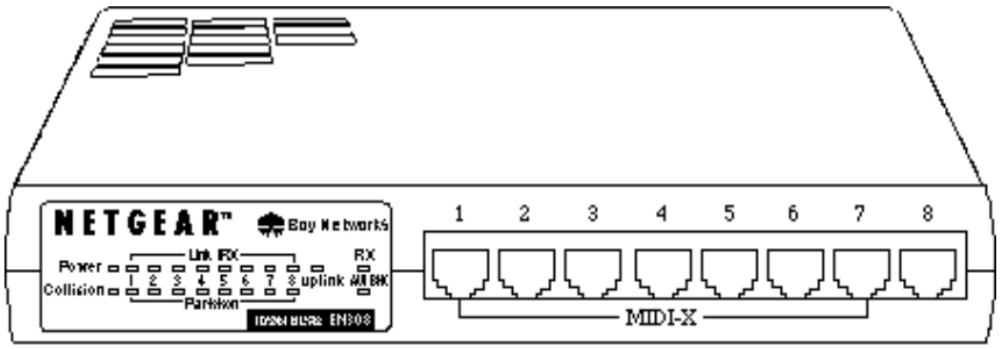

Рисунок 1.2 - Зовнішній вигляд комутатора

Канальний рівень обслуговують мости (рис. 1,3) і комутатори (рис. 1.2). Міст виконує роль фільтра між сегментами мережі, аналізуючи MAC-адреси й передаючи лише ті пакети, які призначені для конкретного сегмента.

Рисунок 1.3 - Структура моста

Комутатор, у свою чергу, є більш досконалим пристроєм, який дозволяє створювати окремі канали зв'язку між пристроями в межах однієї мережі, тим самим зменшуючи кількість колізій і підвищуючи ефективність передавання даних. Комутатори (рис. 1.4) можуть бути різних рівнів - від простих (некерованих моделей) до керованих, які в свою чергу підтримують: VLAN, QoS, протоколи безпеки та віддалене адміністрування.

Рисунок 1.4 - Комутатори компанії Cisco (Stackable Management Switches Series 350x)

На мережевому рівні працюють маршрутизатори (рис.1.5). Їхнє головне завдання – це визначення оптимального маршруту для передавання даних між різними мережами. Маршрутизатори аналізують IP-адреси та таблиці маршрутів, приймаючи рішення щодо найефективнішого шляху руху пакетів. У сучасних умовах маршрутизатори не лише спрямовують трафік, а й підтримують функції фільтрації, захисту, VPN-з'єднань та управління пропускною здатністю каналів.

Рисунок 1.5 - Маршрутизатор компанії Miktotik (CRS418-8P-8G-2S+5axQ2axQ-RM)

Вищі рівні моделі OSI: транспортний, сеансовий, представницькний та прикладний. Вони охоплюють більш складні пристрої, такі як шлюзи та міжмережеві екрани. Шлюз виконує функцію з’єднання між різними мережами або протоколами, наприклад між локальною мережею підприємства і глобальним Інтернетом. Міжмережевий екран забезпечує контроль і фільтрацію трафіку, створюючи бар’єр між внутрішнім і зовнішнім середовищем. У сучасних рішеннях ці пристрої часто об’єднані в одному комплексі, який виконує одразу кілька функцій: маршрутизацію, фільтрацію, антивірусний захист і управління доступом користувачів.

Сеансовий рівень є критично важливим для забезпечення керованого зв'язку між двома пристроями або програмами. Його основне призначення полягає у встановленні, підтримці та коректному завершенні сеансів зв’язку (логічних з’єднань). Це схоже на регулювання того, як довго триває телефонний дзвінок і чи можливо відновити розмову, якщо зв'язок раптово обірвався. До його ключових функцій належать: встановлення сеансу, що створює початкове логічне з’єднання; підтримка сеансу, яка забезпечує його стабільність та синхронізацію даних, дозволяючи продовжити роботу з потрібного місця після можливого переривання; та завершення сеансу, яке відповідає за правильне закриття з’єднання та звільнення системних ресурсів.

Представницький рівень виконує роль «перекладача» між прикладним рівнем, з яким взаємодіє кінцевий користувач, і нижчими рівнями мережі.

Його основне завдання полягає у забезпеченні того, щоб дані, які передаються, були відформатовані, кодовані та представлені таким чином, аби обидві комунікаційні сторони могли їх коректно зрозуміти та обробити.

Останнім найвищим рівнем моделі OSI є прикладний рівень. Основним призначення прикладного рівня полягає у наданні мережевих послуг безпосередньо кінцевому користувачу та його прикладним програмам. Цей рівень є точкою, де користувач взаємодіє з мережею. Він не надає послуг іншим рівням, а лише отримує їх від нижчих рівнів.

Якщо розглядати класифікацію за функціональним призначенням, то мережеве обладнання поділяють на передавальне, керуюче, захисне та допоміжне. Передавальні пристрої відповідають за фізичне переміщення даних усередині мережі, а керуючі забезпечують організацію взаємодії між різними сегментами або мережами. Захисні в свою чергу відповідають за безпеку та фільтрацію трафіку, а допоміжні використовуються для оптимізації навантаження і підвищення продуктивності. Наприклад, системи балансування навантаженням розподіляють запити між кількома серверами, щоб уникнути перевантаження, тоді як мультиплексори дозволяють передавати декілька потоків даних через один фізичний канал.

Ще одним важливим критерієм є масштаб використання. У цьому випадку виділяють три основні категорії обладнання: домашнього або офісного рівня (SOHO), корпоративного рівня та провайдерського або магістрального рівня. Домашнє обладнання зазвичай просте в налаштуванні та має обмежений набір функцій. Корпоративні рішення, навпаки, передбачають підтримку складних протоколів, розширене керування, інтеграцію з системами моніторингу та високий рівень безпеки. Провайдерське обладнання використовується великими телекомунікаційними компаніями і характеризується максимальною пропускною здатністю, надійністю та здатністю працювати під великим навантаженням цілодобово.

Сучасний розвиток мережевих технологій призвів до появи нових класів обладнання, заснованих на принципах програмно-конфігурованих мереж (SDN - Software Defined Networking) та віртуалізації мережевих функцій (NFV - Network Function Virtualization). Такі рішення дозволяють централізовано керувати мережевими ресурсами за допомогою програмного забезпечення, що значно спрощує адміністрування, підвищує гнучкість і забезпечує швидке масштабування інфраструктури.

Отже, класифікація мережевого обладнання є необхідним етапом у дослідженні комп'ютерних мереж. Вона дає змогу зрозуміти логіку побудови мережевих структур, виявити взаємозв'язки між різними типами пристроїв і оцінити їх роль у забезпеченні ефективної комунікації. На основі цієї систематизації стає можливим детальний аналіз провідних брендів, які створюють і впроваджують мережеві технології у світі.

1.2 Огляд мережевого обладнння, що постачають провідні компанії.

Після розгляду класифікації мережевого обладнання наступним кроком є аналіз провідних компаній, які випускають ці пристрої. Вивчення брендів дозволяє зрозуміти, як різні виробники реалізують функціональні можливості мережевих пристроїв, які технологічні рішення вони пропонують і які ринкові позиції займають.

Надалі розглянемо декілька ключових брендів світового ринку, серед яких: Cisco Systems, Juniper Networks, Huawei, Hewlett Packard Enterprise і MikroTik, а також проаналізуємо їхні основні продукти, функціональні можливості та роль у розвитку сучасних мереж.

Cisco Systems є одним із світових лідерів у виробництві мережевого обладнання та рішень для комп'ютерних мереж. Компанія пропонує широкий спектр продуктів (таблиця 1.1), які охоплюють маршрутизатори, комутатори, безпровідні точки доступу, міжмережеві екрани, системи управління мережею та програмні платформи для автоматизації. Завдяки широкому

портфелю рішень Cisco дозволяє будувати мережеву інфраструктуру різного масштабу - від невеликих офісів до великих корпоративних і провайдерських систем (Cisco, Networking Products).

Таблиця 1.1 - Порівняння серій комутаторів Cisco Catalyst (1200 / 1300 / 9200 / 9300 / 9400 / 9500) із ключовими характеристиками.

Серія	**Орієнтовне призначення**	**Приблизна портова щільність / uplink**	**Приблизна пропускна здатність / стекування**	**Примітка**
Catalyst 1200	Початковий рівень, малі офіси	~8-24 порти, 1 G або 10 G uplink	менше ніж у корпоративних серіях (Cisco)	Бюджетне рішення
Catalyst 1300	Малі та середні бізнес-мережі	більше ніж 1200-серія, ширший вибір портів	середній рівень (PivIT Global)	Проміжний варіант бюджету
Catalyst 9200	Доступ і агрегація в кампусах, малі/середні мережі	24-48 портів, 1/10 G uplink (Cisco)	Стекування до ≈ 160 Gbps (Cisco)	Основа сімейства 9000
Catalyst 9300	Середній-великий бізнес, доступ + агрегація	24-48 портів, 1/2.5/5/10/25/40 G uplink (Cisco)	Близько 640 Gbps (перемикача) (Cisco)	Потужніший, функції корпоративного рівня
Catalyst 9400	Модульний доступ та агрегація в великих мережах	великі конфігурації (120-192 порти, мультигіг) (Cisco)	До ~0.48-3.8 Tbps в залежності від моделі (Cisco)	Модульна архітектура
Catalyst 9500	Ядро і розподіл у великих підприємствах	приклади: 32×100 G, 48×25 G, 48×10/1 G (Cisco)	До ~6.4 Tbps перемикальної здатності (Cisco)	Фіксований ядровий комутатор

У сегменті комутаторів компанія пропонує як рішення для доступного рівня, так і пристрої для ядра мережі та магістральних сегментів. Наприклад, серія Cisco Catalyst застосовується у корпоративних локальних мережах і підтримує сучасні функції безпеки, управління трафіком, створення віртуальних мереж VLAN та автоматизацію адміністрування (Cisco, Catalyst Switches). Ці комутатори підтримують новітню оптику (100 G, 400 G), мають апаратну основу на програмованих ASIC-чипах (Cisco Silicon One / UADP) і керуються операційною системою IOS XE. Cisco+1

Наприклад, моделі доступу (access) в серії можуть мати 48 портів і підтримку 1/2.5/5/10 G, що дозволяє організовувати зв’язок у великих

приміщеннях. Ця серія підходить для побудови «ядро–агрегат–доступ» мереж і використовується там, де потрібна масштабованість та інтеграція із бездротовою частиною мережі.

Також популярною серією є Nexus, яка орієнтована на дата-центри і великі магістральні мережі. Моделі як-от *Nexus 9000* підтримують екстремально високу щільність портів, 100 G/400 G, низьку затримку, телеметрію та протоколи, оптимізовані для середовищ типу AI/ML чи хмарних платформ. Cisco+1. Ці комутатори використовуються там, де мережа має бути максимально потужною, надійною і з високою пропускною спроможністю. Наприклад, у центрі обробки даних або для міждата-центрових зв'язків.

Маршрутизатори Cisco застосовуються як у філіях підприємств, так і в масштабних мережах операторів (таблийя 1.2). Компанія також розвиває рішення для SD-WAN і підтримує концепцію Intent-Based Networking (IBN), що передбачає автоматизацію мережевих процесів, інтелектуальне налаштування та адаптацію до змінних умов експлуатації (Cisco, Enterprise Networking Solutions). А популярні серії ISR від Cisco призначені для організації мережевого зв'язку на периферії мережі, тобто такі як філії, офіси та віддалені точки. Вони поєднують такі функції, як маршрутизація, безпека, комутація і доступ до WAN, дозволяючи спрощувати інфраструктуру.

Таблиця 1.2 - Маршрутизатори Cisco ISR

Серія / модель	**Орієнтовна продуктивність та ключові характеристики**	**Призначення**
ISR 1000 Series	Комбінація маршрутизації, комутації, Wi-Fi, безпеки та DSL/LTE-підключення.	Віддалені офіси, філії, легка гілкова мережа
ISR 1100 / 1100X	Платформи для периферії: WAN-транспорти, VPN, SD-WAN-функції.	Менші офіси/філії з потребою в гнучкому WAN
ISR 4000 Family	Платформа «розумного WAN»: висока продуктивність, інтеграція сервісів (безпека, хмарні зв'язки).	Середні підприємства, офіси з великою інфраструктурою

Перевагами є універсальність, тобто оптимізована маршрутизація і забезпечення безпеки, а також можливість масштабування, ця серія є оптимальною для офісів і філій. Недоліками є те, що ця серія не завжди підходить для магістральних мереж провайдерів, адже функціонал може бути обмежений порівняно з найвищими моделями.

Однією з найпопулярніших серій є також - ASA, що призначена для забезпечення безпеки мережі, тобто захисту трафіку, VPN-з'єднань, фільтрації, контролю доступу (таблиця 1.3). Моделі ASA 5500-X охоплюють діапазон продуктивності від кількасот мегабіт до кількох гігабіт на секунду, з підтримкою таких функцій, як інспекція трафіку, VPN, модулі мережевої безпеки. Cisco+1 Ця лінійка включає різні форм-фактори та моделі, які підходять як для середніх підприємств, так і для великих мережевих середовищ. Операційна система ASA Software дозволяє інтегрувати firewall-функції з VPN, захистом від вторгнень і кластеризацією для підвищеної доступності.

Переваги серії: висока надійність, довга історія, широкий вибір форм-факторів, а недоліками є те, що старі моделі можуть бути у стадії завершення підтримки, потребують оновлення ПЗ, для дуже великих навантажень можуть бути менш оптимальними ніж рішення класу провайдер-рівня. Важливо, що цінова категорія залежить від моделі і пропускної здатності, тому може бути дуже високою.

Таблиця 1.3 - Порівняння серій міжмережевих екранів Cisco серії ASA

Серія / модель	**Орієнтовна продуктивність та ключові характеристики**	**Призначення**
ASA 5500 Series	Серія включає моделі, орієнтовані на малий/середній бізнес; кілька гігабітів пропускної здатності. (Cisco)	Філії, офіси, периферійні підрозділи
ASA 5500-X Series Next-Gen	Платформи з NGFW-функціями: багатосервісний захист, модульність, висока продуктивність. (Cisco)	Більші підприємства, рішення з високою безпекою
ASA Virtual (ASAv)	Віртуальне рішення для середовищ, які потребують firewall / VPN функцій у віртуалізованій чи хмарній інфраструктурі. (Cisco)	Хмарні середовища, віртуалізація мережі

Переваги Cisco включають:

- надійність і стабільність роботи обладнання;
- широкий вибір моделей для різних рівнів мережевої інфраструктури;
- високий рівень безпеки та підтримка сучасних протоколів;
- потужна технічна документація та навчальні ресурси;
- підтримка програмного керування мережами та автоматизації (SDN, IBN).

Недоліки Cisco полягають у:

- висока вартість обладнання порівняно з іншими брендами;
- досить складне налаштування для початкових користувачів без попереднього досвіду;
- необхідність регулярного оновлення ліцензій та програмного забезпечення для підтримки всіх функцій

Cisco відноситься до преміального сегменту мережевого обладнання. Навіть базові моделі корпоративних комутаторів або маршрутизаторів мають вартість вище середнього рівня ринку, тоді як дата-центрові рішення і високопродуктивні маршрутизатори відносяться до високого цінового сегмента. Вартість обґрунтована широким функціоналом, надійністю та тривалим терміном експлуатації, що робить продукцію Cisco привабливою для великих компаній і організацій, де критично важлива стабільність і безпека мережі.

Juniper Networks є також одним із провідних світових виробників мережевого обладнання, орієнтованого насамперед на корпоративний сектор та телекомунікаційних операторів. Продукція компанії широко застосовується у побудові високопродуктивних магістральних, дата-центрових і хмарних мереж, де ключовими вимогами є стабільність, масштабованість і надійність передачі даних.

Одним із головних технологічних досягнень компанії є операційна система Junos OS, яка використовується практично на всіх пристроях Juniper: маршрутизаторах, комутаторах, брандмауерах. Junos OS створена на основі

ядра FreeBSD і забезпечує єдину архітектуру управління для всієї лінійки пристроїв, що суттєво спрощує адміністрування мережі. Завдяки цьому адміністратори можуть централізовано конфігурувати обладнання, автоматизувати оновлення та віддалено керувати великою кількістю вузлів без ризику несумісності або конфліктів у конфігураціях.

Комутатори серій QFX і EX призначені для рівнів доступу, агрегації та ядра мережі. Пристрої серії QFX застосовуються переважно у центрах обробки даних завдяки підтримці високошвидкісних інтерфейсів 40/100 Гбіт/с і можливості побудови Spine-Leaf архітектури. Комутатори EX більше орієнтовані на корпоративний сегмент і забезпечують оптимальний баланс між ціною, продуктивністю та енергоспоживанням.

Таблиця 1.4 - Порівняльна характеристика маршрутизаторів Juniper серій MX, PTX та SRX

Серія маршрутизаторів	**Основне призначення**	**Ключові характеристики**	**Типові сфери застосування**	**Особливості**
Juniper MX	Універсальна платформа для операторських і корпоративних мереж	Висока пропускна здатність, підтримка MPLS, SDN, QoS, віртуалізація сервісів	Магістральні мережі, ядро мережі провайдерів, корпоративні центри обробки даних	Надійність, масштабованість, розширені можливості трафік-інжинірингу
Juniper PTX	Платформа для магістральних і дата-центрових мереж	Оптимізована для високошвидкісної маршрутизації, підтримка 100G/400G інтерфейсів, енергоефективність	Великі дата-центри, глобальні мережі операторів зв’язку	Орієнтація на продуктивність і ефективність, мінімальні затримки
Juniper SRX	Поєднання маршрутизатора і системи безпеки	Інтегровані функції міжмережевого екрана, IPS, VPN, фільтрація трафіку	Корпоративні мережі, захист периметра, віддалений доступ	Забезпечення комплексної безпеки, централізоване управління через Junos Space Security Director

Маршрутизатори серій MX, PTX та SRX – забезпечують високошвидкісну маршрутизацію для операторських і корпоративних мереж. Серія MX часто використовується для побудови магістральних каналів зв’язку та агрегування трафіку, тоді як PTX орієнтована на роботу в масштабних дата-центрах. Серія SRX поєднує функції маршрутизатора та

міжмережевого екрана, що дозволяє реалізовувати комплексну безпеку мережевого периметра.

Мережеві рішення безпеки включають міжмережеві екрани (firewalls) і системи запобігання вторгненням (IPS), що інтегруються в загальну екосистему керування Juniper Security Director.

До ключових переваг рішень Juniper належать:

- висока продуктивність та стабільність при роботі під великим навантаженням,
- надійність завдяки модульній архітектурі та відмовостійкості,
- єдина операційна система для всіх пристроїв, що знижує складність управління,
- підтримка автоматизації і глибокої інтеграції з SDN та хмарними платформами.

Таблиця 1.5 - Порівняльна характеристика комутаторів Juniper серій QFX та EX

Серія комутаторів	**Основне призначення**	**Ключові характеристики**	**Типові сфери застосування**	**Особливості**
Juniper QFX	Високопродуктивні комутатори для центрів обробки даних і магістральних мереж	Підтримка технологій EVPN-VXLAN, 25G/100G/400G портів, масштабовані фабрики, SDN-сумісність	Дата-центри, хмарні інфраструктури, великі корпоративні мережі	Оптимізовані для Spine-Leaf архітектури, низькі затримки, інтеграція з Junos Fusion
Juniper EX	Універсальні комутатори доступу і агрегації для корпоративних мереж	Підтримка PoE/PoE+, Virtual Chassis, безпечного доступу, автоматизації через Junos Space	Офісні мережі, кампусні рішення, середні підприємства	Простота розгортання, централізоване управління, сумісність із QFX для побудови єдиної мережевої інфраструктури

Разом з тим, обладнання Juniper має і певні недоліки. Основним є висока вартість у порівнянні з конкурентами, що робить ці рішення менш доступними для малого бізнесу. Крім того, навчальна база та кількість спеціалістів, сертифікованих саме у напрямі Juniper, помітно менша, ніж у випадку Cisco. Це ускладнює підготовку персоналу та технічну підтримку у невеликих організаціях.

Huawei Enterprise є одним із провідних світових виробників мережевого обладнання для корпоративних, операторських і дата-центрових рішень. Компанія пропонує широкий асортимент пристроїв для побудови сучасних мережевих інфраструктур, тобто від базових офісних комутаторів і маршрутизаторів до складних систем для центрів обробки даних і хмарних платформ. Основна мета Huawei у цьому напрямі полягає в поєднанні високої продуктивності, масштабованості та інтелектуального управління мережевими ресурсами.

Серед маршрутизаторів можна виділити найпопулярніші серії, а саме AR та NE. Серія AR (Access Router) орієнтована на корпоративні та філіальні мережі, а серія NE (Network Engine) призначена для операторських мереж.

Таблиця 1.6 - Порівняльна характеристика маршрутизаторів Huawei серій AR та NE і магістральних мереж.

Серія маршрутизаторів	**Основне призначення**	**Ключові характеристики**	**Типові сфери застосування**	**Особливості**
Huawei AR (Access Router)	Корпоративні та філіальні мережі	Підтримка маршрутизації, VPN-з'єднань, безпеки та SD-WAN; інтеграція дротового й бездротового доступу (Huawei Carrier)	Офіси, філії, віддалені точки підприємств	Орієнтація на офісну/філіальну архітектуру, гнучкість у доступі
Huawei NE (Network Engine)	Операторські і магістральні мережі	Передача великих обсягів трафіку до ~400 Гбіт/с; підтримка протоколів MPLS, Segment Routing, IPv6 (Router-Switch.com)	Магістральні мережі операторів, великі датацентри	Орієнтовані на високу пропускну здатність, масштабованість і надійність мережі

Комутатори CloudEngine створені для центрів обробки даних, підтримують архітектуру Spine-Leaf і високошвидкісні інтерфейси 25/40/100 Гбіт/с. Серія S-Series, в свою чергу, орієнтована на рівні доступу та агрегації корпоративних мереж, підтримує енергозбереження, PoE, функції безпеки й віртуалізації.

Компанія пропонує точки доступу Wi-Fi 6 (серії AirEngine) і контролери для управління бездротовими мережами, що дозволяє

забезпечувати високу щільність підключень та стабільне покриття у великих офісах або кампусах.

Huawei має власні міжмережеві екрани (firewalls) серій USG та USG FLEX, які поєднують функції міжмережевого екрану, системи запобігання вторгненням (IPS), антивірусного захисту та шифрування VPN. Ці пристрої забезпечують гнучке управління політиками безпеки та інтегруються у загальну екосистему Huawei eSight і HiSec.

Таблиця 1.7 - Порівняльна характеристика комутаторів Huawei серій CloudEngine та S-Series

Серія комутаторів	**Основне призначення**	**Ключові характеристики**	**Типові сфери застосування**	**Особливості**
Huawei CloudEngine (CE)	Комутатори для дата-центрів і магістральних мереж	Підтримка архітектури Spine-Leaf, інтерфейси 25/40/100/400 Гбіт/с, SDN та автоматизація через CloudFabric, підтримка VXLAN	Центри обробки даних, хмарні інфраструктури, операторські магістральні мережі	Висока масштабованість, низька затримка, інтеграція з системами AI та керування трафіком
Huawei S-Series	Комутатори доступу та агрегації корпоративних мереж	Підтримка PoE/PoE+, енергозбереження (Energy Efficient Ethernet), безпека (802.1X, ACL, MACsec), віртуалізація iStack	Корпоративні мережі, кампусні мережі, філіальні офіси	Просте розгортання, висока надійність, централізоване керування через iMaster NCE

Технологічні особливості мережевого обладнання Huawei зосереджені на підтримці SDN (Software Defined Networking) та інтелектуальному управлінні мережею. Основою для цього є програмні платформи Huawei CloudCampus, iMaster NCE і CloudFabric, які дозволяють централізовано адмініструвати як дротові, так і бездротові сегменти, автоматизувати конфігурацію пристроїв, моніторинг і оптимізацію трафіку. Також підтримується інтеграція з хмарними сервісами та відкриті API для сумісності з рішеннями інших виробників.

Однією з ключових переваг Huawei є масштабованість і універсальність її рішень. Компанія пропонує обладнання для будь-яких сценаріїв - від невеликих офісів до національних операторів зв’язку. Серед інших переваг варто відзначити конкурентну ціну, високу

продуктивність, енергоефективність, а також розвинену систему автоматизації мережевого управління.

Серед недоліків можна відокремити обмежену локальну технічну підтримку у деяких регіонах, меншу кількість навчальних матеріалів англійською мовою, а також регуляторні обмеження на використання обладнання Huawei у певних країнах.

Загалом, рішення Huawei Enterprise належать до середнього та преміум сегментів ринку, поєднуючи високу функціональність із доступнішою ціною порівняно з конкурентами

Таблиця 1.8 - Порівняльна характеристика комутаторів Aruba серій 2930 / 2930M та 3810M

Серія комутаторів	**Основне призначення**	**Ключові характеристики**	**Типові сфери застосування**	**Особливості**
Aruba 2930 / 2930M	Комутатори доступу та агрегації в кампусних і корпоративних мережах	Підтримка стекування до 10 пристроїв, PoE/PoE+, 10GbE uplink-порти (SFP/SFP+), Layer 3 routing (RIP, OSPF), QoS, безпека 802.1X	Корпоративні мережі малого та середнього бізнесу, відділення компаній, навчальні заклади	Простота розгортання, підтримка централізованого керування через **Aruba Central**, інтеграція з бездротовими рішеннями Aruba
Aruba 3810M	Комутатор для агрегації та ядра середнього корпоративного рівня	Високошвидкісні інтерфейси 40GbE, модульна архітектура, стекування до 10 пристроїв, підтримка VRRP, QoS, IPv6, ACL, PoE/PoE+	Кампусні мережі, середні та великі офіси, корпоративні ядра	Підвищена масштабованість, резервування живлення/вентиляції, інтеграція з **ArubaOS-Switch** і **Aruba Central**, зручність адміністрування

Hewlett Packard Enterprise (HPE) разом із брендом Aruba пропонують комплексні мережеві рішення, орієнтовані на корпоративні, кампусні та малі мережеві середовища. До портфеля компанії входять дротові комутатори серій Aruba 2930 та 2930M та Aruba 3810M, бездротові точки доступу, маршрутизатори, контролери, а також платформи централізованого управління мережею, серед яких ключову роль відіграє Aruba Central. Ці рішення забезпечують інтегроване керування як дротовими, так і бездротовими сегментами інфраструктури, що спрощує моніторинг, адміністрування та масштабування мережі.

Технологічно HPE/Aruba підкреслює інтеграцію апаратних засобів із програмним управлінням, а технології підтримують:

- сучасні бездротові протоколи (Wi-Fi 6 / Wi-Fi 6E);
- SD-WAN, Private 5G;
- централізовану платформу управління (Aruba Central);
- автоматизацію мережевих процесів.

Комутатори та точки доступу можуть інтегруватися в одну мережеву екосистему, що дозволяє централізовано моніторити і адмініструвати як дротові, так і бездротові сегменти інфраструктури.

Серед переваг HPE/Aruba варто виділити: зручність адміністрування, надійне обладнання, адаптацію до середнього бізнесу, а також ефективне поєднання дротових і бездротових компонентів. Разом з тим існують і недоліки: менша присутність на ринку великих провайдерських чи магістральних мереж порівняно з лідерами галузі, певна обмеженість при дуже високих навантаженнях/дата-центрах. В цілому рішення HPE/Aruba належать до середнього та преміум сегмента ринку, пропонуючи баланс між ціною, функціональністю та надійністю.

Тут ми розглянули основні сімейства мережевого обладнання, яке випускається провідними компаніями світу і найбільш розповсюджене на ринку. Але програмне забезпечення даних брендів для керування мережевим обладнанням в більшості випадків коштує додаткових затрат і керує тільки своїм обладнанням. Тому потрібні засоби для універсального управління різноманітним обліднанням від різних виробників, використовуючи стандартні протоколи які повинно підтримувати кероване мережеве обладняння.

1.3 Способи керування пристроями комп’ютерної мережі

Контроль роботи мережевих пристроїв передбачає використання різних способів і засобів, які дозволяють стежити за їхнім станом, аналізувати роботу та керувати функціонуванням. У межах кваліфікаційної роботи

розглядаються основні системи перевірки працездатності пристроїв та методи їх моніторингу.

Одним із важливих механізмів забезпечення стабільності роботи мережевих систем є технологія WatchDog (з англійської - «Сторожовий пес»). У сфері комп'ютерних систем цей термін позначає поєднання двох складових: сторожового таймера (рис.1.6) (WatchDog Timer, WDT) та програмного забезпечення Watchdog. Обидві ці частини працюють разом для підтримки надійності системи, своєчасно виявляючи помилки або збої у роботі програм чи обладнання.

Рисунок 1.6 – Мікросхема сторожевого таймера DS1386-32-120+

Сторожовий таймер може бути реалізований як окремий апаратний модуль або у вигляді частини програмного коду. Його головне завдання – це стежити, щоб основна програма або операційна система регулярно надсилала сигнал, який «скидає» таймер і підтверджує, що система працює справно. Якщо з будь-якої причини операційна система або програма перестає відповідати, наприклад, зависає або аварійно завершує роботу, таймер спрацьовує після закінчення заданого часу. У цей момент виконується заздалегідь визначена дія, зазвичай перезапуск системи або відновлення її роботи, що дозволяє уникнути тривалих простоїв і забезпечити безперервність функціонування пристрою.

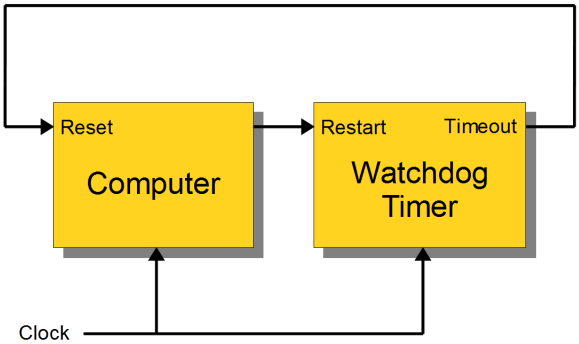

Рисунок 1.7 - Схема прикладу роботи Стороржовго таймера.

Якщо ж програмне забезпечення зависає або потрапляє у нескінченний цикл, процес скидання не відбувається. У такому випадку таймер продовжує відлік до кінця встановленого інтервалу, після чого автоматично ініціює перезавантаження системи чи окремої програми. Це дозволяє уникнути тривалого простою та повернути пристрій до стабільної роботи без втручання користувача.

Схематично робота сторожового таймера (рис. 1.7), (рис. 1.8) виглядає так: основна програма з певною періодичністю повинна подавати сигнал на його скидання. Якщо цього не відбувається у визначений проміжок часу, таймер активує наперед задану дію, зазвичай це примусовий перезапуск системи, як продемонстровано на рисунку 1.8.

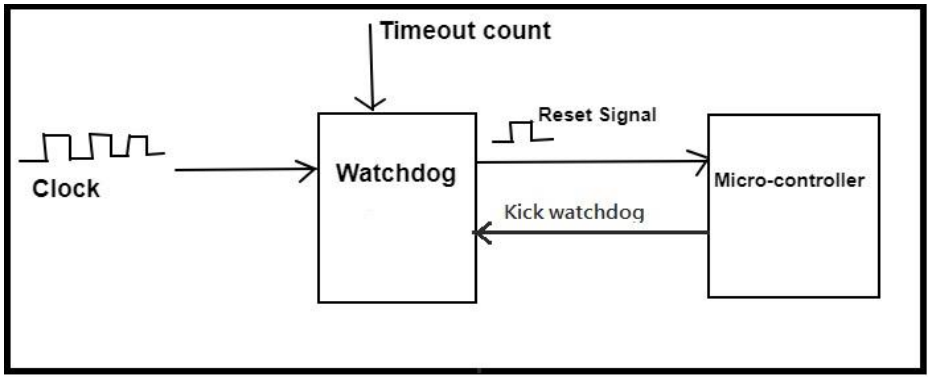

Рисунок. 1.8 – Схема алгоритму роботи сторожового таймера

Сторожовий таймер WDT може виконувати кілька дій, спрямованих на відновлення або захист системи у разі її збою. Найчастіше він автоматично перезавантажує програму чи операційну систему, що дозволяє усунути помилку та повернути систему до нормального стану. У деяких випадках таймер може перевести пристрій у безпечний режим, у якому працює лише мінімальний набір функцій — це дає змогу проаналізувати проблему або відновити дані без повного зупинення роботи. Крім того, сторожовий таймер здатен надсилати попередження чи повідомлення про помилки, що допомагає своєчасно виявити потенційні несправності.

Сфера застосування WDT доволі широка. Такі таймери часто використовують у серверах і блейд-системах, де стабільність і безперервність роботи мають критичне значення. Вони також поширені у вбудованих рішеннях, наприклад, у мікроконтролерах, промислових контролерах або автоматизованих системах керування, де навіть короткочасний збій може призвести до серйозних наслідків.

У мережевих пристроях, таких як маршрутизатори, комутатори чи брандмауери, сторожові таймери використовуються для підтримання стабільності та безперервності роботи мережі. Їхнє значення особливо помітне у критично важливих системах безпеки, наприклад, у медичному обладнанні або авіаційних системах керування, де навіть короткочасний збій може створити небезпеку. У таких випадках WDT виконує роль механізму, який забезпечує надійність і своєчасне відновлення роботи пристроїв.

У більшості серверів базові параметри сторожового таймера можна налаштувати безпосередньо в системному BIOS. Як показано на рисунку 2.4, у BIOS серверів, зокрема моделі Tyan**,** передбачено можливість увімкнення або вимкнення функції WDT. Крім того, користувач може вибрати, яку саме дію виконуватиме система у разі зависання — повне вимкнення живлення чи автоматичне перезавантаження. Також доступне налаштування часу, через який таймер має спрацювати, наприклад, через 5, 10, 15 або 20 хвилин після втрати відповіді від основної програми.

Рисунок 1.9 – Налаштування Watchdog сервера Tyan

Деякі сторожові таймери мають розширені можливості, що дозволяють налаштовувати допустимий час скидання у заданому діапазоні (рис. 1.9), роблячи їх зручнішими для складних систем. Такі Watchdog можуть контролювати різні параметри: завантаження процесора, використання пам’яті, швидкість реакції програм чи стан мережевого підключення. У разі виявлення відхилень вони здатні сповіщати адміністратора через електронну пошту, SMS або звукові й візуальні сигнали. Крім того, сторожові таймери можуть допомагати виявляти та запобігати атакам типу DoS, відстежуючи перевантаження системних ресурсів і автоматично перезапускаючи проблемні служби.

Сторожові таймери мають низку переваг, що робить їх важливими елементами системної стабільності. Вони підвищують надійність роботи пристроїв, адже здатні автоматично виявляти й усувати збої, запобігаючи втраті даних і простою системи. Завдяки цьому забезпечується постійна працездатність навіть критично важливих компонентів мережі. Крім того, використання WDT дає змогу швидко відновити роботу після збою без потреби у ручному втручанні, що значно скорочує час простою.

Окрім апаратних реалізацій, існують і програмні варіанти сторожових систем. Таке сторожове програмне забезпечення (СПЗ) працює всередині операційної системи й контролює її стан. Воно відстежує використання

процесора, пам’яті, дискового простору та мережеву активність, а також реагує на зміни швидкості виконання програм. Якщо певна програма відповідає занадто повільно або зависає, це може свідчити про потенційні проблеми.

Переваги сторожового програмного забезпечення можна узагальнити так:

- на відміну від апаратних рішень, програмні системи пропонують ширші можливості контролю та реакції на збої, дозволяючи адаптувати роботу під різні сценарії;

- можливість персонального налаштування є важливою перевагою, адже користувач може самостійно визначати, які параметри потрібно відстежувати та які дії виконувати у разі виявлення проблеми;

- СПЗ забезпечує централізований контроль**.** Таке програмне забезпечення здатне здійснювати моніторинг кількох пристроїв у мережі з одного центру, що забезпечує повну картину стану всієї системи.

Програмні сторожові таймери широко застосовуються у середовищах, де стабільність і безперебійність роботи мають ключове значення, зокрема на серверах, що підтримують бізнес-додатки. Вони дозволяють контролювати роботу окремих програмних модулів, забезпечуючи стабільну продуктивність системи та своєчасне усунення збоїв. ІТ-фахівці використовують такі інструменти для раннього виявлення неполадок у мережі й запобігання їх негативному впливу на роботу пристроїв. Прикладом ефективного використання сторожового програмного забезпечення є системи SCADA (наглядовий контроль і збір даних), які застосовуються для керування промисловими процесами. У таких системах Watchdog відіграє ключову роль, забезпечуючи безперервність роботи, виявлення збоїв і швидку реакцію на можливі проблеми.

Через IPMI можна підключатися до сервера за допомогою IP-з’єднання та користуватися функціями KVM через IP**,** що забезпечує повний віддалений доступ до консолі системи. Інтерфейс також дозволяє

здійснювати моніторинг фізичного стану обладнання, зокрема температури компонентів, рівня напруги, швидкості обертання вентиляторів та інших показників.

Крім спостереження, IPMI надає такі засоби керування: можна виконувати перезавантаження сервера, вмикати або вимикати живлення, монтувати ISO-образи для інсталяції системи чи оновлювати програмне забезпечення. Адміністратори також мають змогу переглядати журнал подій, перевіряти стан периферійних пристроїв і зберігати інформацію про конфігурацію обладнання.

У більшості сучасних серверів IPMI інтегровано безпосередньо в материнську плату, тоді як у старіших моделях ця функція може бути додана через окремий модуль. Налаштування інтерфейсу зазвичай здійснюється через BIOS або UEFI, де задаються параметри автентифікації, такі як: IP-адреса, логін і пароль користувача. Після цього IPMI забезпечує можливість повного дистанційного керування сервером, навіть якщо він недоступний у межах основної мережі.

Для керування сервером через IPMI можна використовувати різні способи, зокрема веб-інтерфейс або спеціалізовані програми, такі як IPMI View, FreeIPMI та OpenIPMI**.** Ці інструменти надають зручний інтерфейс, що спрощує віддалене управління сервером рис. 11.10) і дозволяє ефективно виконувати різні адміністративні дії.

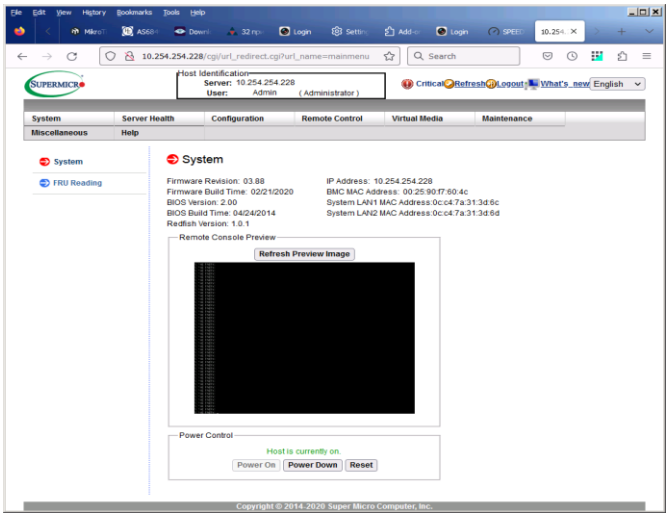

Рисунок 1.10 – Віддалене керування сервером Supermicro

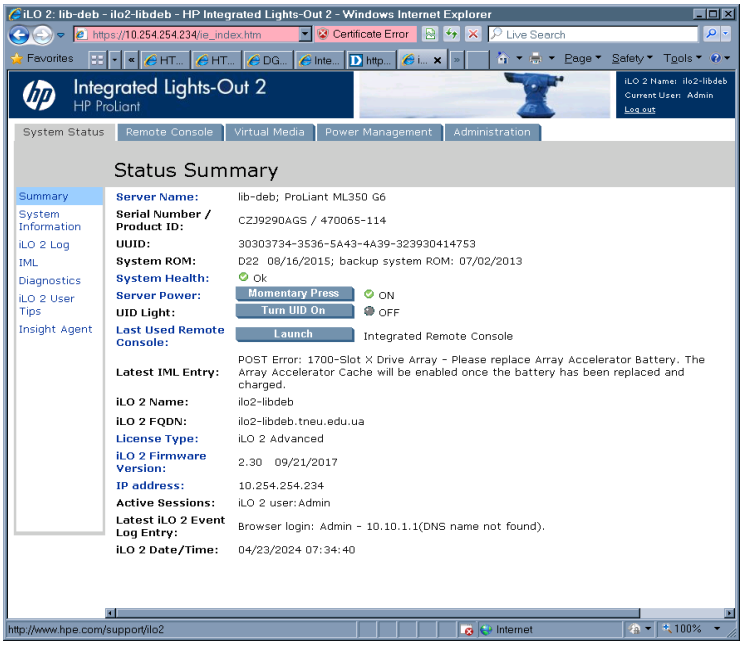

Рисунок 1.11 – Віддалене керування ILO сервера HP.

Рішення різних виробників відрізняються кількома ключовими характеристиками. По-перше, вони мають різний рівень візуалізації стану обладнання, що впливає на зручність моніторингу. По-друге, кожна система пропонує свій набір програмних інструментів для відновлення працездатності серверів у разі відмови окремих компонентів. Також вони відрізняються здатністю збирати статистику по всіх складових сервера, включаючи ті, що підключені через карти розширення PCI, NVM та інші інтерфейси. Крім того, деякі технології IPMI можна використовувати не лише у серверному обладнанні, а й у звичайних комп'ютерах, завдяки спеціальним платам розширення PCI-Express.

Основним компонентом IPMI є мікроконтролер BMC (Baseboard Management Controller), який є «мозком» системи і відповідає за віддалене управління сервером. По суті, BMC — це окремий невеликий комп'ютер із власним програмним забезпеченням та мережним інтерфейсом. Він може бути інтегрований безпосередньо на материнській платі або підключений як плата розширення через шину PCI. На рисунку 1.12 зображена структура основних компонентів IPMI та їхнє взаємодія.

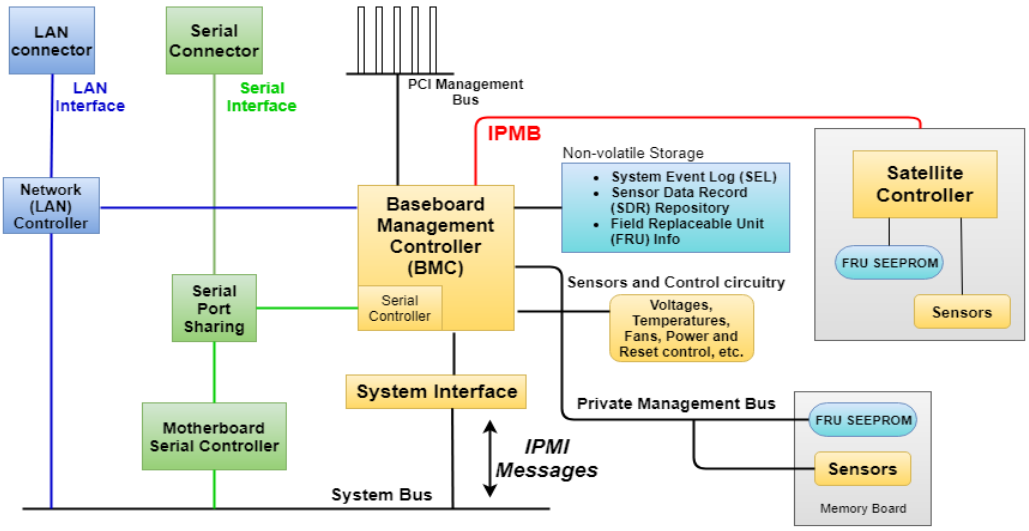

Рисунок 1.12 – Основні компоненти IPMI

Мікроконтролер BMC отримує живлення від основної материнської плати, що дає змогу йому функціонувати незалежно від стану сервера (рисунок 1.13).

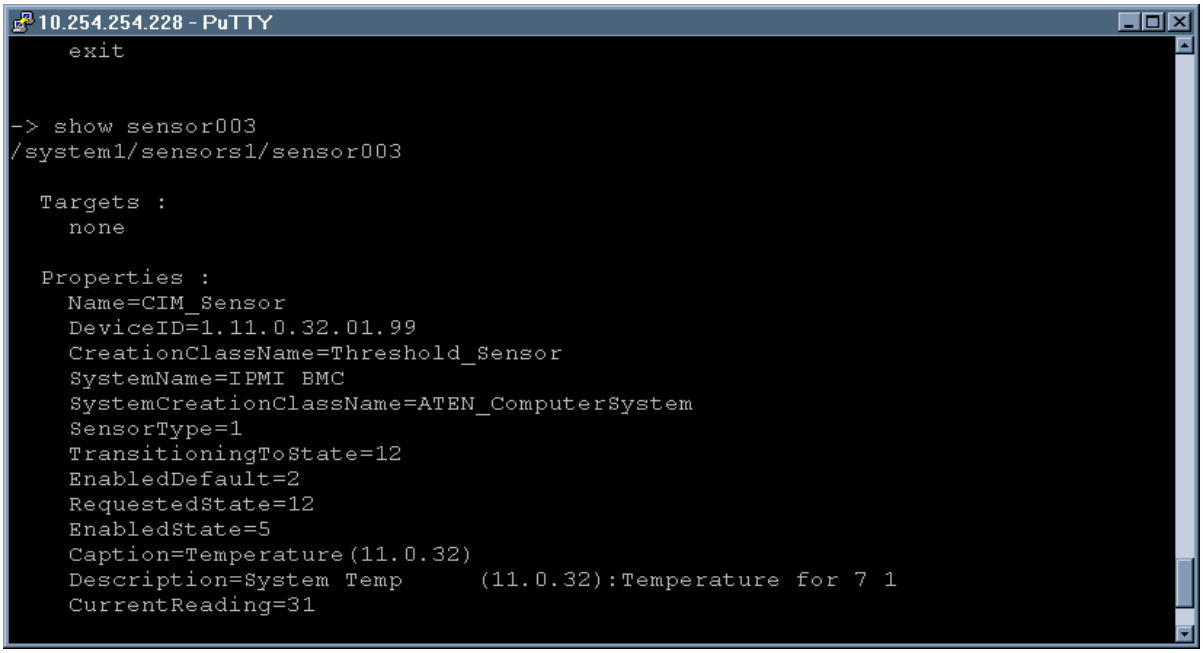

Рисунок 1.13 – Інформація про BMC

До BMC можна підключати додаткові контролери управління (Management Controllers, MCs) для розширення базових можливостей керування системою. Наприклад, поки основний BMC здійснює загальне

управління сервером, MCs відповідають за моніторинг окремих підсистем, таких як резервні джерела живлення, RAID-масиви та периферійні пристрої.

MCs постачаються як окремі плати і працюють незалежно від центрального BMC, тому їх часто називають супутниковими контролерами (Satellite Controllers). Хоча кількість додаткових контролерів може варіюватися, центральний BMC завжди залишається один.

Контролери підключаються до BMC через IPMB (Intelligent Platform Management Bus) — шину інтелектуального управління платформою, яка базується на протоколі I2C (Inter-Integrated Circuit).

BMC через IPMB виконує такі дії:

- взаємодіє з додатковими контролерами (MCs);
- отримує дані від сенсорів (рисунок 1.14);
- звертається до енергонезалежного сховища (Non-Volatile Storage).

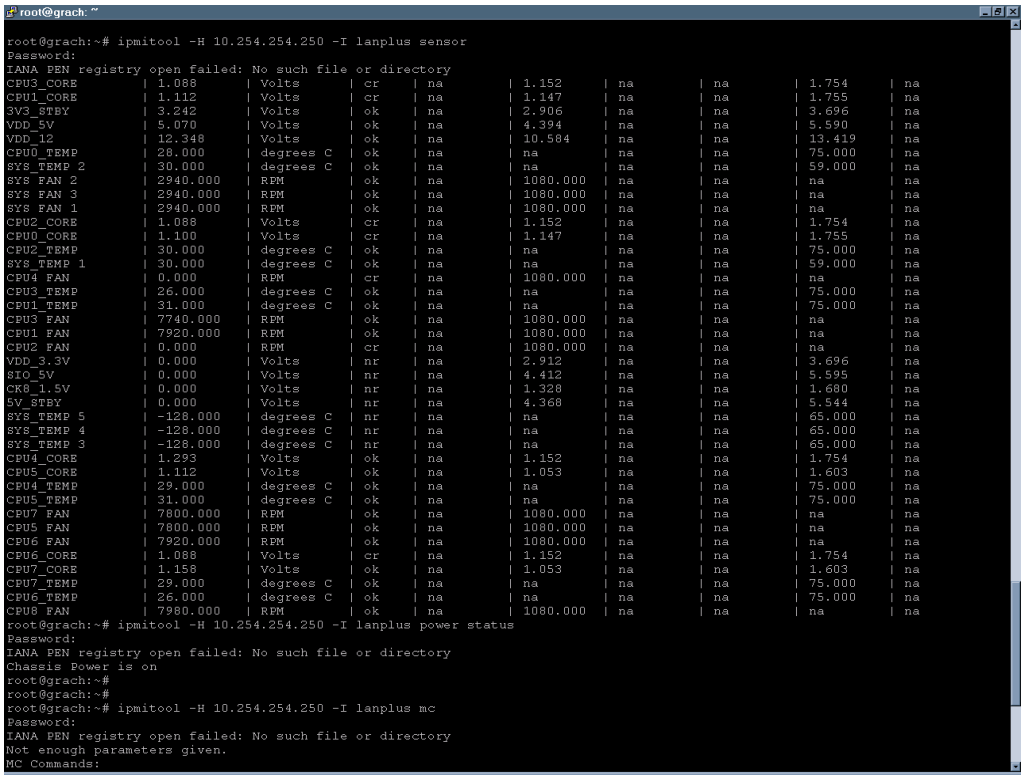

Рисунок 1.14 – Сенсори IPMI BMC

Архітектура IPMI надає віддаленому адміністратору обмежений доступ до компонентів системи. Наприклад, щоб отримати дані з сенсорів,

адміністратор надсилає команду на BMC, який потім взаємодіє безпосередньо з відповідними сенсорами. Окрім ручного керування, BMC можна налаштувати на автоматичне виконання певних дій за допомогою спеціальних механізмів, таких як PEF (Platform Event Filtering), Watchdog Timer та Firmware Firewall.

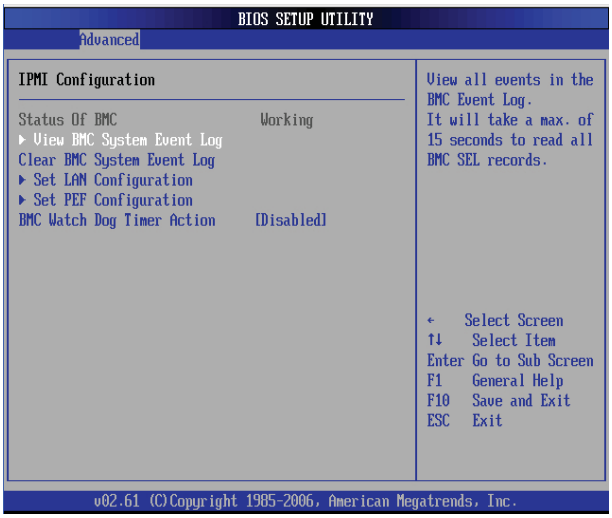

Рисунок 1.15 – Налаштування BMC у БІОС

У початкових версіях IPMI віддалена консоль підключалася до BMC через послідовний інтерфейс (Serial Interface). У специфікації IPMI v2.0 з'явилася можливість використання мережевого інтерфейсу (LAN Interface).

Мережевий інтерфейс LAN реалізується через виділений мережевий порт BMC, який має власну IP-адресу.

Ще одним способом контролю стану мережевих пристроїв є протокол SNMP (Simple Network Management Protocol), який є стандартним мережевим протоколом для керування та моніторингу пристроїв у мережах IP. Він інтегрований у різноманітні пристрої, такі як маршрутизатори, комутатори, концентратори, мости, повторювачі, шлюзи, сервери, брандмауери та бездротові точки доступу, і дозволяє отримувати доступ до них за допомогою IP-адрес. SNMP забезпечує стандартизований механізм передачі керуючої

інформації між пристроями у локальних і глобальних мережах і належить до прикладного рівня моделі OSI.

Для організації даних у SNMP застосовуються бази керування (MIB - Management Information Bases), які визначають структуру даних, доступних для отримання або зміни на пристрої. Існують стандартні MIB, розроблені організаціями стандартизації, такими як IETF і ISO, а також пропрієтарні MIB, які визначають окремі виробники обладнання та постачальники програмного забезпечення, наприклад Cisco, Microsoft та Oracle.

Інструменти моніторингу SNMP дозволяють автоматично виявляти, контролювати та керувати мережевими пристроями, а також відстежувати ключові показники їхньої продуктивності як на рівні окремих пристроїв, так і на рівні інтерфейсів. Протокол забезпечує детальне бачення стану мережевих елементів, а системний адміністратор може встановлювати порогові значення і створювати сповіщення у разі виникнення аномалій.

SNMP функціонує шляхом надсилання запитів SNMP GET до мережевих пристроїв, які у відповідь передають необхідні дані. Усі взаємодії відстежуються, і спеціалізовані інструменти моніторингу використовують ці запити для збору інформації. Протокол здатний працювати з усіма пристроями мережі, незалежно від джерела трафіку (рис.1.16).

На пристроях SNMP зазвичай попередньо налаштований. Після його активації пристрої починають збирати та зберігати статистику продуктивності. Кожен сервер або мережевий елемент має власні файли MIB (Management Information Base), які використовуються для запитів та отримання моніторингових даних. Робота протоколу базується на його компонентах, що забезпечують ефективне управління ресурсами. Оскільки SNMP є частиною стеку TCP/IP, повідомлення передаються через UDP, що дозволяє швидко обмінюватися даними без встановлення з’єднання.

Рисунок 1.16 – Принцип передачі SNMP

Системи контролю стану пристроїв, такі як Watchdog, IPMI та SNMP**,** відіграють ключову роль у забезпеченні надійності та доступності мережевих пристроїв і серверів. Були розглянуті їхні переваги та недоліки. Перші дві системи здебільшого реалізовані в дорогому мережевому або серверному обладнанні, тому для Автоматизованої системи керування обладнанням комп'ютерних мереж буде використаний SNMP**.**

2 АНАЛІЗ КОМП'ЮТЕРНОЇ МЕРЕЖІ ТА ЗБОЇВ МЕРЕЖЕВОГО ОБЛАДНАННЯ

2.1 Огляд існуючої комп'ютерної мережі

Розглянемо корпоративну комп'ютерну мережу (ККМ) університету. Апаратне забезпечення (серверне обладнання, маршрутизатори та комутатори рівня розподілу) фізично розміщені у двох серверних приміщеннях: I корпусу (СП 1) та II корпусу (СП 2).

Крім того, існують окремі серверні приміщення, що призначені для:

- розміщення серверного обладнання, яке забезпечує функціонування програмного забезпечення і локальної обчислювальної мережі в цілому відділу бухгалтерії (СП бухгалтерії);

- розміщення цифрової автоматичної телефонної станції та серверів VoIP-телефонії знаходиться у III корпусі, (СП АТС);

- розміщення активного мережевого обладнання корпусу бібліотеки (СП бібліотеки). Схема об'єднання корпусів, гуртожитків та філій Університету зображена на рисунку 2.1.

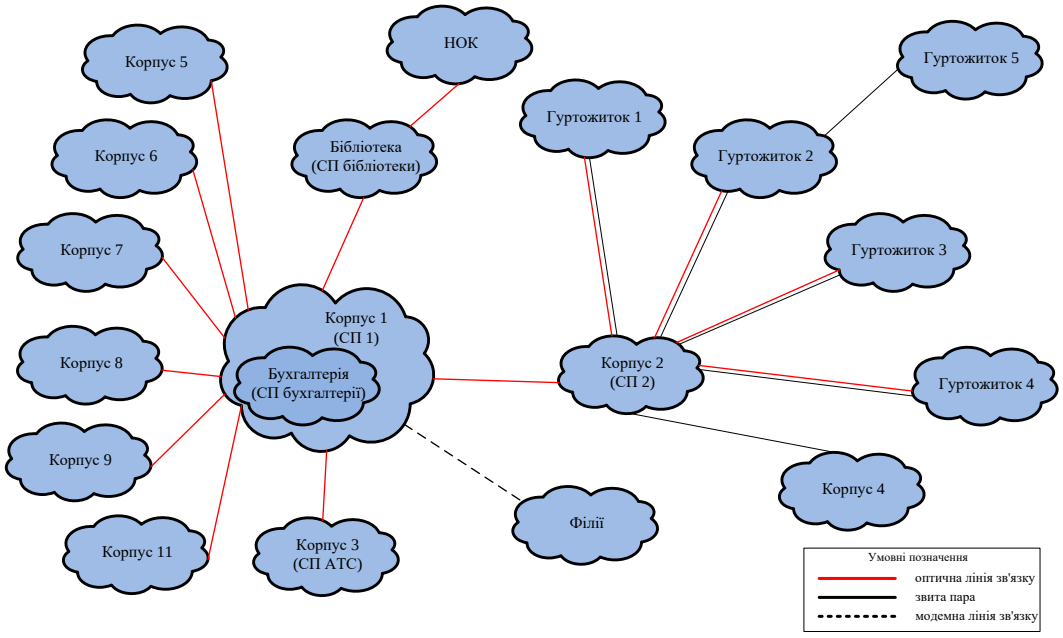

Рисунок 2.1 - Схема корпоративної комп'ютерної мережі університету

Також до складу ККМ університету входять локальні обчислювальні мережі гуртожитків.

Загалом ККМ університету об'єднує мережі 11+2 корпусів (з урахуванням корпусів бібліотеки та національного олімпійського комітету), 5 гуртожитків, а також мережі 12 віддалених філій.

Ядром корпоративної комп'ютерної мережі університету є стек комутаторів 3 рівня Cisco Nexus N5K-C5548UP (мережеве ім'я sw-nexus), який забезпечує комутацію серверного та активного мережевого обладнання (в тому числі комутаторів рівня розподілу, які знаходяться на територіально відокремлених майданчиках – корпусах, гуртожитках, філіях), а також маршрутизацію між віртуальними локальними мережами (далі – VLAN) університету.

В будівлях, які входять до складу університету, побудовано структуровану кабельну систему (СКС). Для підключення АРМ співробітників до СКС використовуються розетки RJ-45 та кабелі типу «вита пара», які прокладено у коробах та/або по підлозі вздовж стін. У випадку, якщо ємності розеток не вистачає використовуються додатково розміщені концентратори.

Серверне приміщення 1 корпусу (рис. 2.2) розміщено на першому поверсі 1 корпусу університету у підсобному приміщенні одного з кабінетів. Апаратне забезпечення змонтовано в комутаційних шафах. Кабелі прокладено за допомогою утримувачів або навісним монтажем. Периметр приміщення не екрановано, доступ обмежено дерев'яними дверима з механічним замком. Пожежна сигналізація присутня. Система автоматичного пожежогасіння присутня. У якості засобу пожежогасіння використовується вуглекислотний вогнегасник.

Система контролю/підтримки кліматичних умов представлена двома кондиціонерами типу спліт-система загальною охолоджуючою потужністю 5-6 кВт.

Охоронна сигналізація представлена об’ємними датчиками. Приміщення опечатується та здається під охорону.

Доступ до серверного приміщення, а також порядок фіксації робіт, що в ньому проводяться (журнал відкриття/закриття серверного приміщення, журнал реєстрації та обліку робіт на серверах тощо), не визначено.

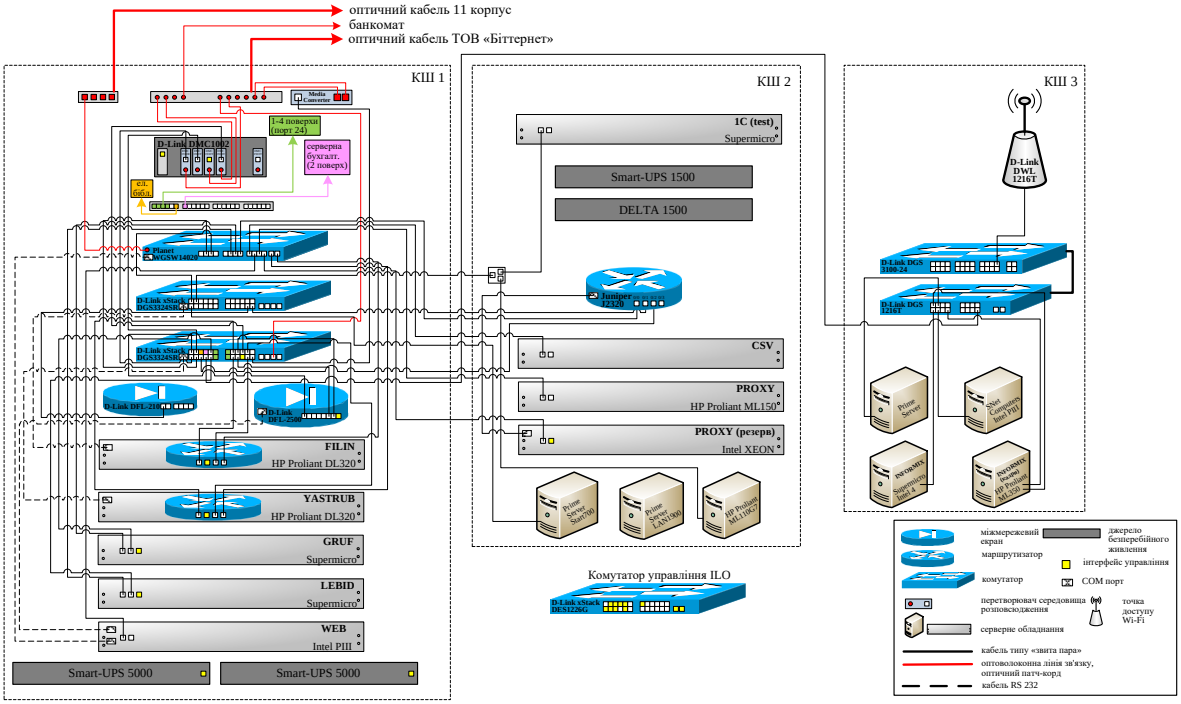

Рисунок 2.2 - Схема комутації активного мережевого, серверного обладнання та обладнання захисту ККМ університету серверного приміщення №1

Серверне приміщення 2 корпусу розміщено на першому поверсі 2 корпусу університету в одному з кабінетів. Апаратне забезпечення змонтовано в комутаційній шафі. Кабелі прокладено за допомогою утримувачів або навісним монтажем. Периметр приміщення не екрановано, доступ обмежено дерев’яними дверима з механічним замком. Пожежна сигналізація та система автоматичного пожежогасіння присутня. У якості засобу пожежогасіння використовується вуглекислотний вогнегасник.

Система контролю/підтримки кліматичних умов представлена кондиціонером типу спліт-система загальною охолоджуючою потужністю 3 кВт.

Охоронна сигналізація представлена об’ємними датчиками та системою контролю доступу. Приміщення ставиться на сигналізацію. Опечатування не здійснюється.

Також проводиться порядок фіксації робіт: журнал відкриття/закриття серверного приміщення, журнал реєстрації та обліку робіт на серверах тощо.

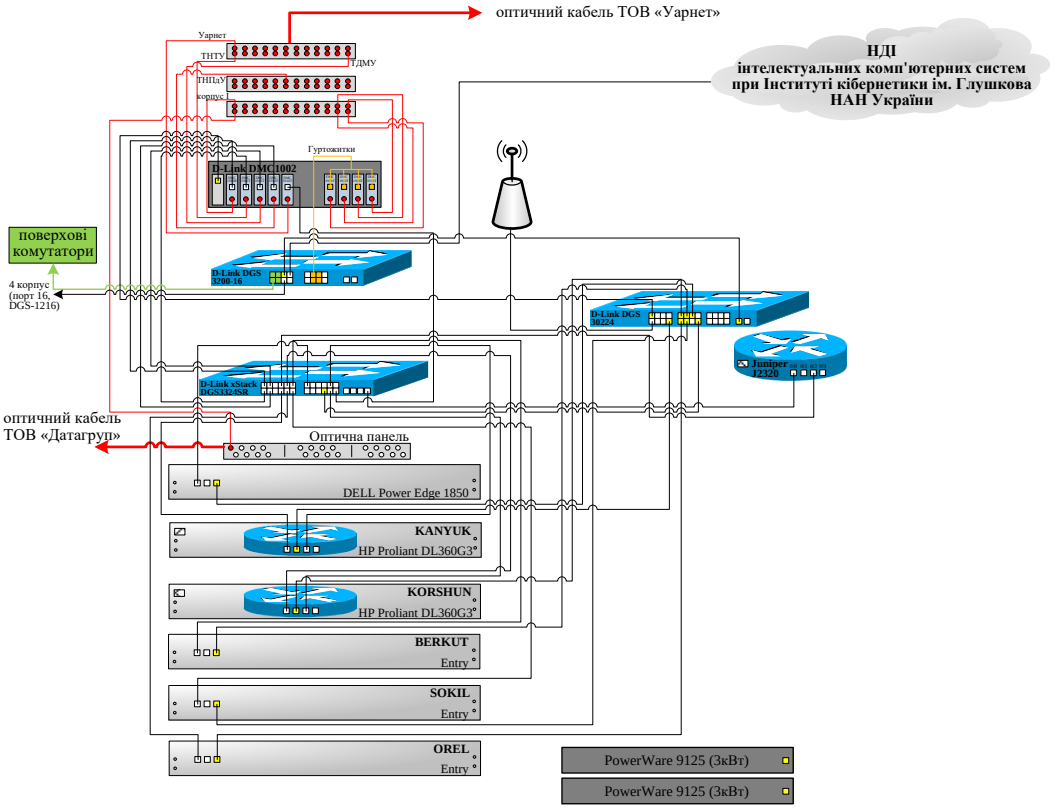

Рисунок 2.3 - Схема комутації активного мережевого, серверного обладнання та обладнання захисту ККМ університету серверного приміщення №2

Серверне приміщення автоматичної телефонної станції розміщено на першому поверсі 3 корпусу в окремому кабінеті. Кабелі прокладено за допомогою утримувачів або навісним монтажем. Периметр приміщення не екрановано, доступ обмежено броньованими дверима з механічним замком. Пожежна сигналізація та система автоматичного пожежогасіння присутні. Система контролю/підтримки кліматичних умов представлена кондиціонером типу спліт-система загальною охолоджуючою потужністю 3 кВт.

Охоронна сигналізація представлена об’ємними датчиками. Приміщення ставиться на сигналізацію. Опечатування не здійснюється.

Серверне приміщення бібліотеки розміщено на другому поверсі корпусу бібліотеки в кабінеті адміністратора локальної обчислювальної мережі бібліотеки (рис.2.5). Кабелі прокладено за допомогою утримувачів. Периметр приміщення не екрановано, доступ обмежено дерев’яними дверима з механічним замком. У якості засобу пожежогасіння використовується вуглекислотний вогнегасник.

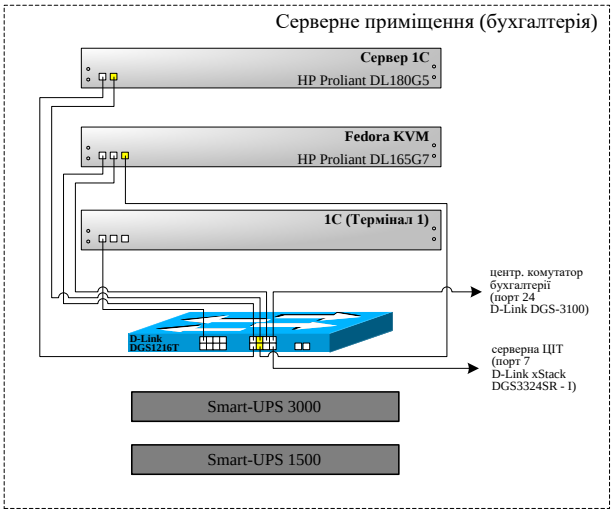

Рисунок 2.4 - Схема комутації активного мережевого, серверного обладнання та обладнання захисту ККМ університету серверного приміщення бухгалтерії

Система контролю/підтримки кліматичних умов представлена кондиціонером типу спліт-система загальною охолоджуючою потужністю 2 кВт. Охоронна сигналізація представлена об’ємними датчиками. Приміщення ставиться на сигналізацію.

Порядок фіксації робіт, що в ньому проводяться (журнал відкриття/закриття серверного приміщення, журнал реєстрації та обліку робіт на серверах» тощо), регламентовано.

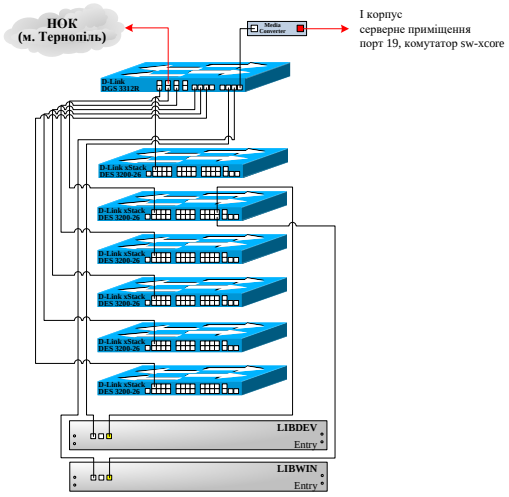

Рисунок 2.5 - Схема комутації активного мережевого, серверного обладнання та обладнання захисту ККМ університету серверного приміщення бібліотеки.

Окрім серверних кімнат до ККМ університету відноситься система відеоспостереження. Периметри всіх будівль обладнано відеокамерами, які виведено на пости охорони. Крім того, кожний поверх гуртожитку також обладнання системою відеоспостереження. Термін зберігання відеоархіву складає близько місяця.

Електроживлення будівель університету здійснюється від міської електромережі. Детальне вивчення системи електроживлення не проводилось.

Слід зазначити, що для забезпечення безперервності електроживлення, серверне та активне мережеве обладнання під’єднано (в тому числі за допомогою основного і резервного виходів) до блоків безперебійного живлення (ББЖ). Термін функціонування апаратного забезпечення від ББЖ складає близько 90 хв.

По результатам фізичного огляду та огляду конфігураційних файлів мережевого обладнання було розроблено схеми комутації активного мережевого, серверного обладнання та обладнання захисту ККМ університету для кожного з вище наведених серверних приміщень (додаток З).

Під час ознайомлення з конфігураційними файлами апаратного забезпечення було проаналізовано порядок побудови та функціонування локальних обчислювальних мереж які є складовими корпоративної комп'ютерної мережі (ККМ) університету, а також визначено механізми адресації, маршрутизації та сегментації, які використовуються.

ККМ університету є розгалуженою корпоративною мережею, побудовано за мережевою топологію «зірка». Мережа університету, за допомогою технології VLAN, віртуально розділена на близько 70 логічно відокремлених сегментів, завдяки чому зменшено широкомовний домен та мінімізовано загрозу виникнення такого негативного явища як «broadcast-шторм».

Крім того, також завдяки VLAN, інформаційні потоки таких підрозділів як відділ кадрів, відділ бухгалтерського обліку та звітності, а також технологічні інформаційні потоки (мережа управління) відокремлено від інших мережевих сегментів, що ліквідує можливість циркуляції інформації, яка містить персональні дані, фінансову, службову і технологічну інформацію (логіни, паролі) у всій ККМ.

В такій ситуації, отримання доступу до зазначених вище інформаційних потоків можливе лише у випадку отримання доступу елементів цих віртуальних мереж.

Окремо слід зазначити, що для керування апаратним забезпеченням у ККМ створено окремий мережевий сегмент 10.254.254.0/24, який, здебільшого, формується шляхом підключення менеджмент-інтерфейсів апаратного забезпечення ILO до комутаторів управління.

Університет має свою автономну систему AS50303 з адресацією 193.104.213.96/27. З'єднання (маршрутизація) з мережами операторів (провайдерів) телекомунікацій забезпечується використанням проколу BGP.

В якості граничних маршрутизаторів виступають 4 сервери (FILIN, KANYUK, KORSHUN, YASTRUB) з операційною системою Debian 12.0 та

програмним забезпеченням Quagga. Зазначені маршрутизатори отримують та зберігають повну таблицю маршрутизації всього Інтернету (Full view).

Маршрутизація між серверами-маршрутизаторами та апаратними маршрутизаторами Juniper J 2320 організована за допомогою протоколу iBGP.

Для внктрішньої маршрутизації в університеті використовується протокол OSPF. Маршрутизація між VLAN забезпечується стеком комутаторів рівня ядра.

Адресація у ККМ статична та для кожного з VLAN окрема (по префіксу /16) з мережевого діапазону 10.0.0.0/8.

На поточний момент ККМ університету має три оптоволоконні канали доступу до Інтернет:

- ТОВ «Колумбус» - 17 % трафіку, сусідня AS 16223, 250 Мбіт/с;

- ТОВ «Уарнет» - 43 % трафіку, сусідня AS 3255, 650 Мбіт/с;

- ТОВ «Датагруп» - 40 % трафіку, сусідня AS 21219, 400 Мбіт/с.

Окрім віртуальних мереж, які є локальними обчислювальними мережами для будівель та корпусів, розташованих у межах студмістечка ТНЕУ існують мережі:

- територіально відокремлених корпусів 5,6,7,8 та 9-го, що підключені до ІТС ЗУНУ за допомогою ліній зв'язку ТОВ «Колумбус», який забезпечує транспорт відповідних VLAN-ів (1035-1039);

- територіально відокремлених 12-ти філій, підключення яких до ІТС ТНЕУ забезпечується за допомогою апаратного міжмережевого екрану D-Link DFL 2500 та технології IPSec VPN. В даному випадку роль VPN-серверу виконує D-Link DFL 2500, а VPN-клієнтів – міжмережеві екрани D-Link DFL 210, які встановлені у філіях.

Необхідно додати, що за допомогою D-Link DFL 2500 також створюється адміністративний мережевий сегмент 10.254.254.0/24 та окремий сегмент для мережі бездротового доступу Wi-Fi 172.20.20.0/16.

За допомогою мережі взаємопов’язаних точок доступу D-Link DWL 3200AP та технології Wi-Fi, в межах студмістечка університету побудовано бездротову мережу з відкритим доступом.

За результатами проведеного аналізу розроблено схему адресації та маршрутизації (рис. 2.6) в інформаційно-телекомунікаційній системі корпоративної комп’ютерної мережі університету

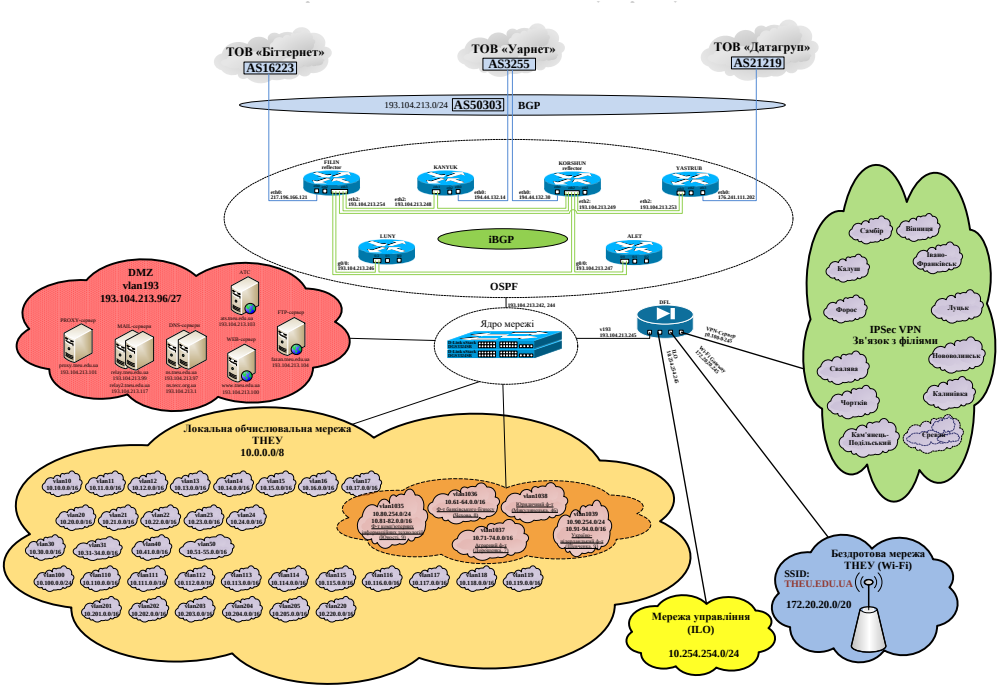

Рисунок 2.6 - Схема адресації та маршрутизації в інформаційно-телекомунікаційній системі корпоративної комп’ютерної мережі університету

Автоматизовані робочі місця співробітників університету (далі – АРМ) в робочі групи чи домен не об’єднано та являють собою окремо розташовані комп’ютери. Здебільшого, зазначені комп’ютери розташовано на кафедрах, у лабораторіях та електронних залах. Обов’язки щодо їх адміністрування покладено на інженерів кафедр.

Слід зазначити, що така ситуація призводить до неможливості централізованого керування мережею, впровадження політик інформаційної безпеки та розробки єдиних вимог із захисту інформації.

Як наслідок, політика безпеки (політика облікових записів, політика паролів, параметри безпеки) не впроваджена. Кожен співробітник на своєму АРМ має адміністративні привілеї, що дозволяє останнім встановлювати стороннє програмне забезпечення, вносити зміни в налаштування АРМ тощо. Як приклад, на АРМ співробітників університету відключені стандартні міжмережеві екрани (брандмауери Windows), а заборона на автоматичний запуск змінних носіїв інформації (USB-flash а ін.) відсутня.

Вимог до стійкості паролів, а також заборони щодо використання порожніх паролів не існує. У зв'язку із цим на більшості АРМ паролі не використовуються взагалі або являються простими. Зазначене також стосується і деяких паролів адміністраторів.

Крім того, невизначеність щодо політики та вимог з інформаційної безпеки призводить до випадків коли, наприклад, критичні автентифікаційні дані, конфігураційні файли, сертифікати та інша службова (технологічна інформація) зберігаються у відкритому вигляді на АРМ співробітників.

Оновлення операційних систем та іншого програмного забезпечення проводиться інженерами. А автоматичне встановлення оновлень для операційних систем, наприклад, за допомогою серверу WSUS, програмного забезпечення та програмних додатків не реалізовано.

Внаслідок цього на більшості АРМ (в тому числі в сегменті бухгалтерія та адміністративному сегменті) відсутні критичні оновлення безпеки,зокрема MS08-067 та MS12-020 (мал. 1 додатку 5).

Для забезпечення антивірусного захисту використовується ліцензійне програмне забезпечення Symantec Endpoint Protection та ESET NOD 32.

Оновлення останнього здійснюється з серверу Центру антивірусного захисту інформації. Оновлення для Symantec Endpoint Protection вручну розміщуються на локальному сервері оновлень.

Слід зазначити, що під час проведення сканування на вразливості були виявлені АРМ, бази сигнатур вірусів на котрих були не в актуальному стані або ті, на яких антивірусне програмне забезпечення було відсутнє взагалі.

Порядок доступу співробітників до Інтернет регламентується при допомозі PROXY-сервера. В корпоративній комп'ютерній мережі університету функціонує PROXY-сервер з програмним забезпеченням Squid та системою керування обліковими записами власної розробки фахівців.

Відповідно до налаштувань PROXY-серверу, в університеті впроваджена чітка політика щодо необхідності автентифікації користувачів перед доступом до Інтернет, обмеженої кількості дозволених протоколів (80, 443), заборонених програм/сервісів (можливість заборони ICQ, MSN), переліку сайтів, відвідування яких дозволено/заборонено, файлів, завантаження яких дозволено тощо.

Завдяки зазначеному загроза ураження шкідливим програмним забезпеченням обладнання та пристроїв університету зменшена.

Контроль підключень до всіх ЛОМ унівеситету (наприклад, за допомогою налаштування на комутаторах доступу опції port-security, фільтрації MAC-адрес, використання технології 802.1x) не здійснюється, що створює передумови для організації несанкціонованого під'єднання до ККМ.

Обмін електронними поштовими повідомленнями між співробітниками університету реалізовано за допомогою поштового серверу з програмним забезпеченням Exim (Dovecot). У якості клієнтського програмного забезпечення використовується Outlook Express.

Для отримання/відправки пошти використовуються протоколи POP3/SMTP, функціонал який не передбачає шифрування інформаційних потоків, в результаті чого інформація (в першу чергу логіни та паролі) передаються у відкритому вигляді (мал. 15 додатку 5).

Моніторинг працездатності активного мережевого та серверного обладнання, засобів захисту і каналів зв'язку впроваджено здійснюється за допомогою протоколу SNMP та програмного забезпечення HP Open View та Solar Winds, яке встановлено на двох серверах моніторингу.

Сканування серверного обладнання, АРМ та встановленого на них програмного забезпечення на предмет наявності вразливостей не здійснюється.

Для адміністрування серверного та активного мережевого обладнання здебільшого використовуються протоколи, які підтримують шифрування, а саме – ssh, https.

Разом з тим, виявлено випадки віддаленого адміністрування обладнання по протоколам HTTP та RDP, які вразливі до атак типу Man-in-the-Middle, Крім того, для локального адміністрування використовуються послідовні з'єднання – COM-порти.

Технологічні інформаційні потоки ізольовано від загальної мережі університету шляхом створення окремого мережевого сегменту.

Для забезпечення мережевого захисту, а саме – фільтрації інформаційних потоків та мережевих портів, на серверному обладнанні функціонує програмний міжмережевий екран IPTABLES. Детальний опис загроз, які можуть призвести до порушення функціонування мережі університету та порядку обробки інформації, що в ньому циркулює, моніторить та фільтрує пакети даний міжмережевий екран.

Також він захищає від сканування зі сторони мережі Інтернет вразливостей в DNS-сервері, щоб дозволило отримати інформацію стосовно структури ККМ університету.

За результатами оцінки та аналізу виявлено низку загроз, реалізація яких може призвести до порушення штатного режиму функціонування ККМ університету, а також конфіденційності, цілісності та доступності інформації, що в ньому циркулює.

2.2 Аналіз збоїв мережевого обладнання

З огляду на велику кількість комп'ютерних пристроїв та їх компонентів, під час роботи можуть виникати різні несправності, що здатні

спричинити збої мережевого обладнання. До основних причин належать проблеми з електроживленням, перегрівання обладнання, зависання пристроїв та порушення роботи каналів зв'язку.

Стабільне живлення є ключовою умовою коректного функціонування будь-якої комп'ютерної системи. Воно подає необхідну енергію для роботи всіх електронних вузлів: центрального процесора, накопичувачів, відеокарти, оперативної пам'яті та інших компонентів. Дотримання належних параметрів електропостачання забезпечує безперебійну роботу обладнання та допомагає уникнути його пошкодження або втрати даних.

До основних елементів системи живлення належать блок живлення (Power Supply Unit, PSU), стабілізатори напруги та джерела безперебійного живлення (Uninterruptible Power Supply, UPS).

Блок живлення виконує перетворення змінного струму з електромережі на постійний, подаючи його з відповідними параметрами на всі компоненти комп'ютера. Він оснащений різними роз'ємами та кабелями, що дозволяють живити материнську плату, відеокарту, накопичувачі, процесор та інші пристрої.

Стабілізатори напруги забезпечують вирівнювання та підтримання стабільного рівня напруги, що надходить до блока живлення комп'ютера. Їх завданням є убезпечити обладнання від стрибків чи падінь напруги, які можуть виникати через нестабільність електромережі.

Джерела безперебійного живлення (ББЖ, UPS) підтримують роботу комп'ютера у випадку раптового вимкнення електрики або короткочасних збоїв у мережі. Вони накопичують енергію у внутрішній батареї, яка активується за необхідності. Час автономної роботи UPS може відрізнятися, це може бути від кількох хвилин до годин - залежно від моделі та призначення пристрою.

Проблеми з електроживленням можуть мати різний характер і становити серйозну загрозу для роботи комп'ютерних систем. Неправильне підключення, наприклад некоректно вставлений кабель у розетку або в сам

комп’ютер, здатне спричинити збої в роботі обладнання. Різкі перепади в електромережі можуть викликати перегрів кабелів або розеток, що у крайньому випадку призводить до займання. Несправний чи пошкоджений блок живлення також може викликати відмову системи або, при перегріванні, стати причиною пожежі.

Перевантаження електромережі виникає тоді, коли до одного кола або розетки підключено надто багато пристроїв. Це створює ризик пошкодження обладнання, втрати даних або навіть займання. Нестабільне електроживлення, до якого призводять: перевантаження лінії, раптові відключення чи просідання напруги здатні викликати некоректну роботу електроніки або її повну відмову.

Кабелі живлення можуть виходити з ладу через механічні ушкодження, зношування чи неправильне використання. Пошкоджений кабель часто стає причиною переривання подачі живлення та супутніх збоїв у роботі всієї системи.

Пристрої комп’ютерних мереж можуть працювати нестабільно через порушення в електроживленні. Йдеться про просідання напруги або її різкі стрибки, що часто виникають під час гроз, перевантаження електромережі чи з інших зовнішніх причин. Такі зміни здатні пошкоджувати або тимчасово виводити з ладу компоненти мережевих систем, зокрема мікропроцесори, накопичувачі та іншу електроніку.

Перегрівання обладнання є ще однією поширеною та небезпечною проблемою, яка негативно впливає як на продуктивність, так і на тривалість роботи пристроїв. Підвищення температури може виникати через слабку вентиляцію корпусу, забруднення систем охолодження, надмірне навантаження на апаратні компоненти, неякісний тепловий контакт або неправильне розміщення обладнання.

На роботу мережевого обладнання суттєво впливають умови навколишнього середовища. Висока вологість, різкі перепади температури, пил, вібрації чи надмірна щільність розміщення обладнання можуть сприяти

виникненню коротких замикань, корозії контактів або механічних пошкоджень, що в результаті порушує роботу мережі.

Недостатній повітрообмін призводить до накопичення тепла всередині мережевих пристроїв. Це трапляється, коли корпус має мало вентиляційних отворів, конструкція не забезпечує належну циркуляцію або коли потік повітря перекритий сторонніми предметами. У таких умовах найбільше страждають компоненти з високим тепловиділенням, зокрема центральний процесор і відеокарта.

Забруднені пилом вентилятори також знижують ефективність охолодження. Пил осідає на лопатях і корпусах вентиляторів, уповільнює їх роботу та зменшує обсяг повітря, що проходить через систему. Як наслідок температура всередині пристрою підвищується, а окремі компоненти починають перегріватися.

Перевантаження апаратних компонентів теж може спричиняти підвищене нагрівання. Виконання ресурсомістких операцій збільшує температуру процесора, графічного модуля та інших вузлів, що у разі недостатнього охолодження веде до зниження швидкодії або до перегріву.

Неправильне встановлення елементів тепловідведення також є поширеною причиною підвищення температури. Якщо радіатори чи інші частини системи охолодження погано прилягають до компонентів, тепло відводиться неефективно, що спричиняє локальний перегрів.

На температуру впливає й розташування обладнання. Якщо пристрій стоїть у замкненому, тісному чи непровітрюваному місці, тепло від нього відводиться значно повільніше, що сприяє поступовому нагріванню корпусу та його внутрішніх частин.

Щоб уникнути перегріву, необхідно регулярно обслуговувати мережеве обладнання, стежити за чистотою систем охолодження, забезпечувати адекватну вентиляцію та не допускати надмірного навантаження на компоненти. Також важливо розміщувати пристрої у добре провітрюваних місцях і не перекривати вентиляційні отвори.

З часом апаратні компоненти мережевих пристроїв природно зношуються. Елементи живлення, конденсатори, мікросхеми та навіть порти комунікацій поступово втрачають свої характеристики, що призводить до нестабільної роботи, періодичних збоїв або повної відмови пристрою. Така деградація не завжди проявляється одразу й часто супроводжується спорадичними помилками, які важко діагностувати.

Зависання обладнання - це стан, коли операційна система або встановлене програмне забезпечення перестає реагувати на дії користувача і фактично «завмирає». Подібні ситуації можуть виникати з різних причин і проявлятися по-різному: від короткочасної зупинки окремої програми до повного блокування роботи системи.

Однією з найпоширеніших причин зависання мережевого обладнання є надмірне використання системних ресурсів. Коли програми активно навантажують процесор, оперативну пам’ять чи диск, система може не встигати обробляти всі запити, що призводить до її уповільнення або повної зупинки. Наприклад, одночасний запуск великої кількості ресурсоємних програм або виконання складних обчислень нерідко викликає зависання.

Ще однією причиною можуть бути помилки чи недоліки програмного забезпечення. Ненадійно реалізовані програми або програми з вадами у коді інколи працюють некоректно, некоректно керують пам’яттю або неправильно використовують ресурси системи, що у підсумку також спричиняє зависання.

Також зависання може виникати у разі конфліктів між програмами чи пристроями. Якщо кілька додатків або апаратних компонентів намагаються одночасно працювати з одними й тими самими ресурсами, система може заблокуватися або перестати відповідати на команди. Окрім цього, причиною нестабільності можуть бути неправильні налаштування операційної системи, шкідливе програмне забезпечення чи віруси, а також апаратні несправності, зокрема дефектні компоненти. Застаріла або некоректно встановлена версія прошивки може викликати несумісність із новим обладнанням чи

протоколами, що призводить до збоїв у передачі даних. Неправильне або перерване під час оновлення прошивання також може повністю вивести пристрій з ладу.

Збої каналів зв'язку також можуть виникати з багатьох причин і проявлятися по-різному. Основні серед них: переривання зв'язку, інтерференція та шуми, помилки в налаштуваннях мережі, проблеми із з'єднанням, відмова обладнання й несправності програмного забезпечення. Їх аналіз дозволяє оцінити важливість стабільної роботи каналів передачі даних.

Переривання зв'язку між мережевими пристроями зазвичай пов'язане з пошкодженням кабелів, несправними роз'ємами або відмовою обладнання. Такі ситуації часто спричиняють короткочасну або повну втрату з'єднання між мережевими вузлами: комп'ютерами, серверами чи іншими пристроями.

Інтерференція та шуми в каналах зв'язку можуть виникати через електромагнітні впливи або роботу інших пристроїв поблизу. У таких випадках сигнал спотворюється, дані передаються з помилками, що призводить до втрати пакетів та зменшення швидкості передачі.

Помилки у конфігурації мережевого обладнання або програмного забезпечення теж часто стають джерелом проблем із підключенням. Серед найбільш поширених: неправильно введені IP-адреси чи маски підмережі, некоректні налаштування шлюзу за замовчуванням, що унеможливлює доступ до інших пристроїв; помилки в налаштуваннях DNS, через які доменні імена не перетворюються на IP-адреси; неправильна конфігурація мережевого обладнання, що призводить до збоїв у роботі й ускладнює встановлення з'єднання.

Проблеми зі з'єднанням також можуть бути наслідком ненадійно під'єднаних або пошкоджених роз'ємів Ethernet чи інших мережевих кабелів. Подібні дефекти спричиняють періодичні розриви зв'язку або повну неспроможність встановити стабільне підключення.

Відмова мережевих пристроїв: маршрутизаторів, комутаторів чи мережевих адаптерів також може стати причиною порушень у роботі каналів

зв’язку. Наприклад, якщо виходить з ладу маршрутизатор, усі пристрої, підключені до нього, автоматично втрачають доступ до мережі.

Людські помилки також часто стають причиною збоїв каналів зв’язку. Сюди належать некоректне налаштування обладнання, неправильне введення конфігураційних параметрів, випадкове відключення кабелів або встановлення невідповідних версій програмного забезпечення. Такі помилки можуть тимчасово або повністю порушити роботу мережі.

Несправності або некоректна робота мережевого програмного забезпечення також часто призводять до проблем зі зв’язком. Це може статися, коли програми для аналізу чи моніторингу мережі конфліктують з іншими застосунками або обладнанням, що в підсумку спричиняє збої у передачі даних.

З огляду на можливі ризики і наслідки таких проблем, важливо вчасно вживати заходів для їх запобігання та усунення. Одним із дієвих підходів до підвищення стабільності роботи мережевого обладнання є використання сучасних джерел безперебійного живлення, які допомагають уникнути наслідків раптових збоїв електропостачання.

2.3 Проектування засобів моніторингу збоїв комп’ютерних мереж

Основою системи контролю стану і керуванням мережевим обладнанням буде виступати стандартний SNMP-менеджер HP OpenView. Це відома платформа для моніторингу та адміністрування корпоративних і великих комп’ютерних мереж.

Для реалізації представлення пристроїв мережі необхідно спроектувати представлення параметрів URL-адреси мережевого репрезентатора (Network Presenter). Network Presenter – це версія компонента відображення OpenView Window на базі Java. Він забезпечує кілька переглядів даних карти на сервері керування або консолі керування з будь-якої машини з віртуальною машиною Java.

У цьому випадку виправлення до Network Presenter додано два додаткові параметри URL-адреси. Тепер можна вказати назву або тип підкарти, з якої об’єкт має бути відкритий у Network Presenter. Але спочатку потрібно визначити типи запуску Network Presenter. Його можна запустити двома способами: за допомогою URL-адреси та OVLauncher.

Ці методи детально пояснюються в багатьох розділах офіційної документації HP. Так для запуску Network Presenter за допомогою URL-адреси в браузері необхідно ввести наступну URL-адресу в адресний рядок:

http://hostname[:port]/OvCgi/jovw.exe?[MapName=<mapname>&ObjectName=<selectionname> & ServerName=<servername>(PreferredSubmapType=<submaptype> | SubmapName=<submapname>)]

У системі UNIX необхідно ввести номер порту 3443. У системі Windows номер порту не потрібен. MapName – назва карти, до якої потрібно підключитися Network Presenter. ObjectName – назва вибору об’єкта, що відображається спочатку. ServerName – назва станції керування.

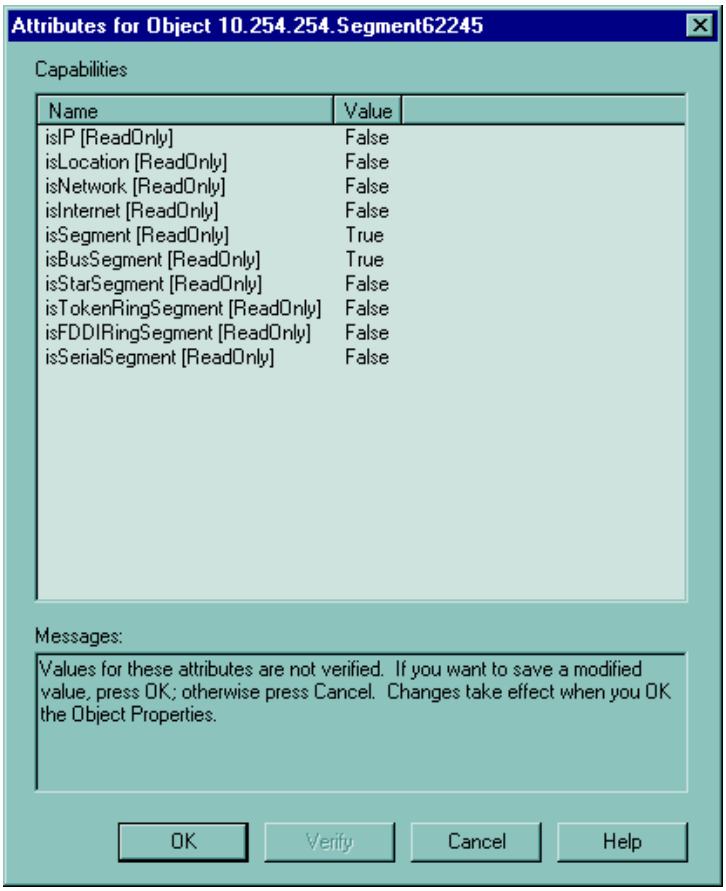

Рисунок 2.7 – Вигляд сегменту типу шина

Додано два нові параметри URL-адреси: SubmapName і PreferredSubmapType. SubmapName – для відкриття об’єкта з певної підкарти. Параметр SubmapName не можна використовувати з параметром PreferredSubmapType. Ці параметри є взаємовиключними. Крім того, параметр SubmapName не працюватиме, якщо не вказати ObjectName в URL-адресі:

http://hostname[:port]/OvCgi/jovw.exe?[MapName=<mapname>&ObjectName=<selectionname> & SubmapName=<submapname>)]

PreferredSubmapType – для відкриття об’єкта з указаного типу підкарти. Параметр PreferredSubmapType не можна використовувати з SubmapName, як зазначено вище. Також параметр PreferredSubmapType не працюватиме, якщо не вказати параметр ObjectName. Параметр PreferredSubmapType може мати наступні значення:

- isInternet
- isNetwork
- isSegment
- isLocation
- isUnknowContainer

Підкарта, яка відображається для параметра ObjectName базується на значенні, яке встановлене для PreferredSubmapType. Значення параметра PreferredSubmapType описані наступним чином.

Якщо це значення вказано як isNetwork, тоді тип «Мережа» відображається в Network Presenter, якщо підкарта містить символ для ObjectName. Якщо більше однієї підкарти «Мережа» містить такий символ, тоді відображається перша підкарта «Мережа», яка містить цей символ.

http://hostname[:port]/OvCgi/jovw.exe?[MapName=<mapname>&ObjectName=<selectionname> &(PreferredSubmapType=isNetwork)]

isInternet – це значення, вказане для типу PreferredSubmapType і означає «Internet» і відображається в Network Presenter, якщо підкарта містить символьне значення для ObjectName.

http://hostname[:port]/OvCgi/jovw.exe?[MapName=&ObjectName=
<selectionname>&(PreferredSubmapType=isInternet)]

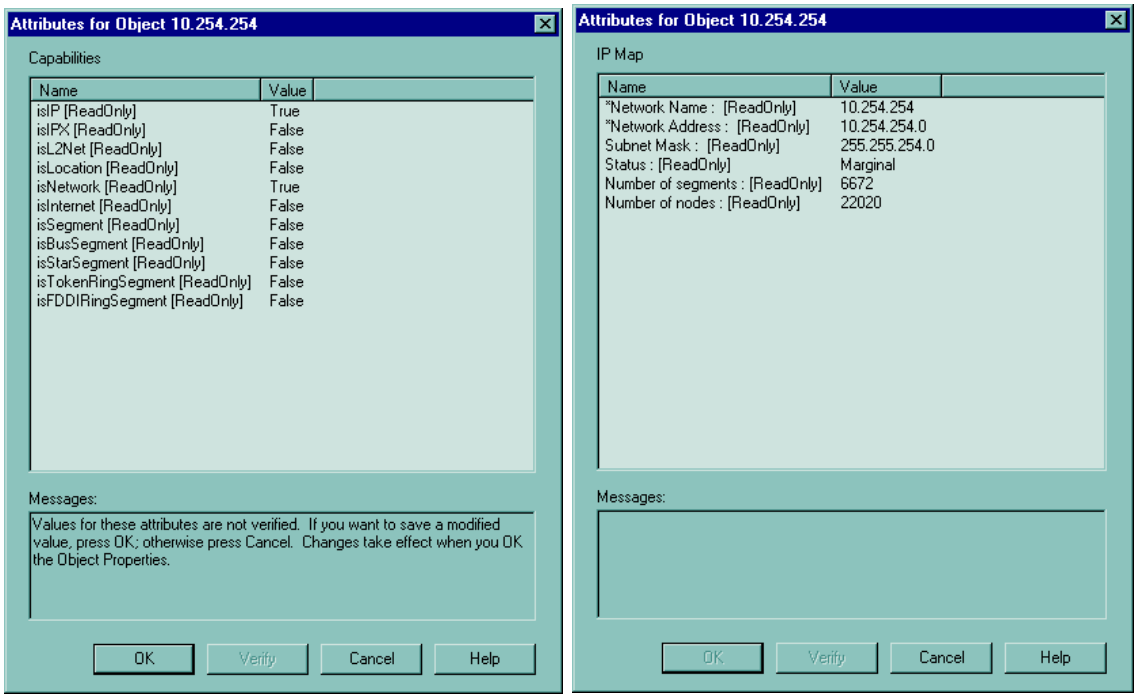

Рисунок 2.8 – Вигляд IP-підмережі типу isNetwork

isLocation – це значення, вказане для типу розміщення. «Location» відображається в Network Presenter, якщо підкарта містить символ для ObjectName. Якщо такий символ містить більше однієї підкарти «Location», відображається перша підкарта «Location», яка містить цей символ.

http://hostname[:port]/OvCgi/jovw.exe?[MapName=&ObjectName=
<selectionname>&(PreferredSubmapType=isLocation)]

Якщо вказано значення як isSegment для типу PreferredSubmapType, «Segment» відображається в Network Presenter, якщо підкарта містить символ для ObjectName. Якщо більше однієї підкарти «Segment» містить такий символ, відображається перша підкарта «Segment», яка містить цей символ.

http://hostname[:port]/OvCgi/jovw.exe?[MapName=>&ObjectName= <selectionname>
&(PreferredSubmapType=isSegment)]

При значеннях isUnknowContainedSubmapType відображається підкарта контейнера. Підкарти контейнерів не мають жодних атрибутів в ovwdb (наприклад, атрибутів, таких як помилки) Якщо це значення вказано для

PreferrisNetwork, isInternet, isLocation або isSegment). Якщо символ ObjectName існує в кількох таких підкартах, відображається перша підкарта, яка містить цей символ

http://hostname[:port]/OvCgi/jovw.exe?[MapName=&ObjectName=
<selectionname>&(PreferredSubmapType=isUnknownContainer)]

Щоб запустити Network Presenter за допомогою OVLauncher (рис.2.9), виконується одна із наступних дій. У вікні OVLauncher переходять на вкладку Об'єкти та виберають Мережі IP. Або у вікні OVLauncher переходять на вкладку Мережеві інструменти та виберають NNM Network Presenter.

Також необхідно описати, де та чому можна та потрібно використовувати інструмент BatchPing.ovpl.

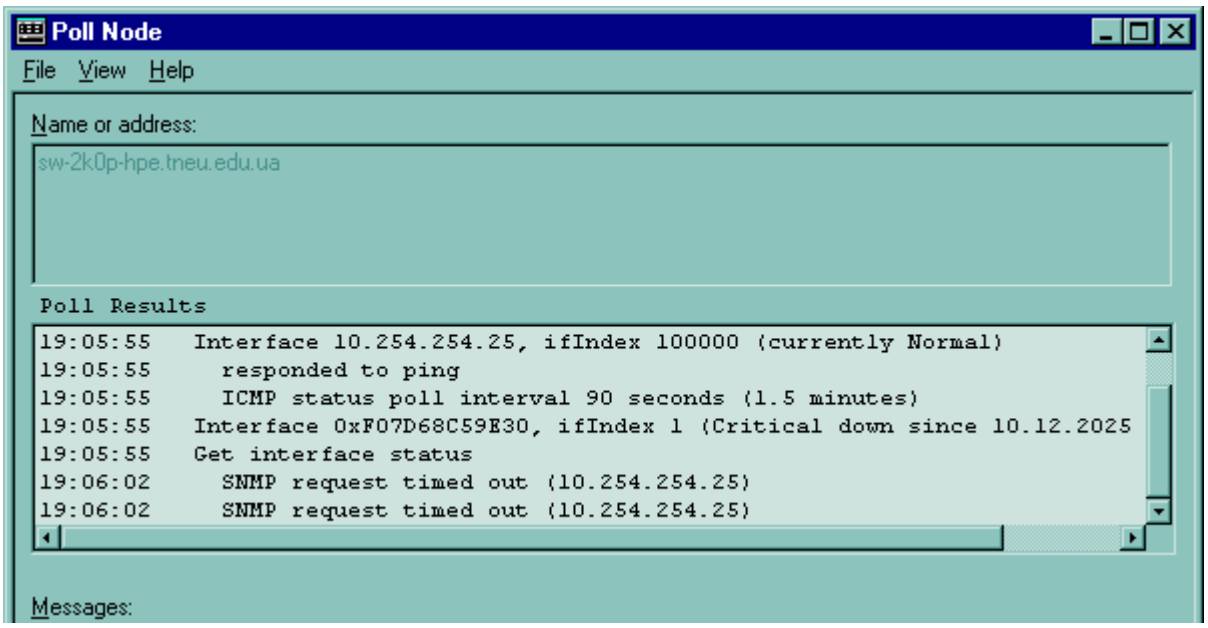

Рисунок 2.9 – Запуск інструменту BatchPing.ovpl

Цей інструмент вимагає PERL і наразі працює на HP-UX, Solaris, Linux, та Windows. Мета цього інструменту — допомогти заповнити таблиці бази даних переадресації комутаторів Ethernet, якими активно керує Network Node Manager Extended Topology (Extended Topology). Заповнюючи FdbTable комутаторів MAC-адресами сусідніх комутаторів, дані виявлення Extended Topology будуть більш повними.

Інструмент BatchPing.ovpl досягає цього, надсилаючи ICMP Echo Requests (ping) комутаторам, що цікавлять. Цей інструмент є найбільш ефективним для комутаторів, які не використовують протокол комутатор-комутатор, такий як CDP від Cisco, що призведе до заповнення FdbTable. За наявності CDP від Cisco або подібного протоколу цей інструмент не повинен бути потрібен.

Щоб ефективно використовувати інструмент BatchPing.ovpl напочатку визначається список IP-адрес адрес або імена хостів у файлі з одним записом на рядок комутаторів, які ви потрібно виявити за допомогою розширеної топології (рис. 2.10). Цей файл використовується інструментом як цільовий список для ping.

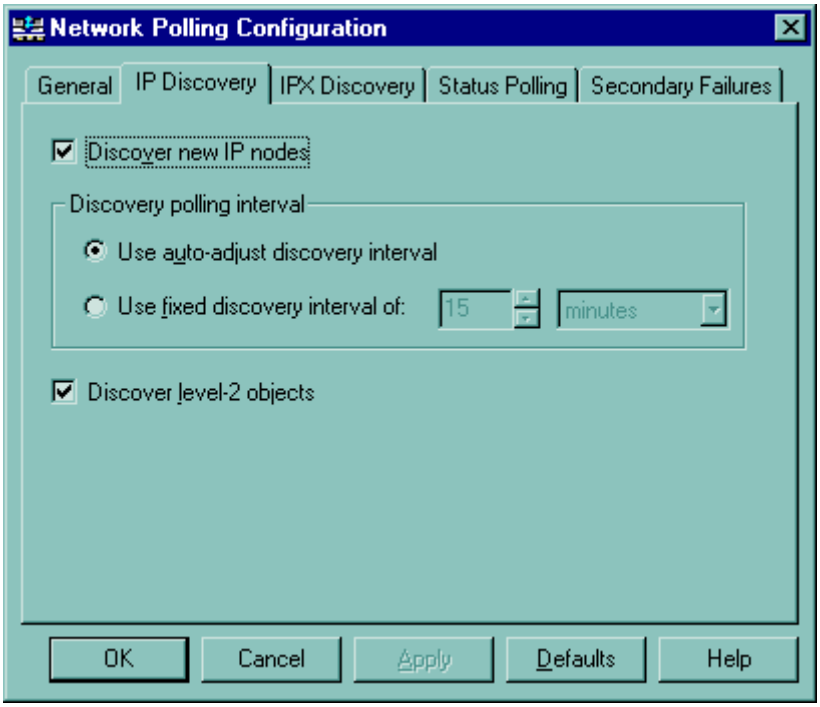

Рисунок 2.10 – Розкриття топології мережі

Далі розгортається цей інструмент на зовнішньому краю мережі, якомога далі від станції керування розширеною топологією. Потім планується запуск цього інструменту не більше ніж за 15 хвилин до початку кожного етапу виявлення.

Ресурси, що споживаються цим інструментом, достатньо малі, щоб запускати його кожні 10 або 15 хвилин через cron або інший процес планування. Якщо розгорнути BatchPing.ovpl у кількох віддалених місцях, які використовують інший мережевий шлях для трафіку ping, ніж станція керування розширеною топологією, то можна побачити ефективніші результати.

Кожному екземпляру інструменту потрібно пропінгувати лише комутатори у своєму регіоні мережі. Для спрощення налаштування можна використовувати один цільовий файл для всіх екземплярів BatchPing.ovpl.

Використовуються наступні оператори інструменту BatchPing.ovpl:

- кількість процесів
- файл списку хостів
- перелік хостів

Так можна задати оператором –p кількість дозволених одночасних процесів ping. за замовчуванням їх 4, оператор -h вимагає "перелік хостів у лапках", а оператором -f задаємо імя файлу, що містить список хостів, записаних один на рядок. Також при допомозі оператора -d можна увімкнути повне відлагодження програмного інструменту із командного рядка. Типовий командний рядок:

BatchPing.ovpl -p <number of processes> -f <host file> || -h "hosts"

Таким чином даний інструмент дає нам можливість швидше розкривати і моніторити обліднання та топологію комп’ютерної мережі не використовуючи занадто великі ресурси.

3 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ КЕРУВАННЯ ОБЛАДНАННЯМ КОМП'ЮТЕРНИХ МЕРЕЖ

3.1 Функціональна схема програмного забезпечення

Для системи керування обладнанням комп'ютерної мережі, що базується на системі HP OpenView, створено програмні модулі мовами Python та Perl. Вони виконують ролі SNMP-агентів, обмінюється даними з SNMP-менеджером і, у разі відсутності з'єднання, ініціюють процедуру перевірки працездатності мережевого обладнання та за потреби здійснюють перемикання його електроживлення.

Схему взаємодії між менеджером та агентами наведено на рисунку 3.1.

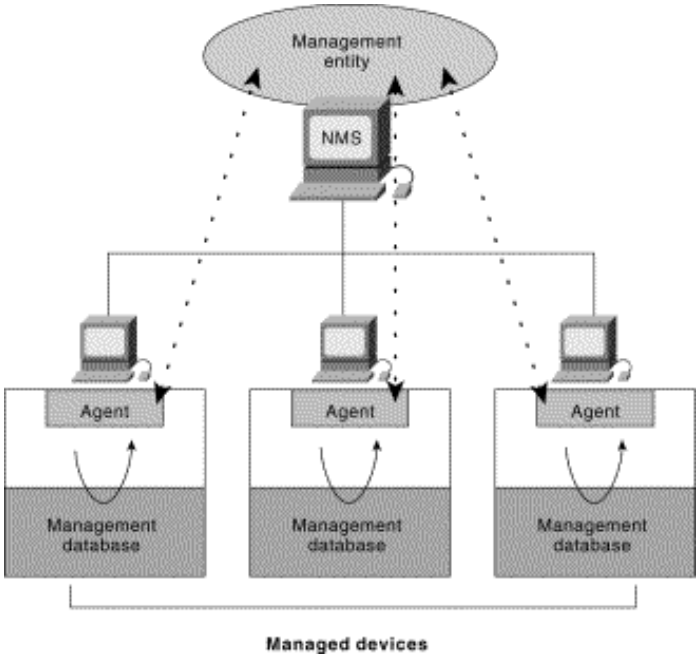

Рисунок 3.1 – Схема функціональної взаємодії менеджера SNMP та агентів

HP OpenView є потужним інструментом для управління IT-інфраструктурою підприємств різних масштабів і сфер діяльності. Система має модульну архітектуру та забезпечує широкі можливості моніторингу й керування локальними мережами та серверними платформами, такими як HP-UX, Solaris, AIX, Novell, Linux та різні версії Windows. Топологія

корпоративної комп'ютерної мережі університету на IP-рівні представлено на рисунку 3.2.

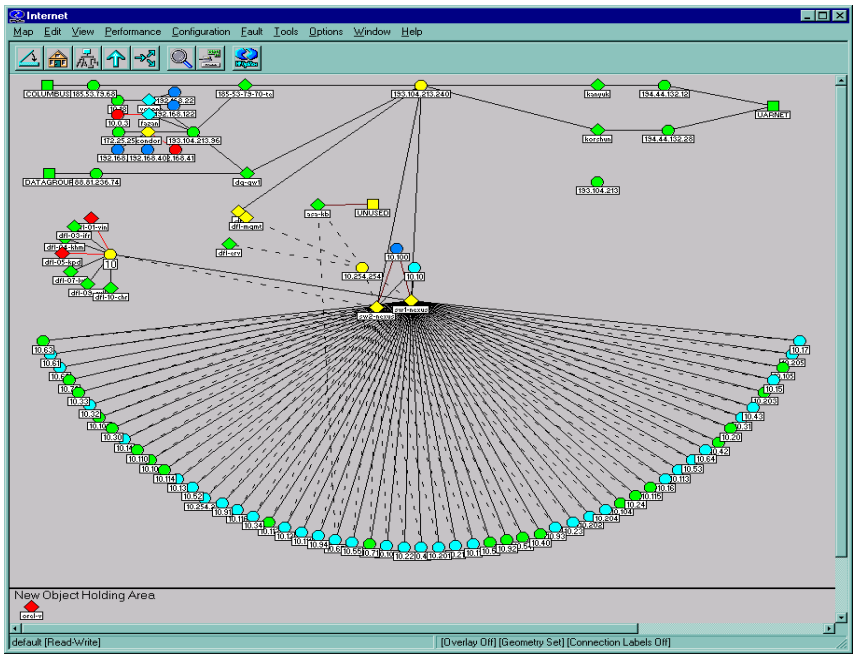

Рисунок 3.2 – ККМ університету на HP OpenView

Система SNMP-менеджера також дає змогу керувати різноманітними застосунками, серед яких SAP, Oracle, Sybase, MS SQL, Exchange, DB2, Informix та MS Active Directory.

Приклад топології мережі комутаторів представлено на рисунку 3.3.

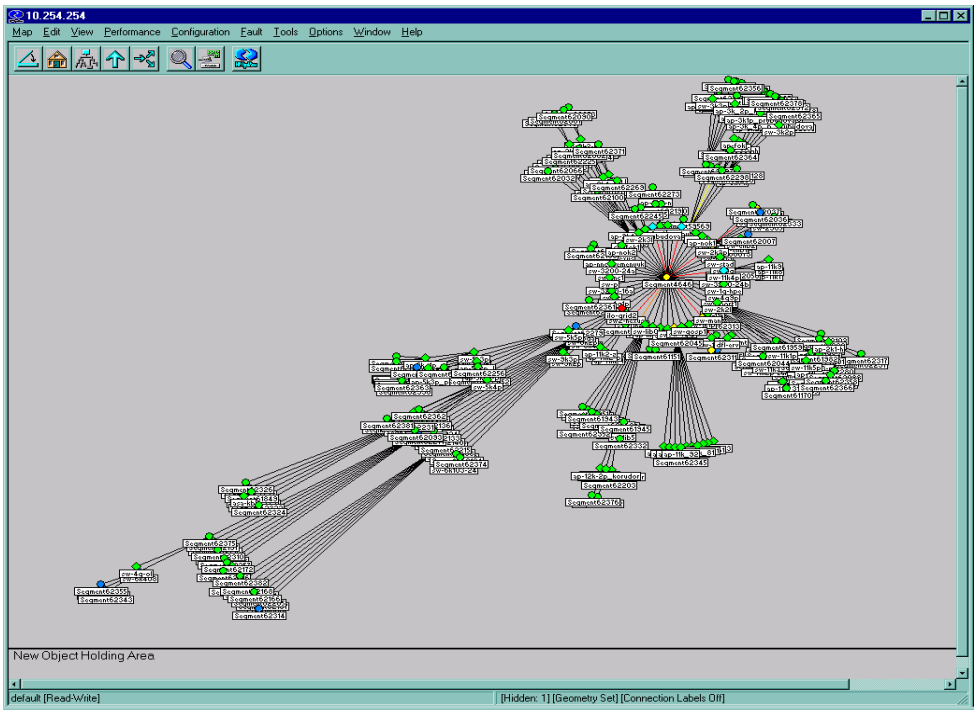

Рисунок 3.3 – Моніторинг мережі комутаторів в HP OpenView

Крім того, вона забезпечує роботу з користувацькими робочими станціями:

- веде інвентаризацію;
- дозволяє віддалено встановлювати операційні системи;
- оновлення й програмне забезпечення, створювати облікові записи;
- контролювати використання установлених програм;

Платформа також підтримує функціонування диспетчерської служби та містить інструменти для побудови IT-інфраструктури відповідно до принципів ITIL та ITSM. Загалом у її складі доступно понад п’ятдесят програмних продуктів, що охоплюють широкий спектр задач- від резервного копіювання до моніторингу бізнес-процесів у режимі реального часу. Основні процеси системи NMS OpenView представлені на рисунку 3.4.

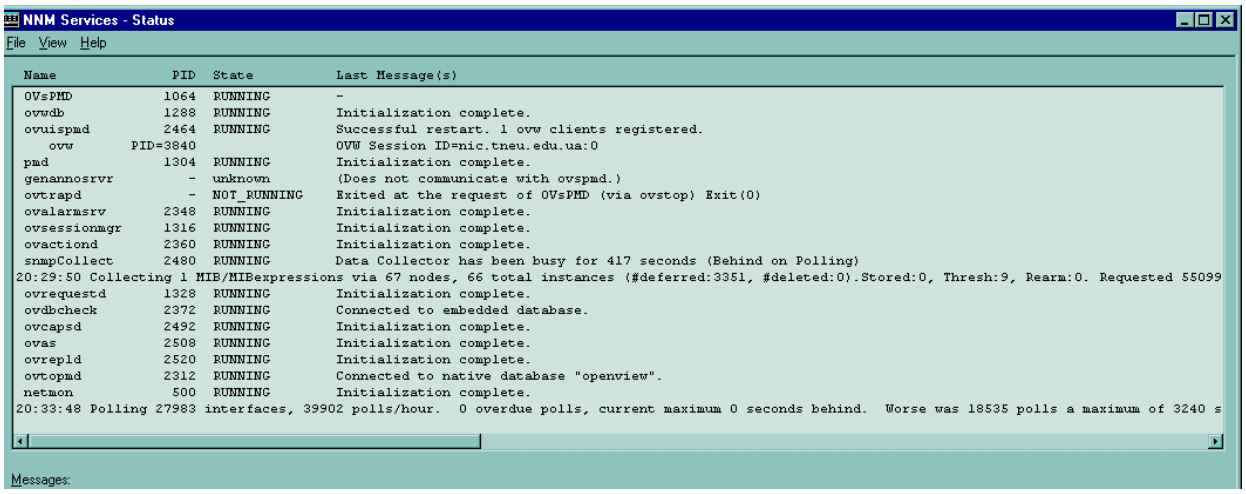

Рисунок 3.4 - Основні процеси системи NMS Openview

SNMP-менеджер виконує роль центральної системи, яка відповідає за моніторинг мережі SNMP. Його також називають станцією керування мережею (Network Management System, NMS). Він підтримує зв’язок із мережевими пристроями, на яких працюють SNMP-агенти, і функціонує на одному з вузлів мережі. Менеджер надсилає агентам запити, отримує їхні відповіді, змінює необхідні параметри та обробляє сповіщення, що надходять від агентів.

По суті, SNMP-менеджер - це програмний комплекс, який використовує протокол SNMP для отримання даних, потрібних для виявлення несправностей, оцінки продуктивності та планування подальшого розвитку мережевих ресурсів.

Керований мережевий пристрій (Managed Network Device, MND) - це будь-який мережевий компонент із підтримкою протоколу SNMP, яким управляє SNMP-менеджер. До таких пристроїв належать маршрутизатори, комутатори, концентратори, мости, повторювачі, шлюзи, сервери, брандмауери, принтери та точки бездротового доступу.

SNMP-агент є програмним модулем, що забезпечує збирання та передачу даних про стан і роботу мережевого обладнання. Він реагує на запити менеджера SNMP, надає інформацію щодо поточного стану вузла та зберігає необхідні статистичні дані. Агент працює безпосередньо на пристрої, за яким здійснюється моніторинг, і передає отриману інформацію менеджеру SNMP (рисунок 3.5).

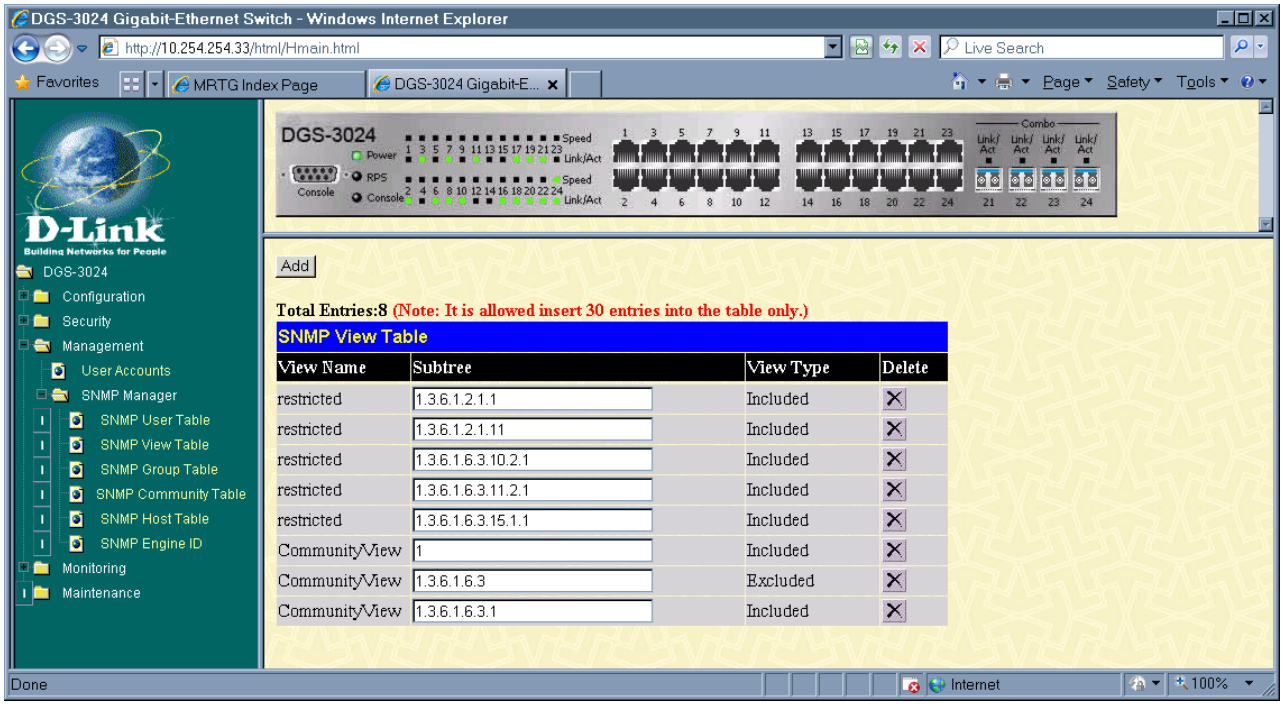

Рисунок 3.5 – Вбудований в комутатор SNMP-агент

Менеджер SNMP може періодично надсилати запити до всіх мережевих пристроїв для контролю їхнього стану. Водночас агенти SNMP

можна налаштовувати так, щоб вони самостійно відправляли повідомлення у разі зміни стану пристрою. Такі повідомлення отримали назву *пастки* (traps).

Пастки SNMP формуються у певному форматі, що містить інформацію про час події, ідентифікатор та значення. Час фіксує момент виникнення події, а ідентифікатор, що походить із MIB, називається OID. Порт, на який менеджер отримує такі повідомлення, зазвичай має номер 162, проте його можна змінити за необхідності.

Однак існують певні обмеження при використанні пасток SNMP. Вони не завжди надійно сповіщають про серйозні проблеми. Іноді агент може надіслати повідомлення про незначну несправність і пропустити критичну ситуацію, яка здатна вивести мережу з ладу. Наприклад, якщо на пристрої відбувається фатальна помилка, що вимикає його повністю, агент SNMP також перестає працювати, і відповідне сповіщення не надсилається.

У таких випадках, коли мережевий пристрій зависає або працює некоректно, критично важливо мати механізм контролю його стану. Для цього наш контролер стану, оснащений власним агентом SNMP, може перезавантажити керований пристрій. Крім того, цей контролер підходить для пристроїв, які не мають вбудованого SNMP, забезпечуючи надійне управління та моніторинг у таких ситуаціях.

3.2 Керування інформаційною базою даних MIB

MIB є ключовим елементом у моделях управління мережею. Вона визначає структуру обміну інформацією в системі SNMP. Кожен SNMP-агент містить власну інформаційну базу даних, яка описує параметри керованого мережевого пристрою. Менеджер SNMP зберігає отримані дані у MIB, утворюючи спільну базу між агентом і менеджером, що забезпечує централізоване управління та моніторинг мережевих об’єктів.

Файли MIB зберігаються у текстовому форматі, що підтримується редакторами MIB, конструкторами SNMP-агентів, інструментами управління мережею та мережевими симуляторами. Така універсальність спрощує

процес розробки, тестування, розгортання та адміністрування мережі. Керовані об’єкти у файлах MIB ідентифікуються за допомогою унікальних ідентифікаторів об’єктів (OID).

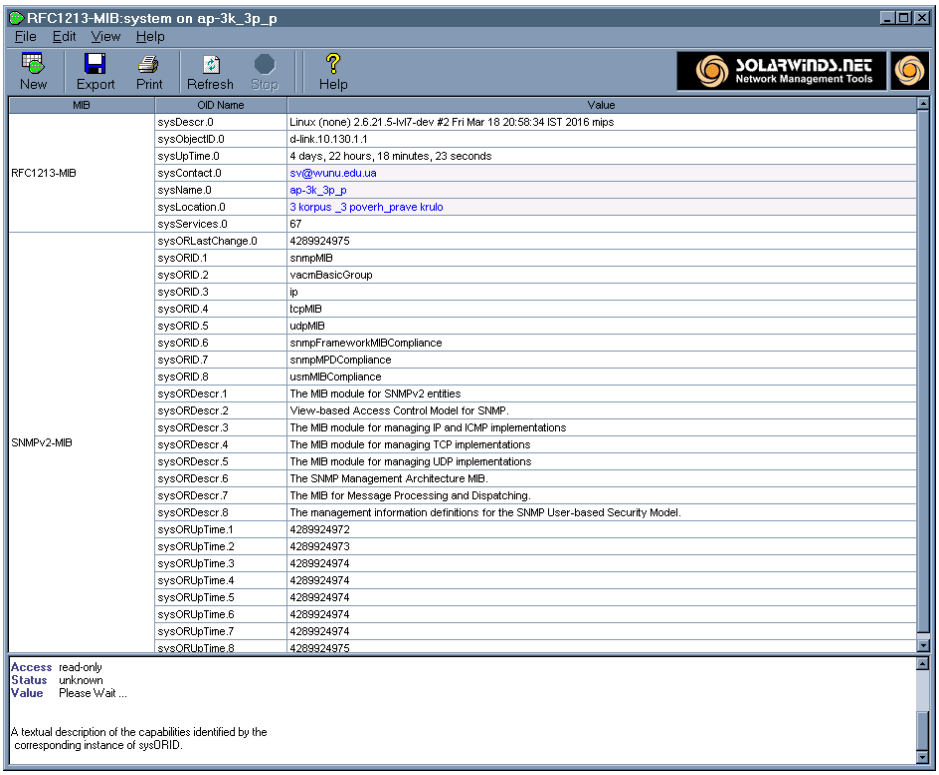

Рисунок 3.6 – Структура керованої інформаційної бази даних MIB

MIB організовані у вигляді дерева OID з ієрархічною структурою (рис.3.6), яка відображає всі керовані функції пристроїв. Кожна гілка дерева має унікальний номер і назву, а кожна точка відповідає конкретному об’єкту та отримує власний унікальний ідентифікатор, який складається з повного шляху від кореня дерева до цієї точки. OID позначає елемент пристрою, що контролюється, наприклад температуру, стан центрального процесора або пам’яті, а також рівень витратних матеріалів, наприклад чорнила в принтері.

SNMP OID можна ідентифікувати за допомогою числових рядків, розділених крапками. Структура дерева MIB представлена на рисунку 3.7.

У системі SNMP виділяють два типи керованих об’єктів. Перший тип — скалярні об’єкти, які мають лише один екземпляр, тобто кожен об’єкт може мати лише одне значення. Другий тип - табличні об’єкти, що

складаються з кількох взаємопов'язаних екземплярів, організованих у вигляді таблиці MIB.

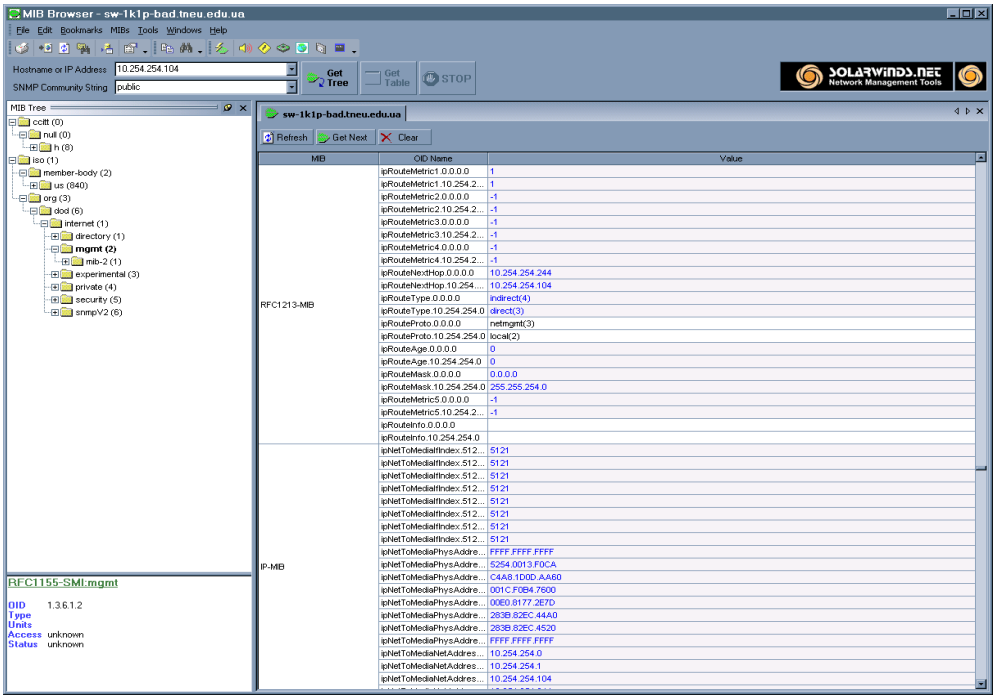

Рисунок 3.7 – Організація MIB у вигляді дерева OID

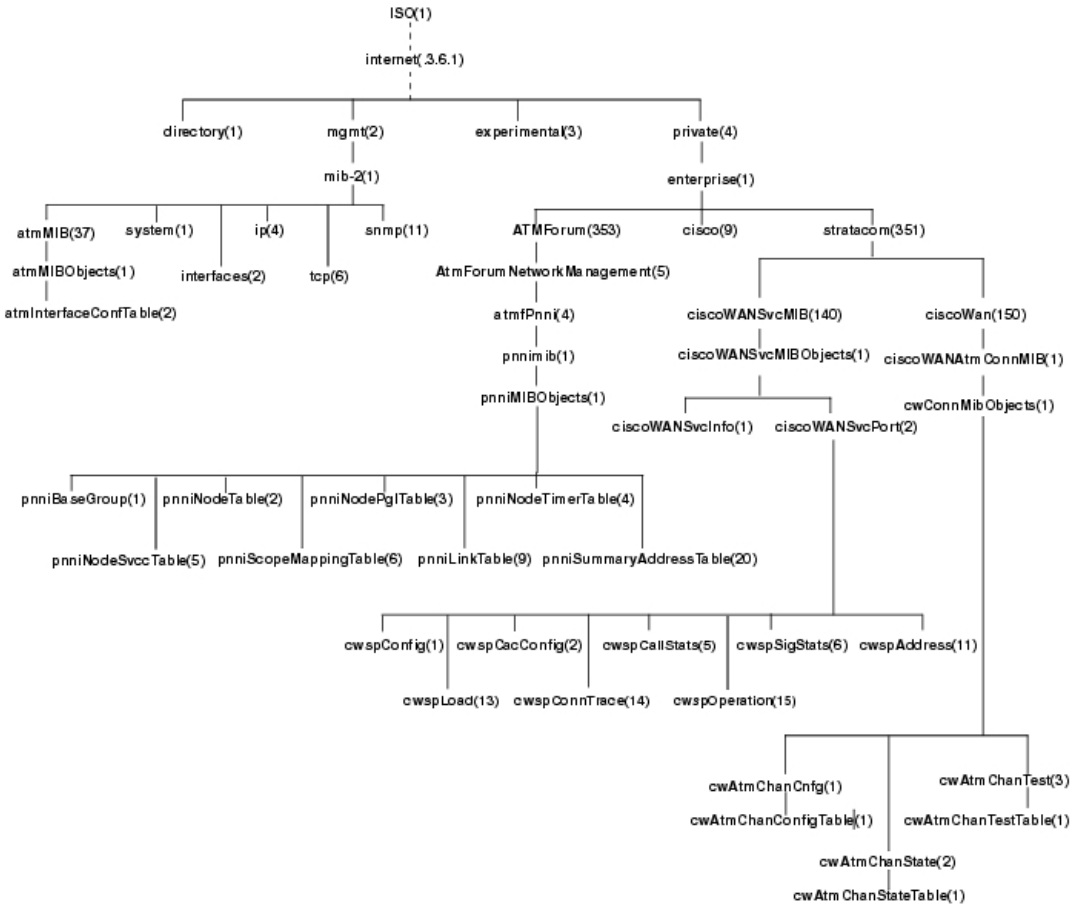

Рисунок 3.8 - Ієрархічне дерево керованих об'єктів у MIB

Отже, у ієрархічному дереві MIB програмний модуль контролю стану застосовує не лише стандартні керовані об’єкти, а й власні, що дозволяє адаптувати систему під специфічні потреби мережевого обладнання.

3.3 Програмний модуль контролю мережевим обладнанням

Поєднання Python та SNMP-менеджера дозволило створити програмний модуль, що запускається також на одноплатному комп’ютері. Мова Python є оптимальною для таких платформ, оскільки розробники створили велику кількість готових бібліотек для управління різноманітними пристроями.

Розроблений модуль сканує мережеві пристрої, визначає таблиці MAC-адрес активних вузлів, формує ієрархічне дерево MIB, взаємодіє з локальним SNMP-сервісом, контролює стан підключеного обладнання, надсилає повідомлення менеджеру SNMP та ініціює процедуру перемикання живлення керованого пристрою.

Перехоплення SNMP-пасток відбувається при виникненні подій, спричинених пристроєм. Це можуть бути зміни стану або виявлення аномалій, які відразу передаються на приймач перехоплень (рисунок 3.9).

Рисунок 3.9 – Моніторинг повідомлень

Як уже зазначалося, розроблений програмний модуль виконує функції агента для менеджера SNMP HP OpenView (скор. HPOV, HP OV), який є сімейством програмних продуктів Hewlett Packard для адміністрування

комп’ютерних мереж і систем. Відкрита архітектура SNMP дозволяє інтегрувати Python-модуль у існуючу систему керування мережевим обладнанням, забезпечуючи його участь у комплексному моніторингу та управлінні.

Програмний модуль складається з кількох підмодулів, кожен із яких відповідає за окремі функції, що наведено в таблиці 3.1.

Таблийя 3.1 – Перелік підмодулів

Підмодуль	Функції та призначення
main.py	Основний модуль, що координує роботу всіх інших підмодулів. Відповідає за логіку взаємодії між процедурами та постійно працює як головний процес на одноплатному комп’ютері. Створює об’єкти та класи, визначає їхні властивості та методи, які використовуються у функціях.
scan.py	Виконує сканування мережевих пристроїв, визначає активні вузли та будує таблиці MAC-адрес.
mac.py	Здійснює збір фізичних адрес (MAC) для подальшого аналізу та ідентифікації пристроїв у мережі.
mib.py	Формує ієрархічну інформаційну базу даних MIB для керованих пристроїв, що використовується програмним модулем для моніторингу та управління.
snmp.py	Виконує запити SNMP до агентів, отримує дані про стан пристроїв та їхні параметри.
trap.py	Відправляє повідомлення-пастки про зміни стану або виявлені аномалії мережевих пристроїв на SNMP-менеджер.
control.py	Забезпечує контроль роботи мережевих пристроїв, взаємодіючи з агентами для перевірки стану та управління параметрами.
reset.py	Виконує перезавантаження керованих мережевих пристроїв у разі збоїв або зависання.

Кожен підмодуль містить набір функцій і процедур, які забезпечують моніторинг та управління пристроями мережі. Зокрема, main.py координує роботу всіх підмодулів, постійно функціонуючи як головний процес на основному менеджері. У цьому підмодулі створюються класи та об’єкти, визначаються їхні властивості і методи, що дозволяють виконувати операції над ними. Для оптимізації коду частина змінних передається у функції без повторного оголошення.

Отримані повідомлення про події (рисунок 3.10) дозволяють автоматично виявляти несправності, забезпечуючи оперативний контроль стану мережевого обладнання.

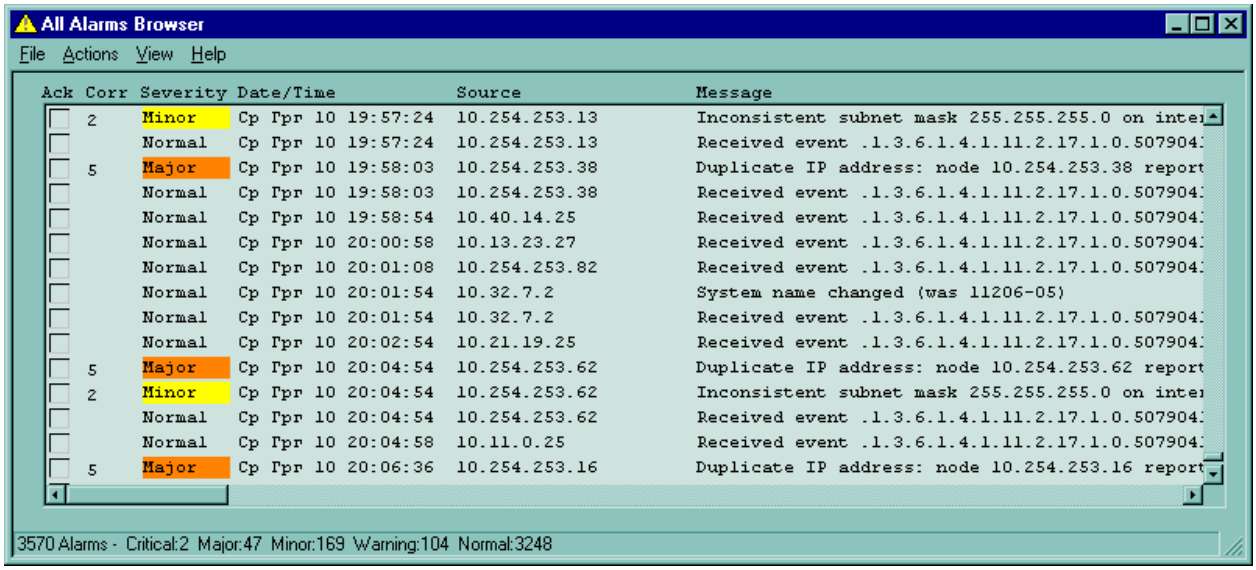

Рисунок 3.10 – Повідомлення про події

У системі контролю мережевого обладнання кожен підмодуль виконує спеціалізовану функцію, яка дозволяє забезпечити комплексний моніторинг та управління мережевими пристроями.

Підмодуль scan.py відповідає за безперервне сканування мережевих пристроїв, введення IP-адрес, створення переліку активних вузлів та відстеження змін їхнього стану. Він також забезпечує прикріплення до серверних скриптів для отримання актуальної інформації. Фактично scan.py формує повну картину мережевого оточення, яке «бачить» програмний модуль, що є ключовим для подальшого моніторингу та управління.

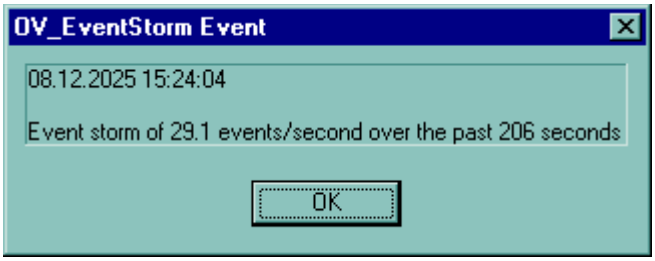

Рисунок 3.11 – Інтенсивність сканування мережевого обладнання

Модулем для обробки MAC-адрес є mac.py Підмодуль відповідає за встановлення зв'язку програмного модуля з базою даних MAC-адрес, куди записуються результати сканування, отримані менеджером SNMP. Ця база містить фізичні адреси всіх мережевих пристроїв у зоні видимості одноплатного контролера, що дозволяє вести точний облік підключених вузлів і визначати їхню активність.

Підмодуль mib.py працює з інформаційною базою MIB (рис 3.12). Цей підмодуль реалізує функції для класифікації мережевих пристроїв та віднесення їх до відповідних груп. Дані з бази MIB обробляються та представляються у вигляді ієрархічних дерев і зв'язків між об'єктами. Результат обробки передається менеджеру SNMP, що дозволяє відображати його на моніторинговій панелі мережевого адміністратора у зрозумілому та структурованому вигляді.

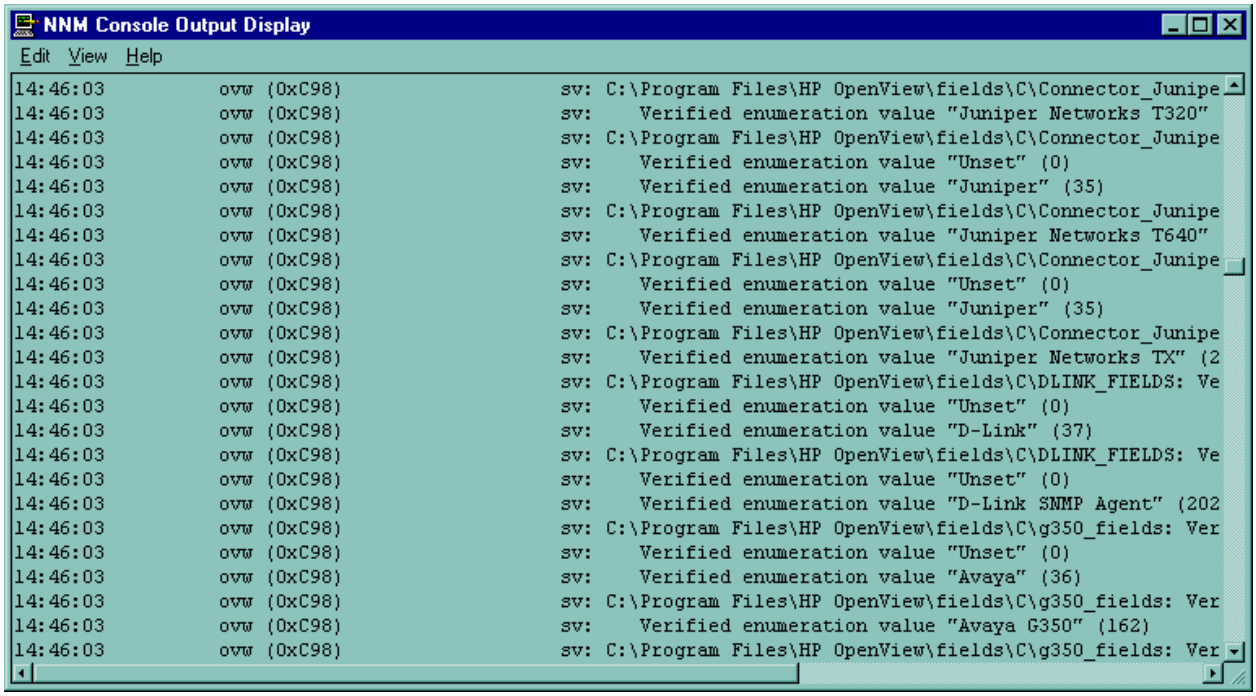

Рисунок 3.12 – Опрацювання MIB мережевого обладнання

Модулем для виконання SNMP-запитів є snmp.py. Цей підмодуль забезпечує обмін даними з мережевими пристроями за допомогою команд SNMP. Він формує таблиці MAC-адрес та визначає належність портів до відповідних VLAN. Якщо отримані дані не змінюються у порівнянні з

попередньою перевіркою, це свідчить про стабільну роботу мережі, що дозволяє адміністраторам оперативно оцінювати стан інфраструктури.

Функція trap.py є для обробки SNMP-пасток. Даний підмодуль відповідає за динамічне відправлення повідомлень про зміни стану мережевих пристроїв. При виникненні події, що змінює стан обладнання, trap.py автоматично надсилає відповідне повідомлення менеджеру SNMP, забезпечуючи швидке інформування адміністратора про потенційні проблеми або відхилення від нормального режиму роботи.

Функцію контролю стану мережевих пристроїв виконує підмодуль control.py.Він є основним виконавчим компонентом програмного модуля. Він містить класи, об’єкти, методи та алгоритми, які визначають поточний стан керованого обладнання. У нормальному режимі вся інформація передається менеджеру SNMP, і модуль функціонує як стандартний агент. У разі виникнення проблем адміністратор миттєво отримує сповіщення, яке зазвичай кодується і розкодовується за допомогою процесора перехоплень SNMP (рисунок 3.13), що дозволяє забезпечити швидкий і точний контроль стану мережі.

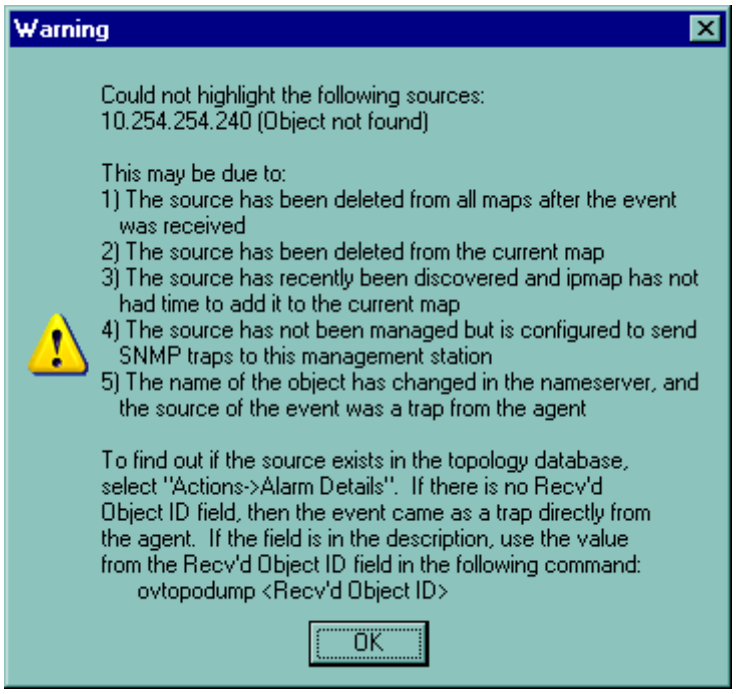

Рисунок 3.13 – Розкодування перехопленого сповіщення

У разі виявлення підмодулем control.py нестабільності або несправностей у роботі мережевого пристрою, він намагається передати цю інформацію менеджеру SNMP. Після отримання повідомлення мережевий адміністратор або оператор може здійснити необхідне втручання, зокрема перезавантаження пристрою. Якщо віддалене управління неможливе, автоматично активується внутрішня процедура контролю стану мережевого обладнання.

Ця процедура включає перевірку доступності пристрою за допомогою ICMP-запитів, а також аналіз наявності фізичної адреси пристрою у таблиці ARP. У випадку відсутності будь-яких відповідей і неможливості отримати інформацію про поточний стан пристрою, викликається функція reset.py, що показано на рис.3.14.

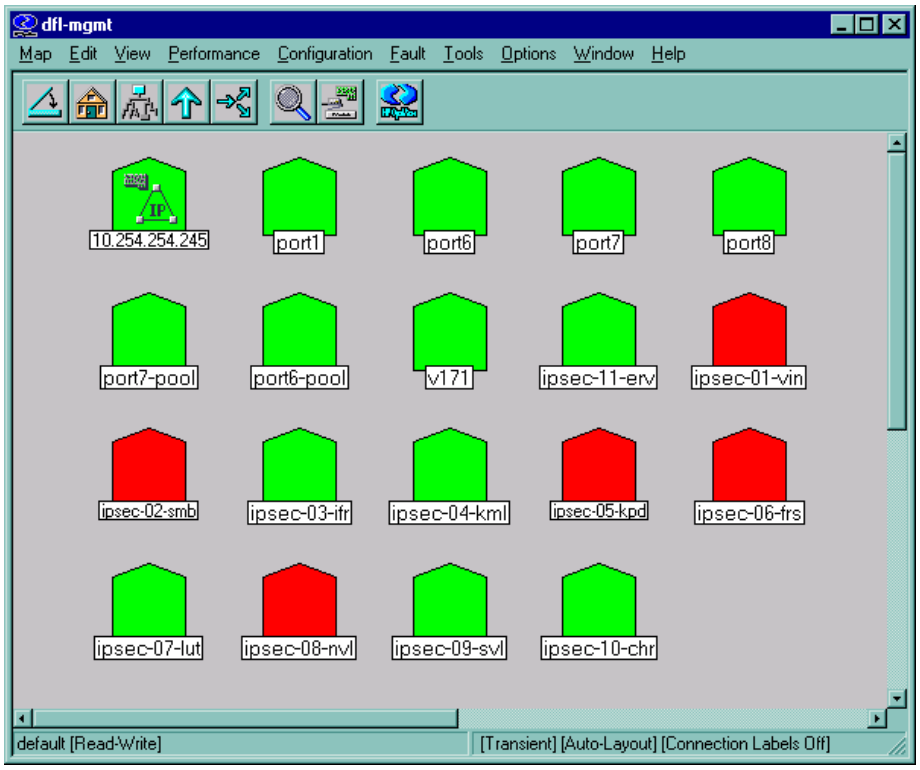

Рисунок 3.14 – Стан одного із мережевих пристроїв

Крім того, control.py зберігає останні дані про стан контрольованого пристрою у тимчасовий файл. Після нормалізації роботи мережі ця інформація передається менеджеру SNMP, що дозволяє вести історію подій

та забезпечує адміністратора актуальною інформацією про мережеву інфраструктуру.

Функція reset.py відповідає за фізичне відключення живлення мережевого пристрою на визначений час, після чого відновлює його електроживлення і повідомляє підмодуль control.py про завершення операції. Це дозволяє автоматично відновлювати працездатність обладнання без необхідності ручного втручання.

ВИСНОВКИ

Дана робота показала можливість реалізації веб-інтерфейсу для керування комутаторами та маршрутизаторами по SNMP. Інтерфейс виявився зручним у використанні, так як не потребує інсталяції на комп'ютер. До розробленої програми є доступ з будь-якого хосту мережі. Функціонал продукту задовольняє усі потреби адміністрування. Швидкодія, веб варіанту програми, не перевищує загальний час виконання схожих запитів на прикладних аналогах. Крім того, в більшості існуючих продуктів, що працюють по SNMP, не реалізована можливість відобразити ієрархічну будову мережі.

Також було виявлено основні проблеми в реалізації продуктів, що працюють через SNMP (англ. Simple Network Management Protocol):

- персональні MIB;
- відносна застарілість протоколу;
- відсутність підтримки SNMP на більшості моделей активного мережевого обладнання

Для написання роботи використовувались:

- протокол SNMP;
- мови програмування PHP, JavaScript;
- підмережа з активним мережевим обладнанням, що підтримує SNMP;
- операційна система з налаштованим SNMP;
- веб-сервер Apache;
- база даних MySQL.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ortmeyer, Cliff. A Brief History of Single Board Computers. A Premier Farnell Company, Electronic Design Uncovered, USA, 2014, 06: 11.
2. Murphy, Niall. Watchdog Timers-To keep a watchdog timer from resetting your system, you've got to kick it regularly. But that's not all there is to watchdog science. We will examine the use and testing of a. Embedded Systems Programming, 2000, 13.12: 112-126.
3. Santic, John. Watchdog timer techniques. Embedded Systems Programming, 1995, 8.4: 58-69.
4. Gross, Kevin P., and Tom Holtzen. "Controlling and Monitoring Audio Systems with Simple Network Management Protocol (SNMP)." Audio Engineering Society Convention 105. Audio Engineering Society, 1998.
5. Mauro, Douglas, and Kevin Schmidt. Essential SNMP: Help for System and Network Administrators. " O'Reilly Media, Inc.", 2005.
6. What is SNMP | SNMP Monitoring: Site24x7. *Site24x7*. URL: https://www.site24x7.com/network/what-is-snmp.html (дата звернення: 10.04.2024).
7. K O. What is IPMI?. MonoVM.com. URL: https://monovm.com/blog/what-is-ipmi/ (дата звернення: 05.06.2024).
8. Supermicro IPMI utilities | supermicro server management utilities | supermicro. Supermicro Data Center Server, Blade, Data Storage, AI System. URL: https://www.supermicro.com/en/solutions/management-software/ipmi-utilities (дата звернення: 05.06.2024).
9. Zissis, Dimitrios; LEKKAS, Dimitrios. Addressing cloud computing security issues. Future Generation computer systems, 2012, 28.3: 583-592.
10. Mauro, Douglas, and Kevin Schmidt. Essential SNMP: Help for System and Network Administrators. " O'Reilly Media, Inc.", 2005.

11. Observability and IT management platform | solarwinds. Observability and IT Management Platform | SolarWinds. URL: http://www.solarwinds.com (дата звернення: 16.02.2024).

12. Tanenbaum A. S. Computer networks. New Jersey : Pearson Education International, 2003.

13. Olifer N., Olifer V. Computer networks: principles, technologies and protocols for network design. 3-тє вид. 2006.

14. Методичні рекомендації до виконання кваліфікаційної роботи за освітнім ступенем “магістр” спеціальності – 174 Автоматизація, комп’ютерно-інтегровані технології та робототехніка / .Під ред. А.І. Сегіна – Тернопіль: ЗУНУ, 2024. – 36 с.

15. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення. -К.: Держстандарт України, 1995. -37 с.

16. Douglas M., Schmidt K. Essential SNMP. 2-ге вид. OReilly, 2005. 460 с.

17. Stephenson, Neal. In the beginning... was the command line. New York: Avon Books, 1999.

18. Schoenwaelder,J. ;Jeffree, T. RFC 4789: Simple Network Management Protocol (SNMP) over IEEE 802 Networks. 2006.

19. Law, David. "IEEE 802.3 Clause 30 Management, MIB, Registers and Function." *IEEE P802. 3az, Energy-efficient Ethernet Task Force, Plenary Week Meeting*. 2007.

20. RFC 1065: Structure and Identification of Management Information for TCP/IP-based internets.

21. RFC 1066: Management Information Base for Network Management of TCP/IP-based internets.

22. RFC 1067: A Simple Network Management Protocol.

23. RFC 1155: Structure and Identification of Management Information for TCP / IP-based internets

24. RFC 1212: Concise MIB Definitions

25. RFC 1213: Management Information Base for Network Management of TCP / IP-based internets: MIB-II

26. RFC 1157: A Simple Network Management Protocol v.3

27. IEEE Xplore [Електронний ресурс].- Режим доступу: www.ieeexplore.ieee.org. Standards Digital Library.

28. Microsoft FAQ [Електронний ресурс].- Режим доступу: support.microsoft.com. Служба підтримки Microsoft.

29. GNU/Linux [Електронний ресурс] .- Режим доступу: www.linuxquestions.org. Where Linux users come for help.

30. Cisco [Електронний ресурс].- Режим доступу: https://www.cisco.com/site/us/en/products/networking/cloud-networking-switches/index.html#tabs-9da71fbd27-item-1288c79d71-tab

31. Juniper [Електронний ресурс].- Режим доступу: https://www.juniper.net/documentation/us/en/hardware/ex4300/topics/topic-map/connecting-qfx-series-ex-series-in-qfx-virtual-chassis.html

32. Huawai [Електронний ресурс].- Режим доступу: https://carrier.huawei.com/en/products/fixed-network/b2b/gateway
https://e.huawei.com/ua/products/switches/campus-switches/s5735-s-v2-ge

33. DLink [Електронний ресурс].- Режим доступу: https://www.dlink.com/ua/uk/for-business/switching

34. Netgear [Електронний ресурс].- Режим доступу: https://www.netgear.com/business/wired/switches/

ДОДАТОК 1 – Сторожовий таймер у БІОС

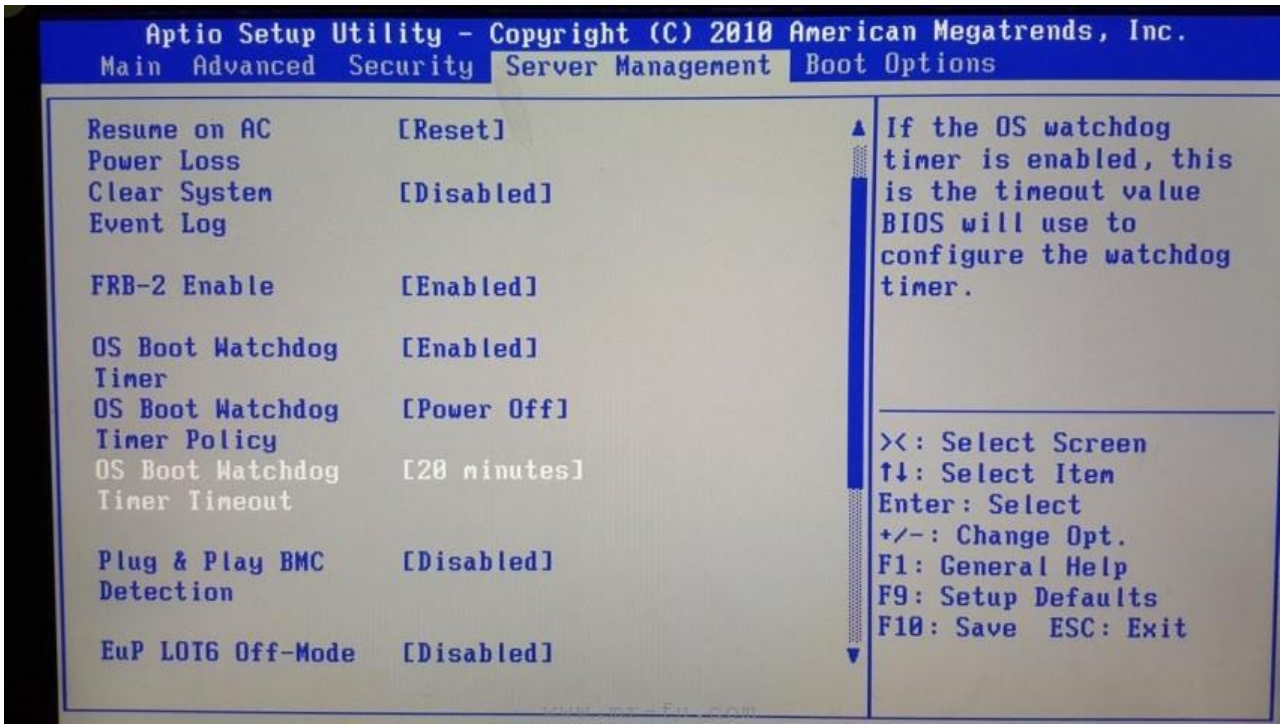

Рисунок 1 Налаштування Watchdog у BIOS сервера

Рисунок 2 Налаштування Watchdog сервера Cisco

ДОДАТОК 2 – Таблиця 1 Перелік OID параметрів комутатора в MIB

RFC1213-MIB	1.3.6.1.2.1.1.1.0	sysDescr.0	DGS-3200-16 Gigabit Ethernet Switch
RFC1213-MIB	1.3.6.1.2.1.1.2.0	sysObjectID.0	1.3.6.1.4.1.171.10.101.2
RFC1213-MIB	1.3.6.1.2.1.1.3.0	sysUpTime.0	5189040
RFC1213-MIB	1.3.6.1.2.1.1.4.0	sysContact.0	@edu.ua
RFC1213-MIB	1.3.6.1.2.1.1.5.0	sysName.0	sw-3200-16a.edu.ua
RFC1213-MIB	1.3.6.1.2.1.1.6.0	sysLocation.0	Aquarium 1101 Old Columbus sw-10c-old
RFC1213-MIB	1.3.6.1.2.1.1.7.0	sysServices.0	3
SNMPv2-MIB	1.3.6.1.2.1.1.8.0	sysORLastChange.0	5189048
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.1	sysORID.1	1.3.6.1.4.1.171.12.93
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.2	sysORID.2	1.3.6.1.4.1.171.12.39
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.3	sysORID.3	1.0.8802.1.1.2.65538.131072.4156424
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.4	sysORID.4	1.0.8802.1.1.2.65538.131072.4165340.12.8802.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.5	sysORID.5	1.0.8802.1.1.2.1.5.4623.65540.131072.4174084
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.6	sysORID.6	1.0.8802.1.1.2.1.5.4795
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.7	sysORID.7	1.3.6.1.4.1.171.12.74
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.8	sysORID.8	1.3.6.1.4.1.171.12.78
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.9	sysORID.9	1.3.6.1.4.1.171.12.77
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.10	sysORID.10	1.3.6.1.4.1.171.12.76
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.11	sysORID.11	1.3.6.1.4.1.171.12.25
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.12	sysORID.12	1.0.8802.1.1.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.13	sysORID.13	1.3.6.1.4.1.171.12.55
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.14	sysORID.14	1.3.6.1.4.1.171.12.30
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.15	sysORID.15	1.3.6.1.2.1.67.1.2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.16	sysORID.16	1.3.6.1.2.1.67.2.2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.17	sysORID.17	1.3.6.1.4.1.171.12.23
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.18	sysORID.18	1.3.6.1.4.1.171.12.73
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.19	sysORID.19	1.3.6.1.4.1.171.12.53
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.20	sysORID.20	1.3.6.1.4.1.171.12.64
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.21	sysORID.21	1.3.6.1.4.1.171.12.79
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.22	sysORID.22	1.3.6.1.4.1.171.12.42
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.23	sysORID.23	1.0.8802.1.1.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.24	sysORID.24	1.3.6.1.2.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.25	sysORID.25	1.3.6.1.2.1.17
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.26	sysORID.26	1.3.6.1.6.3.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.27	sysORID.27	1.3.6.1.2.1.16
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.28	sysORID.28	1.3.6.1.6.3.10
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.29	sysORID.29	1.3.6.1.6.3.11
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.30	sysORID.30	1.3.6.1.6.3.13
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.31	sysORID.31	1.3.6.1.6.3.12
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.32	sysORID.32	1.3.6.1.6.3.15
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.33	sysORID.33	1.3.6.1.6.3.16
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.34	sysORID.34	1.3.6.1.6.3.18
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.35	sysORID.35	1.3.6.1.2.1.67.1.2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.36	sysORID.36	1.3.6.1.2.1.67.2.2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.37	sysORID.37	1.3.6.1.2.1.35
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.38	sysORID.38	1.3.6.1.2.1.17.6
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.39	sysORID.39	1.3.6.1.2.1.17.7
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.40	sysORID.40	1.3.6.1.2.1.16.20.8
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.41	sysORID.41	1.3.6.1.2.1.31

SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.42	sysORID.42	1.3.6.1.2.1.80
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.43	sysORID.43	1.3.6.1.4.1.171.12.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.44	sysORID.44	1.3.6.1.4.1.171.11.101.2.2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.45	sysORID.45	1.3.6.1.4.1.171.12.3
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.46	sysORID.46	1.3.6.1.4.1.171.12.4
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.47	sysORID.47	1.2.840.10006.300.43.131076
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.48	sysORID.48	1.3.6.1.4.1.171.12.5
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.49	sysORID.49	1.3.6.1.4.1.171.12.6
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.50	sysORID.50	1.3.6.1.4.1.171.12.7
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.51	sysORID.51	1.3.6.1.4.1.171.12.8
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.52	sysORID.52	1.3.6.1.4.1.171.12.9
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.53	sysORID.53	1.3.6.1.4.1.171.12.10
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.54	sysORID.54	1.3.6.1.4.1.171.12.11
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.55	sysORID.55	1.3.6.1.4.1.171.12.12
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.56	sysORID.56	1.3.6.1.4.1.171.12.15
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.57	sysORID.57	1.3.6.1.4.1.171.12.16
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.58	sysORID.58	1.3.6.1.4.1.171.12.19
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.59	sysORID.59	1.3.6.1.4.1.171.12.23
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.60	sysORID.60	1.3.6.1.4.1.171.12.26
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.61	sysORID.61	1.3.6.1.4.1.171.12.34
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.62	sysORID.62	1.3.6.1.4.1.171.12.50
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.63	sysORID.63	1.3.6.1.4.1.171.11.101.2.3
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.64	sysORID.64	1.3.6.1.4.1.171.12.35
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.65	sysORID.65	1.3.6.1.4.1.171.12.27
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.66	sysORID.66	1.3.6.1.4.1.171.12.39
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.67	sysORID.67	1.3.6.1.2.1.81
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.68	sysORID.68	1.3.6.1.4.1.171.12.56
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.69	sysORID.69	1.3.6.1.4.1.171.12.64
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.70	sysORID.70	1.3.6.1.4.1.171.12.37
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.71	sysORID.71	1.3.6.1.4.1.171.12.58
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.72	sysORID.72	1.3.6.1.4.1.171.12.72
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.73	sysORID.73	1.3.6.1.4.1.171.12.69
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.74	sysORID.74	1.3.6.1.2.1.158
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.75	sysORID.75	1.3.6.1.4.1.171.12.87
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.76	sysORID.76	1.3.6.1.4.1.171.12.62
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.77	sysORID.77	1.3.6.1.4.1
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.78	sysORID.78	1.3.6.1.4.1.171.12.61
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.79	sysORID.79	1.3.6.1.4.1.171.12.70
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.80	sysORID.80	1.3.6.1.4.1.171.12.38
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.81	sysORID.81	1.3.6.1.4.1.171.12.84
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.82	sysORID.82	1.3.6.1.4.1.171.12.85
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.83	sysORID.83	1.3.6.1.2.1.26.786434
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.84	sysORID.84	1.3.6.1.4.1.171.12.27
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.85	sysORID.85	1.3.6.1.4.1.171.12.3
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.86	sysORID.86	1.3.6.1.4.1.171.12.35
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.2.87	sysORID.87	1.3.6.1.4.1.171.12.92
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.1	sysORDescr.1	swL2ProtocolTunnelMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.2	sysORDescr.2	swJWACMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.3	sysORDescr.3	lldpMib
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.4	sysORDescr.4	lldpXdot1Mib
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.5	sysORDescr.5	lldpXdot3Mib

SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.6	sysORDescr.6	lldpXMedMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.7	sysORDescr.7	swVoiceVLANMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.8	sysORDescr.8	swERPSMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.9	sysORDescr.9	swNlbMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.10	sysORDescr.10	swBpduProtectionMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.11	sysORDescr.11	swPktStormMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.12	sysORDescr.12	ieee8021paeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.13	sysORDescr.13	swRadiusAccountMGMTMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.14	sysORDescr.14	swdot1xMGMTMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.15	sysORDescr.15	radiusAuthClientMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.16	sysORDescr.16	radiusAccClientMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.17	sysORDescr.17	swIpMacBindMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.18	sysORDescr.18	swMcastSnoopingMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.19	sysORDescr.19	swMcastFilterMgmt
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.20	sysORDescr.20	swMcastVlanMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.21	sysORDescr.21	swPPPoEMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.22	sysORDescr.22	swDHCPRelayMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.23	sysORDescr.23	ieee8021paeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.24	sysORDescr.24	rfc1213Mib
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.25	sysORDescr.25	dot1dBridge
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.26	sysORDescr.26	snmpMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.27	sysORDescr.27	rmon2
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.28	sysORDescr.28	snmpFrameworkMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.29	sysORDescr.29	snmpMPDMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.30	sysORDescr.30	snmpNotificationMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.31	sysORDescr.31	snmpTargetMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.32	sysORDescr.32	snmpUsmMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.33	sysORDescr.33	snmpVacmMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.34	sysORDescr.34	snmpCommunityMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.35	sysORDescr.35	radiusAuthClientMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.36	sysORDescr.36	radiusAccClientMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.37	sysORDescr.37	etherMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.38	sysORDescr.38	pBridgeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.39	sysORDescr.39	qBridgeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.40	sysORDescr.40	rmonMibModule
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.41	sysORDescr.41	ifMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.42	sysORDescr.42	pingMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.43	sysORDescr.43	agentGeneralMgmt
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.44	sysORDescr.44	swL2MgmtMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.45	sysORDescr.45	swAuthCtrl
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.46	sysORDescr.46	swLagMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.47	sysORDescr.47	ie8023adMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.48	sysORDescr.48	swAACMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.49	sysORDescr.49	swSSHMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.50	sysORDescr.50	swSSLMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.51	sysORDescr.51	swSingleIPMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.52	sysORDescr.52	swAclMgmtMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.53	sysORDescr.53	swTimeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.54	sysORDescr.54	swEquipmentMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.55	sysORDescr.55	swSysLogMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.56	sysORDescr.56	swMSTPMIB

SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.57	sysORDescr.57	swProtocolVLANMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.58	sysORDescr.58	swSafeGuardMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.59	sysORDescr.59	swIpMacBindMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.60	sysORDescr.60	swIPv6StaticRouteMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.61	sysORDescr.61	swMldSnpMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.62	sysORDescr.62	swTimeRangeMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.63	sysORDescr.63	swL3MgmtMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.64	sysORDescr.64	swMBAMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.65	sysORDescr.65	swWACMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.66	sysORDescr.66	swJWACMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.67	sysORDescr.67	traceRouteMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.68	sysORDescr.68	swSMBVMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.69	sysORDescr.69	swMcastVlanMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.70	sysORDescr.70	swFilterMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.71	sysORDescr.71	swCableDiagMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.72	sysORDescr.72	swDdmMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.73	sysORDescr.73	swPrivateVLANMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.74	sysORDescr.74	dot3OamMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.75	sysORDescr.75	swDULDMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.76	sysORDescr.76	swARPSpoofingPreventMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.77	sysORDescr.77	swSMTPMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.78	sysORDescr.78	swQoSMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.79	sysORDescr.79	swStaticFdbMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.80	sysORDescr.80	swDHCPServerMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.81	sysORDescr.81	swDHCPv6RelayMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.82	sysORDescr.82	swDNSResolverMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.83	sysORDescr.83	mauMod
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.84	sysORDescr.84	swWACMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.85	sysORDescr.85	swAuthCtrl
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.86	sysORDescr.86	swMBAMIB
SNMPv2-MIB	1.3.6.1.2.1.1.9.1.3.87	sysORDescr.87	swZoneDefenseMIB

ДОДАТОК 3 – Приклад MIB комутатора

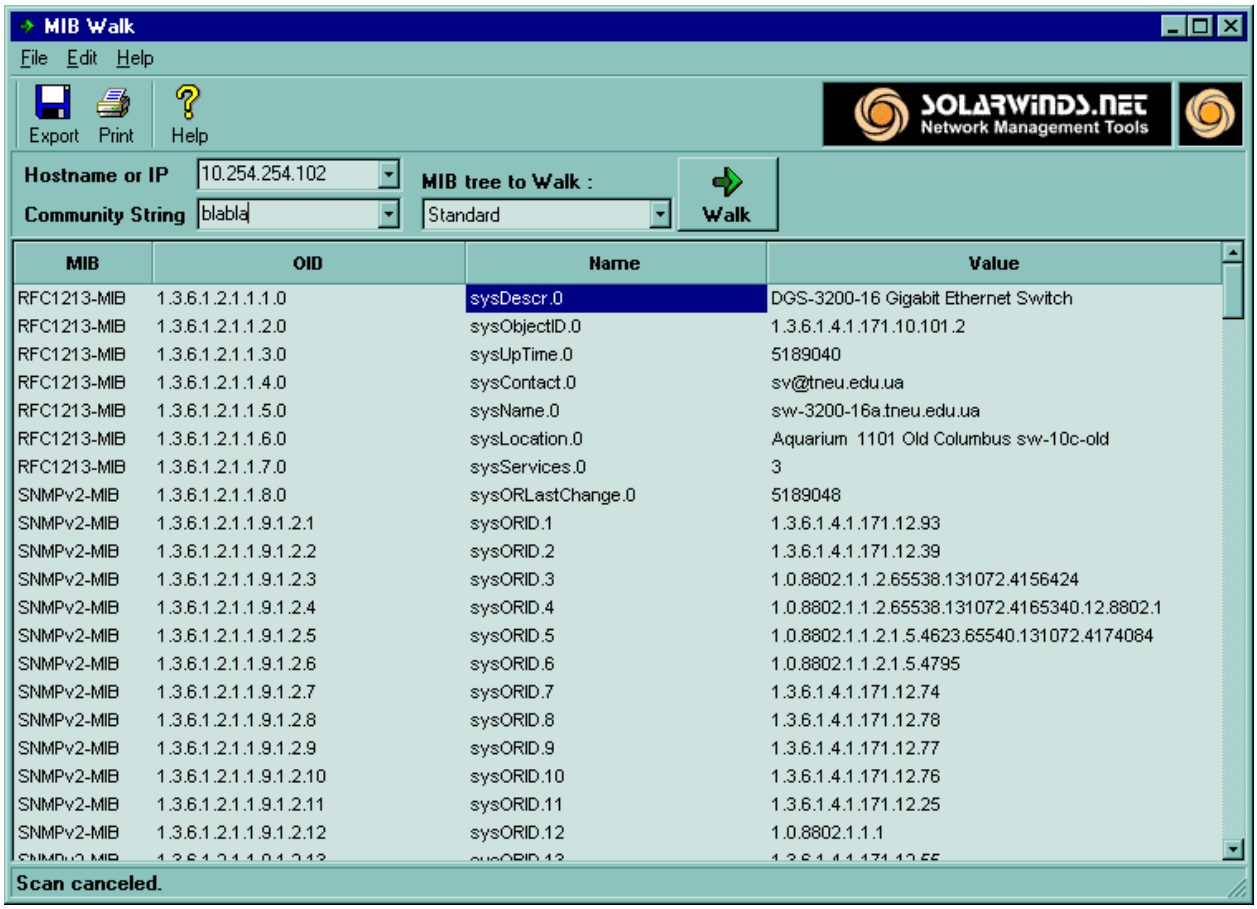

Рисунок 1 – Нумерація OID комутатора D-Link

ДОДАТОК 4 – Файл batchping.ovpl

```
#!/opt/OV/bin/Perl/bin/perl
#!/usr/local/bin/perl
#!/usr/bin/perl

# modify the interpreter line for your installation of PERL
# for Unix, make sure Getopt, POSIX, and Net modules are installed
# On Microsoft Windows pass the name of the script to perl
# MSWindows does NOT use Net::Ping

$| = 1; # flush output

# most of the time DEBUG is only useful to the developer of this tool
# 1 or greater turns on debugging or -d on the command line
$DEBUG = 0 ; # normally set to 0

use Getopt::Std ;
use POSIX ":sys_wait_h";

$win_str= $^O =~ /Win.+/ ; # match the OS string from Microsoft Windows

# This is for HP-UX, Solarus, and Linux systems.
# Can use Net::Ping and can use "fork" on them.
if (! $win_str) {
require Net::Ping ;
}

# print the usage statement if we don't get any arguments.
# or we get the -? help request
if (scalar(@ARGV) < 1) {
    if ($win_str) {
    win_usage() ;
    }
    else {
    usage() ;
    }
} elsif ($ARGV[0] =~ /-\?/) {
    if ($win_str) {
    win_usage() ;
    }
    else {
    usage() ;
    }
}

$opt_d = 0 ; # init this so -w won't complain

#
# -f <file with targets> -h "quoted list of targets on command line"
# -p <number of concurrent processes allowed> -d turn on DEBUG
#
getopts('p:f:h:d') ;

@targets = ();

if ($opt_f) {
    # we have a file name
    open (LIST, "$opt_f") || die ("WARNING: Unable to open $opt_f !") ;
    @targets = <LIST> ;
    chomp @targets ;
    close LIST ;

} # end if ($opt_f)
```

```
if ($opt_p) { # how many concurrent ping processes can we spawn ?
    $p_limit = $opt_p ; # if we get a process limit on the command line
} else {
    $p_limit = 4 ; # else we default to this setting
}

if ($opt_h) {
    @targets = (split (" ",$opt_h)) ;
}


if ($opt_d) {
$DEBUG = 1 ; # user asked for DEBUG
}

# check for a Windows OS
if ($win_str) { # jump to the Windows code and then come back to exit.
run_win() ;

print "$0 finished \n" ;
exit ;

}

# now into Unix section, Windows section is run_win sub.

($basename = $0) =~ s!.*/!! ; # clean up any path gunk
$0 = $basename ;

if ($> != 0) {
    print "\n$0 must be run as root so that I can ping the targets.\n" ;
    print "Please switch user to root and re-run $0.\n\n" ;
    exit -1 ;
}


##
## PING ALL TARGETS
## using the Net::Ping module here to avoid the OS stuff
## down side is that you must be root to do the pings
##


if ($DEBUG >= 1) {
print "Using upto a maximum of $p_limit concurrent ping processes.\n" ;
}

$n_active = 0 ; # init the number of active processes

$p = Net::Ping->new("icmp", 1) ;
TARGET_LOOP: foreach $ping_target (@targets) {

    unless ($pid2 = fork) { # fork a child process

        if ($DEBUG >= 1) {
        print "\nProcessing $ping_target\n" ;
        }

        $p->ping($ping_target) ; # the reason for the whole script

        if ($DEBUG >= 1) {
            if ($p->ping($ping_target)) {
                print "$ping_target is alive\n" ;
```

```
            } else {
                print "$ping_target unreachable\n" ;
            }
        }

        exit 0 ; # child of fork MUST exit here
    } # end unless ($pid2 = fork)

    if ($DEBUG >= 1) {
        print "process $pid2 just started\n" ;
    }

    # put the pids in a hash
    # this hash is used to track the active forks
    $pid_list{$pid2} = $pid2 ; # save the pids
    $n_active ++ ; # inc the number of active processes

    if ($n_active < $p_limit) {
        next TARGET_LOOP ;
    } else {
        $size_of = %pid_list ;

        if ($DEBUG >= 1) {
                print "the pid hash has $size_of entries\n" ;
        }

        if ($DEBUG >= 1) {
        foreach $list_pid (keys %pid_list) {
                print "$pid_list{$list_pid}\n" ;
            }
        }


        # this loop checks the viability of each process and deletes it
      # when it is done
        # and decrements the active counter so we can start another process

        P_MONITOR_LOOP: foreach $list_pid (keys %pid_list) {
        if ($DEBUG >= 1) {
            print "testing for process $list_pid\n" ;
        }
            if (waitpid($list_pid,WNOHANG) == -1) {

                if ($DEBUG >= 1) {
                            print "$list_pid must be finished\n" ;
                }

                delete $pid_list{$list_pid} ; # nuke the entry in the hash
                $n_active -- ;
                next P_MONITOR_LOOP ;

            } else {
                    next P_MONITOR_LOOP ;
            }

        } # end P_MONITOR_LOOP: foreach $list_pid (keys %pid_list)

        unless ($n_active < $p_limit) {
            goto P_MONITOR_LOOP;
        } # end unless ($n_active < $p_limit)

    } # end else
} # end TARGET_LOOP: foreach $target (@targets)
```

```
# we are done with all of the targets, so wait on the last processes
# to finish and then cleanup.

while ($n_active >= 2) {
  foreach $list_pid (keys %pid_list) {

        if ($DEBUG >= 1) {
            print "testing for process $list_pid\n" ;
        }

        if (waitpid($list_pid,WNOHANG) == -1) {

            if ($DEBUG >= 1) {
            print "$list_pid must be finished\n" ;
            }

            delete $pid_list{$list_pid}  ; # nuke the entry in the hash
            $n_active -- ;
            next ;

         } else {
             next ;
         }
  } # end foreach $list_pid (keys %pid_list)

} # end while ($n_active >= 2)

if ($DEBUG >= 1) {
print "active process count is $n_active\n" ;
}

# look for the last process alive
if ($n_active == 1) {
    foreach $list_pid (keys %pid_list) {

        if ($DEBUG >= 1) {
                print "waiting on $list_pid , the last process running\n" ;
        }

        waitpid($list_pid,0) ; # wait for the last process to really finish.
    } # end foreach $list_pid (keys %pid_list)
} # end if ($n_active == 1)

$p->close() ;

# let them know we are done
print "$0 finished \n" ;
exit ;

sub usage() {
    printf("%s \n -p <number of processes>  -f <host file> || -h
\"hosts\"\n", $0);
    printf("\t-p number of concurrent ping processes allowed. default is
4\n") ;
    printf("\t-h requires a \"quoted list\"\n");
    printf("\t-f file containing a list of hosts 1 per line\n");
    printf("\t-d turn on full DEBUG from the command line.\n");
    exit(1);
}

# every thing below is Windows.
#
sub win_usage()
```

```
{

    $os = $^O ;
    print "\nRunning on $os\n" ;
    printf("%s \n -f <host file> || -h \"hosts\"\n", $0);
    printf("\t-h requires a \"quoted list\"\n");
    printf("\t-f file containing a list of hosts 1 per line\n");
    printf("\t-d turn on full DEBUG from the command line.\n");
    exit(1);
}

sub getDevNull() {
    # get either nul or /dev/null for redirecting stderr
    if ($win_str) {
        return "nul";
    } else {
        return "/dev/null";
    }
}


sub run_win() {

    foreach $ping_target (@targets) {

      if ($DEBUG >= 1) {
          print "\nProcessing $ping_target\n" ;
      }

      $cmd ="c:/\WINNT/\system32/\ping.exe -n 2 $ping_target" ;

      if ($DEBUG >= 1) {
          print "running command \n $cmd \n" ;
      }

      if ($DEBUG >= 1) {
          system $cmd ; # let the output come to the screen
      } else {
      # the back ticks capture pings output so it does NOT go the screen.
          $junk = `$cmd` ;
      }

    }
}
```

ДОДАТОК 5 – Файл main.py

```
import time, sys
from time import sleep
from RPi import GPIO
from gpiozero import OutputDevice
from threading import Thread
from snmp import MIB

seg =
[MIB(27,active_high=False),MIB(25,active_high=False),MIB(24,active_high=Fal
se),
MIB(23,active_high=False),MIB(22,active_high=False),MIB(18,active_high=Fals
e),
MIB(17,active_high=False)]
segmentPattern = [[0,1,2,3,4,5],[1,2],[0,1,6,4,3],[0,1,2,3,6],[1,2,5,6],[0,2,3,5,6], #0
to 5
[0,2,3,4,5,6],[0,1,2],[0,1,2,3,4,5,6],[0,1,2,5,6],[0,1,2,4,5,6], #6 to A
[2,3,4,5,6],[0,3,4,5],[1,2,3,4,6],[0,3,4,5,6],[0,4,5,6] ] #B to F
sensor = NetDevice(15,4)
def main() :
print("Display state of network device")
while 1:
distance = sensor.distance * 10 # distance in ms
print("distance",distance)
if distance >= 10.0:
distance = 16.0
display(int(distance))
time.sleep(0.8)
def convert_temp(gen):
for value in gen:
```

```
yield (value * 3.3 - 0.5) * 100
adc = MCP3008(channel=7)
graph = MIBGraph (26, 19, 13, 6, 5, pwm=True)
for temp in convert_temp(adc.values):
bars = temp / 35
graph.value = bars
sleep(1)
def display(number):
for i in range(0,7):
seg[i].off()
if number < 16:
for i in range(0,len(segmentPattern[number])):
seg[segmentPattern[number][i]].on()
# Main program logic:
if __name__ == '__main__':
main()
```

ДОДАТОК 6 – Файл scan.py

```
import scapy.all as scapy
def scan(ip):
    arp_request = scapy.ARP(pdst=ip)
    broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
    arp_request_broadcast = broadcast / arp_request
    answered_list = scapy.srp(arp_request_broadcast, timeout=1, verbose=False)[0]
    results = []
    for element in answered_list:
     result = {"ip": element[1].psrc, "mac": element[1].hwsrc}
     results.append(result)
     return results
def display_results(results):
    print("IP Address\t\tMAC Address")
    print("-----------------------------------------")
    for result in results:
        print(result["ip"] + "\t\t" + result["mac"])
target_ip = "10.254.254.0/24"
scan_results = scan(target_ip)
display_results(scan_results)
```

ДОДАТОК 7 – Файл mac.py

```
import sys
from requests import get
__version__ = "1.0"
class Macpyoui:
    def __init__(self, api):
        self.api = api
site = "https://api.macvendors.com/"
data = Macpyoui(site)
macaddress = input("Please enter the MAC address: ")
def searchmac():
    macsend = data.api + macaddress
    vendorsearch = get(macsend).text
    if "Not Found" in vendorsearch:
        print("MAC address not found.")
    elif len(sys.argv) == 1:
        print("No MAC address entered.")
    else:
        print(vendorsearch)
if __name__ == "__main__":
    searchmac()
```

ДОДАТОК 8 – Файл mib.py

```
import sys
from requests import get

 {
    "ifMIB": {
       "name": "ifMIB",
       "oid": "1.3.6.1.2.1.31",
       "class": "moduleidentity",
       "revisions": [
         "2007-02-15 00:00",
         "1996-02-28 21:55",
         "1993-11-08 21:55"
       ]
     },
    ...
    "ifTestTable": {
      "name": "ifTestTable",
      "oid": "1.3.6.1.2.1.31.1.3",
      "nodetype": "table",
      "class": "objecttype",
      "maxaccess": "not-accessible"
    },
    "ifTestEntry": {
      "name": "ifTestEntry",
      "oid": "1.3.6.1.2.1.31.1.3.1",
      "nodetype": "row",
      "class": "objecttype",
      "maxaccess": "not-accessible",
      "augmention": {
```

```
      "name": "ifTestEntry",
      "module": "IF-MIB",
      "object": "ifEntry"
    }
  },
  "ifTestId": {
    "name": "ifTestId",
    "oid": "1.3.6.1.2.1.31.1.3.1.1",
    "nodetype": "column",
    "class": "objecttype",
    "syntax": {
      "type": "TestAndIncr",
      "class": "type"
    },
    "maxaccess": "read-write"
  },

}
```

ДОДАТОК 9 – Файл snmp.py

```
import re
import json
from sys import argv
from pysnmp.entity.rfc3413.oneliner import cmdgen
import logging
logging.basicConfig(
    level=logging.DEBUG, filename="./log.txt",
    format='%(asctime)s  %(name)s.%(funcName)s  +%(lineno)s:  %(levelname)-8s
[%(process)d] %(message)s',
)
logger = logging.getLogger("./log.txt")
class Device:
    def __init__(self, ipswitch, ro_community, oid_mt, port=161):
        self.ip = ipswitch
        self.ro = ro_community
        self.oid = oid_mt
        self.port = port
        self.if_oids = ['ifAdminStatus', 'ifOperStatus', 'ifInOctets', 'ifOutOctets']
        self.types_response = {'7': 'ifAdminStatus',
                               '8': 'ifOperStatus',
                               '10': 'ifInOctets',
                               '16': 'ifOutOctets'
                               }
        self.re_part        =        re.compile("(\d\.\d\.\d\.\d\.\d\.\d\.)(?P<part_mt>.*?)$",
re.MULTILINE | re.DOTALL)
        self.part_mt_oid = self.re_part.search(self.oid).group('part_mt')
        self.re_mt          =          re.compile(f'\S+({self.part_mt_oid})\.(?P<port>\d{1,
2})\.(?P<sign>\d+)',
                               re.MULTILINE | re.DOTALL)
```

```
        self.re_if = re.compile("\S+\:\:\S+2\.2\.1\.(?P<key>\d+)\.(?P<port>\d{1,2})$",
                                re.MULTILINE | re.DOTALL)
        self.result = {}

    def get_ifwalk(self) -> dict:
        """
        Отримання відповіді від комутатора на ifAdminStatus, ifOperStatus,
ifInOctets, ifOutOctets и переданий тип.
        :return: self.result: dict
        """

        oids_form = [(oid_if,) for oid_if in self.if_oids]
        oids_form.extend((self.oid,))

        try:
            cmdGen = cmdgen.CommandGenerator()

            errorIndication, errorStatus, errorIndex, varBindTable = cmdGen.nextCmd(
                cmdgen.CommunityData(self.ro, mpModel=1),
                cmdgen.UdpTransportTarget((self.ip, self.port)),
                *oids_form)

            if errorIndication:
                raise BaseException(f"errorIndication: {errorIndication}")
            if errorStatus:
                raise BaseException(f"errorStatus: "
                                    f"{errorStatus.prettyPrint(), errorIndex and varBindTable[-
1][int(errorIndex) - 1] or '?'}")
```

```
        # якщо немає помилок в отриманому запиті – записуємо всі параметри в базу даних MIB
        for varBindTableRow in varBindTable:
            for name, val in varBindTableRow:

                founds_mt_responce = self.re_mt.search(name.prettyPrint())
                if founds_mt_responce is not None:
                    port = founds_mt_responce.group("port")
                    self.result.setdefault('sign', {})[port] = founds_mt_responce.group("sign")
                    self.result.setdefault('link', {})[port] = val.prettyPrint()

                found_if_responce = self.re_if.search(name.prettyPrint())
                if found_if_responce is not None:
                    port = found_if_responce.group('port')
                    type_response = self.types_response.get(found_if_responce.group('key'))
                    if (type_response in ['ifAdminStatus', 'ifOperStatus']) and (val.prettyPrint() == '1'):
                        status = 'up' if val.prettyPrint() == '1' else 'down'
                        self.result.setdefault(type_response, {})[port] = status
                        continue
                    self.result.setdefault(type_response, {})[port] = val.prettyPrint()


        except BaseException as bex:
            logger.error(bex)
        return self.result
```

```
if __name__ == "__main__":
    name_script, ip, ro, oid = argv
    device = Device(ip, ro, oid)
    print(json.dumps(device.get_ifwalk()))
```

ДОДАТОК 10 – Файл control.py

```
import time, sys
from gpiozero import NetDevice, OutputDevice
from threading import Thread
sensor = NetDevice(echo = 16, trigger = 20)
IN1_m1 = OutputDevice(17)
IN2_m1 = OutputDevice(18)
IN3_m1 = OutputDevice(21)
IN4_m1 = OutputDevice(22)
IcmpPing_m1 = [IN1_m1,IN2_m1,IN3_m1,IN4_m1] # SNMP v1 ping
IN4_m2 = OutputDevice(19)
IN3_m2 = OutputDevice(13)
IN2_m2 = OutputDevice(5)
IN1_m2 = OutputDevice(6)
IcmpPing_m2 = [IN1_m2,IN2_m2,IN3_m2,IN4_m2] # SNMP v2 ping
Seq = [[1,0,0,1], # Defi ne icmp sequence
[1,0,0,0], # as shown in manufacturer’s datasheet
[1,1,0,0],
[0,1,0,0],
[0,1,1,0],
[0,0,1,0],
[0,0,1,1],
[0,0,0,1]]

Ping = len(Seq)
all_clear = True
running = True
def bump_watch(): # thread to watch for obstacles
global all_clear
while running:
value = sensor.distance
```

```
if value < 0.1: # trigger if obstacle
all_clear = False
else:
all_clear = True
def move_bump(snmp_request='F', seqsize=1, numicmps=2052):
counter = 0 # 2052 icmps = 1 revolution for icmp size of 2
IcmpDir = seqsize # Set to 1 or 2 for fwd, -1 or -2 for back
if snmp_request == 'T':
IcmpDir = IcmpDir * -1
WaitTime = 10/fl oat(1000) # adjust this to change speed
Pinger = 0
while all_clear and counter < numicmps: # if no obstacles
for pin in range(0, 4):
Lpin = IcmpPing_m1[pin]
Rpin = IcmpPing_m2[pin]
if Seq[Pinger][pin]!=0: # F=fwd, B=back, L=left, R=right
if snmp_request == 'L' or snmp_request == 'B' or snmp_request == 'F':
Lpin.on() # Left device
if snmp_request == 'R' or snmp_request == 'B' or snmp_request == 'F':
Rpin.on() # Right device
else:
Lpin.off()
Rpin.off()
Pinger += IcmpDir
if (Pinger>=Ping): # Repeat sequence
Pinger = 0
if (Pinger<0):
Pinger = Ping+IcmpDir
time.sleep(WaitTime) # pause
counter+=1
```

```
t1 = Thread(target=bump_watch) # run as separate thread
t1.start() # start bump watch thread
for i in range(4): # right-handed device
move_bump('F',-2,4104)
move_bump('R',-2,2052)
running = False
```

ДОДАТОК 11 – Файл reset.py

```
from RPi import GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(4, GPIO.IN, GPIO.PUD_UP)
while GPIO.input(4):
pass
print("Device powered off !")
if Rele == 'Off' or Rele == 'false' or Rele == '0':
Rpin.on() # Rele power off
else:
Lpin.off()
Rpin.off()
Pinger += IcmpDir
Return MAIN()
```