МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____С.В. Зибін
"____"_____2020 р.

ДИПЛОМНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
МАГІСТРА

Тема: " Автоматизована система керування та обліку мережевого обладнання"

Виконавець: Машкін Артем Михайлович

Керівниця: доцентка, к.т.н., доцентка Серебрякова Світлана Вікторівна

Нормоконтролер: доцент, к.т.н., доцент Радішевський Микола Федорович

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь магістр
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Програмне забезпечення систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Зибін С.В.
"___" _____ 2020 р

ЗАВДАННЯ
на виконання дипломної роботи студента
Машкіна Артема Михайловича

1. Тема дипломної роботи: «Автоматизована система керування та обліку мережевого обладнання»
затверджена наказом ректора від 21.09.2020 р. № 1715/ст.

2. Термін виконання проекту: з 5.10.2020 р. до 13.12.2020 р.

3. Вихідні дані до роботи: розробити програмний продукт для керування та обліку мережевого обладнання

4. Зміст пояснювальної записки:
   1. Огляд мережевого обладнання
   2. Аналіз систем керування та обліку мережевим обладнанням
   3. Встановлення вимог та проектування системи
   4. Програмна реалізація системи

5. Перелік обов'язкових слайдів презентації:
   1. Мета створення системи
   2. Вимоги до системи
   3. Функціональні можливості
   4. Архітектура системи
   5. Структурна схема системи
   6. Інтерфейс користувача

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1.	Розробка та затвердження графіка роботи..	05.10.2020-11.10.2020	
2.	Підготовка та написання 1 розділу	09.10.2020-18.10.2020	
3.	Підготовка та написання 2 розділу	19.10.2020-01.11.2020	
4.	Перший нормо-контроль 1-2 розділів	21.10.2020-23.10.2020	
	Підготовка та написання 3 розділу	02.11.2020-15.11.2020	
5.	Підготовка та написання 4 розділу	16.11.2020-29.11.2020	
6.	Редагування та друк пояснювальної записки, графічного матеріалу	30.11.2020-06.12.2020	
7.	Проходження нормо-контролю, перепліт пояснювальної записки. Отримання відгуку керівника. Підготовка презентації та тексту доповіді.	30.11.2020-06.12.2020	
8.	Попередній захист дипломної роботи. Підпис ПЗ завідувачем кафедри на допуск до захисту та для отримання рецензії.	07.12.2020-13.12.2020	
9.	Отримання рецензії. Для проходження контролю на плагіат здати секретарю ДЕК текст ПЗ одним файлом.	07.12.2020-13.12.2020	
10.	Здати секретарю ДЕК: ПЗ, ГМ, CD-R з електронними версіями ПЗ, ГМ, презентацію, відгук керівника, рецензію, довідку про успішність, 2 папки, 2 конверта)	07.12.2020-13.12.2020	
11.	Захист дипломної роботи перед ЕК	19.12.2020-21.12.2020	

Дата видачі завдання 17.10.2020 р.

Керівниця дипломної роботи: доц., к.т.н., доц. Серебрякова С.В.

Завдання прийняв до виконання: Машкін А.М.

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Автоматизована система керування та обліку мережевого обладнання» складається з: 86 сторінок, 30 рисунків, 3 таблиць, 14 використаних джерел, 1 додатка.

АВТОМАТИЗАЦІЯ, МЕРЕЖЕВЕ ОБЛАДНАННЯ, КЕРУВАННЯ, ОБЛІК, КОМП'ЮТЕРНА МЕРЕЖА, КОМУТАТОР, МАРШРУТИЗАТОР, PYTHON, HTML, CSS, JAVASCRIPT.

Об'єкт дослідження – мережеве обладнання, його керування та облік.

Предмет дослідження – автоматизована система керування та обліку мережевого обладнання.

Мета дипломної роботи – спрощення процесу керування різним мережевим обладнанням за рахунок автоматизації та уніфікації, покращення процесу обліку.

Методи дослідження – емпіричні, теоретичні методи дослідження, прикладне використання теоретичних навичок та знань.

У процесі роботи був виконаний аналіз інсуючих мережевих систем, визначено їхні переваги та недоліки, що стало у нагоді при розробці та проектуванні власної системи, її архітектури та складових.

Результати роботи можуть бути використані при розробці програмних засобів, призначених для автоматизації та уніфікації керування мережевим обладнанням та веденням його обліку.

Розробка та дослідження проводилися під ОС Windows 10. Розробка програми проводилася у середовищі Sublime Text 3 із використанням Python, HTML, CSS та Javascript.

ABSTRACT

Explanatory note to the thesis "Automated control and accounting management system of network equipment" consists of: 86 pages, 30 figures, 3 tables, 14 sources used, 1 appendix.

AUTOMATION, NETWORK EQUIPMENT, CONTROL, ACCOUNTING, COMPUTER NETWORK, SWITCH, ROUTER, PYTHON, HTML, CSS, JAVASCRIPT.

The object of research is network equipment, its management and accounting.

The subject of research is an automated control system and accounting of network equipment.

The purpose of the thesis - to simplify the process of managing various network equipment through automation and unification, improving the accounting process.

Research methods - empirical, theoretical research methods, applied use of theoretical skills and knowledge.

In the process, the analysis of intrusive network systems was performed, their advantages and disadvantages were identified, which was useful in the development and design of its own system, its architecture and components.

The results of the work can be used in the development of software designed to automate and unify the management of network equipment and its accounting.

Development and research were conducted under Windows 10. The program was developed in Sublime Text 3 using Python, HTML, CSS and Javascript.

ЗМІСТ

ВСТУП

Актуальність теми. Мережа Інтернет досить швидко розвивається. Створення локальних мереж та підключення їх до глобальної мережі стає обов’язковим, як для великих підприємств так і для домашніх користувачів. Зараз важко уявити яке-небудь підприємство без будь-якої обчислювальної техніки, адже вона полегшує виконання більшості сучасних задач. Обчислювальні машини, чи то звичайні смартфони або ноутбуки, чи то потужні сервери, обробляють та передають дуже багато інформації. Така комунікація можлива завдяки мережам зв’язку.

Комп’ютерні мережі, зокрема, складаються із середовища передачі даних, таких як: вита пара, оптоволоконний кабель, коаксіальний кабель, радіохвилі та активного або пасивного мережевого обладнання, наприклад комутатор, маршрутизатор.

Мережеве обладнання створює дуже велика кількість виробників, таких як Juniper, Cisco, HP, Extreme, Mikrotik та багато інших. Всі пристрої відрізняються між собою не тільки зовнішнім виглядом та ціною, але і підтримуваним функціоналом, інтерфейсом командного рядка або веб-інтерфейсу (за нявності). Щоб розбиратись в такій великій кількості мережевого обладнання та у підтримуваних ним технологіях, потрібно витратити досить багато часу на ознайомлення з технічною документацією або на проходження спеціальних курсів. Проте, навіть підготовленому фахівцю потрібно докласти чималих зусиль на конфігурацію того чи іншого обладнання, ситуацію також ускладнює кількість цього обладнання, від кількох десятків в межах якогось звичайного підприємства, до тисяч у випадках з інтернет провайдерами. Тому питання контролю та обслуговування мережевого обладнання є досить актуальним.

Наразі існують готові рішення у вигляді систем керування або обліку мережевого обладнання, проте жодна система не поєднує в собі обидві ці функції та має ряд інших недоліків. Тому було прийнято рішення про розробку власної системи, що відповідала б всім потребам та не мала недоліків інших систем.

Об’єкт дослідження – мережеве обладнання, його керування та облік.

Предмет дослідження – автоматизована система керування та обліку мережевого обладнання.

Мета дипломної роботи – спрощення процесу керування різним мережевим обладнанням за рахунок автоматизації та уніфікації, покращення процесу обліку.

Задачі дослідження:

1. Виконати огляд за тематикою дипломної роботи. Зокрема, проаналізувати сучасне мережеве обладнання.
2. Проаналізувати системи керування та обліку мережевим обладнанням.
3. Встановити вимоги та спроектувати систему керування та обліку мережевого обладнання.
4. Виконати програмну реалізацію системи керування та обліку мережевого обладнання.

Наукова значущість дипломної роботи полягає у застосуванні методів проектування та розробки програмного забезпечення, а також програмних методів до розробки нової системи керування та обліку мережевим обладнанням.

Практична цінність. Розроблений програмний продукт дозволить автоматизувати та уніфікувати процес управління мережевим обладнанням та полегшить завдання його обліку.

У процесі роботи був виконаний аналіз інсуючих мережевих систем, визначено їхні переваги та недоліки, що стало у нагоді при розробці та проектуванні власної системи, її архітектури та складових.

Методи та засоби дослідження – програмні методи розробки програмного забезпечення, Python, HTML, CSS та Javascript.

Результати дипломної роботи можуть бути використані для уніфікованого управління мережевими ресурсами.

РОЗДІЛ 1
ОГЛЯД МЕРЕЖЕВОГО ОБЛАДНАННЯ

1.1. Види мережевого обладнання

Мережеве обладнання – пристрої, що забезпечують функціонування комп'ютерних мереж, до них відносять: маршрутизатор, комутатор, концентратор, патч-панель тощо.

Маршрутизатор – мережеве обладнання, призначене для з'єднання двох або більше мереж та керування процесом маршрутизації, цей процес являє собою передачу пакетів мережевого рівня між різними частинами мережі, відповідно до певних правил та наявної інформації про топологію.

Зазвичай маршрутизатор не тільки займається передачею пакетів між інтерфейсами, а також виконує такі функції як: захист локальної мережі від загроз зовні, обмеження доступу до глобальної мережі користувачам локальної мережі, призначення ip адрес, шифрування трафіку та інше.

Маршрутизатори функціонують на мережевому рівні моделі OSI, вони мають здатність передавати дані між різними мережами, використовуючи таблицю маршрутів, що також називається таблицею маршрутизації, вона може містити статичні записи маршрутів або може вивчати їх за допомогою протоколів динамічної маршрутизації [1].

Окрім того маршрутизатори можуть виконувати перетворення адрес відправника та одержувача (NAT, Network Address Translation), на основі певних правил фільтрувати пакети даних, що передаються, з метою обмеження доступу, шифрувати їх та виставляти приорітет порядку передачі в залежності від типу.

Маршрутизатори не здійснюють передачу широкомовних повідомлень, таких як ARP-запит.

Навантаження мережі зменшують за рахунок маршрутизаторів. Як правило їх застосовують, щоб об'єднати мережі різних типів, у тому числі несумісних за архітектурою і протоколам, наприклад для об'єднання різних локальних мереж, а також надання їх доступу до мережі Інтернет.

Функції маршрутизатора може виконувати не тільки спеціалізоване апаратне забезпечення, а й звичайний комп'ютер. За допомогою спеціального програмного забезпечення, як правило для операційних систем UNIX, робочу станцію або сервер з декількома мережевими платами можна перетворити на маршрутизатор.

Маршрутизатори розрізняють за певними класами: верхній, середній та нижній. Маршрутизатори верхнього класу, також їх ще називаються магістральними (backbone routers) – мають найбільшу продуктивність та об'єднують мережі підприємства, таким чином створюючи центральну мережу, яка може складатися з великої кількості локальних. Таке обладнання здатне обробляти кілька сотень тисяч або навіть кілька мільйонів пакетів за секунду. Вони мають підтримку багатьох протоколів та інтерфейсів. Значна увага приділяється надійності та відмовостійкості, що забезпечується завдяки системам терморегуляції, резервним джерелам живлення, та модулям, що підтримують швидку заміну під час роботи (hot swap).

Середній клас маршрутизаторів використовується для регіональних відділень, що з'єднують їх між собою та з центральною мережею. Мережа регіонального відділення, подібно до центральної мережі, може складатися з декількох локальних мереж. Таке обладнання зазвичай дещо простіше за своїм функціоналом та можливості в порівнянні з маршрутизаторами високого класу. Вони так само можуть працювати з популярними транспортними протоколами та протоколами маршрутизації. Представників цього класу найбільше, їх функціонал може досягати до можливостей магістральних маршрутизаторів або зменшуватись до рівня обладнання віддалених офісів [1].

Маршрутизатори нижнього класу призначені для використання у віддалених офісах, вони підключають невеликі офіси до мережі підприємства. Такі маршрутизатори можуть підтримувати досить небагато інтерфейсів локальних мереж, вони розраховані на виділені лінії з низькою швидкістю. Також в якості резервного каналу може використовуватись телефонна лінія зв'язку. Ці маршрутизатори мають велику популярність в організаціях, яким необхідно розширити чинні міжмережеві зв'язки. Є дуже багато типів маршрутизаторів віддалених офісів. Це обумовлено як великою кількістю потенційних споживачів, так

і спеціалізацією такого типу пристроїв, що проявляється в підтримці одного конкретного типу глобального зв’язку.

Комутатор – це мережевий пристрій, який з’єднує кілька комп’ютерів в одну єдину локальну мережу. Сучасні комутатори мають дуже великий ряд функцій, які дуже сильно можуть полегшити подальшу роботу адміністратора. Від правильного вибору комутаторів залежить функціонування всієї локальної мережі й робота підприємства в цілому.

Комутатор функціонує на канальному (другому) рівні моделі OSI. Комутатори були створені для використання мостових технологій і досить часто розглядаються в якості мостів з великою кількістю портів. Комутатор передає дані лише безпосередньо одержувачеві, окрім широкомовного трафіку, що передається всім вузлам мережі, в той час як концентратор, відправляє трафік від одного підключеного пристрою до всіх інших, не зважаючи на тип трафіку. Це дозволяє підвищити продуктивність і безпеку мережі, позбавляючи інші вузли мережі необхідності обробляти непризначені їм дані.

Комутатор має таблицю комутації, яку зберігає в асоціативній пам’яті, у ній міститься записи відповідності MAC-адреси вузла до порта комутатора. Оскільки після включення комутатора таблиця комутації порожня, він спочатку працює в режимі навчання, передаючи данні на всі інші порти, окрім того з відки вони надійшли. При цьому комутатор аналізує отримані фрейми (кадри), визначаючи MAC-адресу хоста-відправника та записує його до таблиці на певний час.Потім, якщо на один з портів комутатора надійде кадр, призначений для хоста, MAC-адреса якого уже є в таблиці, то такий кадр буде відправлено тільки на той порт, що зазначений у таблиці [1].

Комутатори поділяють на такі, що:

– не мають можливості управління – це прості самостійні пристрої, які самі управляють передачею даних і не мають функцій ручного керування. Деякі моделі некерованих комутаторів мають вбудовані засоби моніторингу. Недоліками таких комутаторів є відсутність знову ж таки засобів управління і низька продуктивність. Зважаючи на це, у великих мережах підприємств некеровані комутатори не

використовуються, оскільки адмініструвати таку мережу досить складно і потрібно витрачати досить багато зусиль;

– керовані комутатори, вони також працюють в автоматичному режимі, але крім цього мають ручне керування. Ручне керування надає можливість дуже гнучко налаштувати роботу комутатора та полегшити життя системного адміністратора. Головним недоліком таких комутаторів є ціна, яка залежить від функціонала і продуктивності [1].

1.2. Мережеве обладнання компанії Juniper

1.2.1. Маршрутизатори серії MX

Лінійка MX – флагманський продукт компанії. На рис. 1.1. показаний весь модельний ряд, - шість пристроїв, від молодшої моделі до старшої, а також віртуальний пристрій vMX.

Рис. 1.1. Серія MX. Мультісервісні маршрутизатори кордону мережі

За рахунок високої універсальності MX можуть використовуватися в найрізноманітніших сценаріях роботи - для управління передплатниками широкосмугового доступу і в Enterprise-сценаріях як VPN концентратор. За рахунок

дуже якісної multicast-підтримки MX добре підходять для розподілу відеопотоку. Ще один варіант використання - сервісний шлюз. MX дозволяють організувати Firewall і NAT (Network Address Translation).

Класична модель розгортання сервісів передбачає, що під кожен сервіс використовується свій апаратний пристрій. Якщо використовувати MX, все це можна поєднати. Один з найбільш затребуваних сценаріїв використання MX в операторському сегменті - це BNG або Bras, - все, що пов'язано з авторизацією абонентів та обліком трафіку [2].

На сьогодні функціонал vMX не повному обсязі відповідає функціоналу апаратного MX. Найближчим часом паритет буде досягнутий, і розробка нового функціоналу буде йти паралельно, - тобто, все, що буде розроблено для «великих» фізичних MX, буде перенесено і на віртуальний MX.

1.2.2. Маршрутизатори серії ACX

Маршрутизатор серії ACX призначені для вирішення завдань рівня доступу.Вони використовуються для побудови таких мереж, як Mobile Backhole, або для реалізації технологічних мереж, що зв'язують енергетичне обладнання (рис. 1.2.).

Interface	ACX1000	ACX1100	ACX2000	ACX2100	ACX4000[1]
TDM – T1/E1	8	-	16	16	-
GbE (Copper)	8	4	8 (PoE+ on 2 ports)	4	8 (PoE+ on 2 ports)
GbE combination Copper or Fiber (SFP)	4	4	-	4	-
GbE SFP	-	4	2	2	2
10GbE (SFP+)	-	-	2	2	2

Рис. 1.2. Лінійка маршрутизаторів серії ACX

Відповідно, ACX повинні працювати в широкому температурному діапазоні. Є варіанти в захищеному виконанні, які можуть працювати в складних умовах.

Перевагою рішення є те, що можна налаштовувати цілком прозору End-to-End MPLS структуру, з усіма її перевагами - управлінням сервісами, QoS і синхронізацією (при міграції з традиційною, класичною SDH, PDH на сучасні пакетні мережі).

Устаткування повністю підтримує синхронний Ethernet. При роботі за протоколом PTP досягається рівень якості SDH 50 мс і менше. Можна створювати і досить великі, масштабовані мережі Metro Ethernet.

Для маршрутизаторів серії ACX, з точки зору окупності інвестицій (ROI), особливістю є те, що вони ліцензуються за кількістю портів, яке може збільшуватися (і, відповідно, оплачуватися) у міру розширення мережі [2].

1.2.3. Комутатори серії EX

Комутатори серії EX призначені в основному для сегмента Enterprise, для рішень рівня кампусних мереж (рис. 1.3).

Рис. 1.3. Асортимент і класифікація за призначенням комутаторів серії EX

Комутатор EX4300 може застосовуватися як в Entrprise-кампусах, так і в датацентрах. Це - універсальне рішення, відмовостійке і масштабується.

Комутатори можуть бути зібрані в Virtual Chassis за допомогою будь-яких інтерфейсів. Цей пристрій має 4 x 40-Gbps інтерфейса, - в залежності від того, який трансивер QSFP (Quad Small Form-factor Pluggable) використовується. Virtual Chassis може працювати на відстані до 80 км. Крім цього, Virtual Chassis може бути змішаним, об'єднуючи 1 Gbps і 10 Gbps пристрої.

Серія EX починається з гігабітних комутаторів (Juniper вийшла на ринок в 2008 р і відразу запропонувала 1 Gbps, минаючи 100 Mbps).

Рішення на базі EX добре масштабуються, починаючи від фіксованих і закінчуючи модульними комутаторами. Останні виконані на базі маршрутизаторів MX, але мають дещо інше ПО і значно меншу ціну. Функціонал апаратної частини адаптований для побудови комутаторів як ядра кампусів і датацентрів. У структурі пропозицій Juniper відповідне місце займають також лінійні карти серії EX [2].

1.3. Мережеве обладнання компанії Cisco

1.3.1. Рішення для малого і середнього бізнеса

Маршрутизатор Cisco 1800.

Маршрутизатор з інтегрованими сервісами Cisco 1800 Series (ISR) об'єднують сервіси передачі даних, забезпечення безпеки і бездротового зв'язку в єдиному пристрої, яке доступне в будь-якій точці, де діє ваше підприємство (рис. 1.4.).

Рис. 1.4. Маршрутизатор Cisco 1800

Особливості.

Маршрутизатор з інтегрованими сервісами Cisco 1800 Series підтримують такі функції:

– гнучкі сервіси, що розвиваються разом з вашим бізнесом. Гнучкість, яка дозволяє почати з 50 і менше підключень віртуальних приватних мереж і розширювати їх до 800 підключень. Модульні можливості маршрутизаторів Cisco 1841 і Cisco 1861 забезпечують гнучкий вибір інтерфейсів WAN, варіантів послуг і багато чого іншого.

– інтегроване резервне копіювання WAN. Маршрутизатор 1800 з фіксованою конфігурацією також забезпечують інтегроване резервне копіювання WAN і балансування навантаження за допомогою аналогового модему V.92 (Cisco 1811) або інтерфейсу BRI для ISDN S / T (Cisco 1812).

– бездротові мережі. Допомагають співробітникам бути більш продуктивними і краще здійснювати спільну роботу завдяки доступу до додатків і інформації з будь-якої робочої точки. Інтегрована функція захищених бездротових підключень забезпечує підтримку ряду стандартів для бездротових мереж.

– голосовий зв'язок. На деяких моделях (Cisco 1861) можна використовувати розширені засоби зв'язку, такі як обробка викликів, голосова пошта, автосекретар і конференції, що дозволяє швидше реагувати на заявки клієнтів і економити додаткові витрати на віддалену зв'язок.

– безпека. Вбудовані системи безпеки: міжмережевий екран, шифрування і захист від зломщиків зменшують бізнес-ризики, пов'язані з вірусами та іншими загрозами безпеці.

– віртуальні приватні мережі. Надає віддаленого персоналу і надомною працівникам безпечний доступ до активів компанії за допомогою захищеного з'єднання [3].

До складу сімейства Cisco 1800 Series ISR входить сім моделей з кількома варіантами мереж для локальних і віддалених підключень, які надають функції для гнучкого розширення бізнесу. Дані маршрутизатори ідеально підходять для мереж малих і середніх компаній, а також для невеликих філій підприємств. Пристрої

дозволяють компаніям знизити витрати завдяки єдиній відмовостійкої системи, що забезпечує надання ряду найважливіших ділових послуг, таких як:

– передача даних;

– безпека;

– уніфіковані комунікації;

– бездротові мережі.

Інтегровані сервіси маршрутизаторів Cisco серії 1800 забезпечують необхідні можливості і гнучкість для надання захищеного доступу до мереж Інтернет і інтранет. Всі моделі цієї серії мають наступні можливості:

– вбудована система безпеки;

– до двох вбудованих маршрутизованих портів 10/100 Мбіт / с.

Моделі з фіксованою платформою (1801, 1802, 1 803, 1 811, 1 812) підтримують різні інтерфейси розподілених мереж, а також:

– швидкості передачі до рівня широкосмугових мереж;

– 8 вбудованих комутованих портів 10/100 Мбіт / с з можливістю харчування по кручений парі (PoE), що дозволяє подавати постійний струм на різні мережеві пристрої - наприклад, IP-телефони;

– до 50 тунелів віртуальних приватних мереж;

– підтримка стандартів бездротових локальних мереж 802.11a / b / g.

Cisco 1841 - це модульна платформа з широким набором підтримуваних інтерфейсів, а також:

– швидкість передачі до T1 / E1;

– до чотирьох вбудованих портів комутації 10/100 Мбіт / с;

– до 800 тунелів віртуальних приватних мереж;

– підтримка стандартів бездротових локальних мереж 802.11a / b / g.

Cisco 1861 - це модульна платформа, побудована на фіксованій основі і підтримуюча широкий набір інтерфейсів, а також:

– інтегровані рішення Cisco Unified Communications Manager Express або Cisco Unified Survivable Remote Site Telephony, що забезпечують обробку викликів;

– рішення Cisco Unity Express, що забезпечує обмін голосовими повідомленнями і роботу автоматичного секретаря;

– вбудований комутатор локальних мереж з підтримкою живлення по витій парі (PoE) і можливістю розширення за допомогою комутаторів Cisco Catalyst;

– вбудовані порти голосового зв'язку для підключення ТфОП, УАТС і офісних міні-АТС [3].

Комутатори Cisco Catalyst 3560.

Комутатори сімейства Cisco Catalyst 3560 - це лінійка комутаторів корпоративного класу з фіксованою конфігурацією (передбачена можливість живлення пристроїв повитій парі (PoE), сумісного зі специфікацією IEEE 802.3af).

Вони призначені для локальних мереж невеликих підприємств або віддалених офісів і відмінно підходять на роль комутатора рівня доступу. Конфігурація з портами 10/100/1000, що підтримує живлення пристроїв по кручений парі (PoE), пропонує максимальну ефективність і захист інвестицій, надаючи в той же час можливість розгортання нових додатків, таких як IP-телефонія, бездротовий доступ, охоронні відеосистеми і системи управління будівлями (рис. 1.5.).

Рис. 1.5. Комутатори Cisco Catalyst 3560

Комутатори Cisco Catalyst 3560, не ускладнюючи традиційну інфраструктуру комутованих локальних мереж, дозволяють впровадити в масштабі всієї мережі такі

інтелектуальні функції, як розширене управління якістю обслуговування (QoS), обмеження швидкості, списки контролю доступу (ACL), управління груповий передачею і високопродуктивна IP-маршрутизація .

Безкоштовно розповсюджується програмне додаток Cisco Network Assistant для централізованого управління комутаторами сімейства Catalyst 3650 спрощує адміністрування комутаторів, маршрутизаторів і бездротових точок доступу Cisco, пропонуючи "майстри конфігурації" для зручності впровадження конвергентних мереж і послуг інтелектуальної мережі.

Комутатори сімейства Cisco Catalyst 3560 можуть поставлятися зі стандартною (SMI) або розширеної (EMI) версіями програмного забезпечення. У набір функцій SMI входить: підтримка розширених функцій управління якістю обслуговування (QoS), обмеження швидкості, підтримка списків контролю доступу (ACL) і базові функції маршрутизації. Конфігурація EMI передбачає додатковий набір функцій корпоративного класу, включаючи апаратно реалізовані функції маршрутизації одноадресних і мультикастових IP-пакетів, а також маршрутизацію на базі політик (PBR) [3].

1.3.2. Рішення для великих підприємств

Маршрутизатор Cisco 7200.

Модульне обладнання Cisco серії 7200, володіє відмінними характеристиками щільності, продуктивності, а так само підтримкою сервісів.

Рис 1.6. Маршрутизатори Cisco 7200

Маршрутизатор Cisco серії 7200 є найбільш використовуваними універсальними рішеннями для агрегації сервісів (див. рис. 1.6.).

Маршрутизатор Cisco серії 7200 мають наступні можливості:

– чудове співвідношення ціна / якість: новий процесор NPE-G2 Network Processing Engine надає можливість агрегації сервісів на швидкостях до 2 млн. Пакетів / с;

– широкий набір підтримуваних з'єднань, легкість в управлінні і обслуговуванні;

– збільшена продуктивність віртуальних приватних мереж (VPN) завдяки новому адаптера сервісів VPN;

– збільшена масштабованість і гнучкість завдяки новій карті адаптера портів.

Переваги:

– граничний сегмент розподілених мереж: відзначена нагородами продуктивність системи управління якістю обслуговування (QoS);

– агрегація широкосмугових мереж: До 16 000 сесій PPP для кожного шасі;

– мультипротокольна комутація на основі міток (MPLS): Найбільш популярний вибір для реалізації граничних сегментів мереж провайдерів послуг;

– віртуальні приватні мережі на базі IPsec: Масштабується до 5 000 тунелів для кожного шасі;

– яке встановлюється у клієнта обладнання (CPE) вищого рівня;

– підтримка шлюзів IP-to-IP: забезпечує точку взаємодії між мережами сигналізації (H.323, SIP), середовищами передачі, трансляцію адрес і номерів портів (забезпечення конфіденційності і приховування топології), функції тарифікації та нормалізації записів про виклики (CDR), а також управління пропускною спроможністю (маркування системи управління якістю обслуговування з використанням TOS);

– інтеграція передачі голосу, відео і даних: Шасі VXR і адаптери голосових портів з підтримкою TDM;

– модульна архітектура: форм-фактор 3RU і широкий набір гнучких модульних інтерфейсів (від DS0 до OC-3);

– гнучкість: Підтримка Fast Ethernet, Gigabit Ethernet, Packet over SONET і ін [3].

Комутатори Cisco Catalyst 4500.

Серія Cisco Catalyst 4500 - це високопродуктивна модульна платформа середнього рівня від Cisco, що забезпечує захищений, гнучкий і постійний обмін даними для підприємств і провайдерів послуг будь-якого розміру.

Дані комутатори побудовані на архітектурі високої продуктивності, що забезпечує високу швидкість роботи послуг, легкість в експлуатації, сумісність з більш старим і більш новим обладнанням, а також надійний захист інвестицій та їх швидку окупність (рис. 1.7.).

Рис. 1.7. Комутатори Cisco Catalyst 4500

Модульні комутатори середнього класу Cisco Catalyst серії 4500 пропонують можливості неблокируемой комутації 2-4 рівня для великих корпоративних мереж, мереж підприємств малого та середнього бізнесу (SMB) і міських мереж Metro Ethernet.

Cisco Catalyst серії 4500 дозволяє охопити функціями контролю граничні сегменти мережі завдяки інтелектуальним мережевих сервісів, що володіє такими властивостями, як передбачувана продуктивність, розвинені засоби безпеки і інтегрований механізм забезпечення відмовостійкості - як на програмному, так і на апаратному рівнях.

Переваги обладнання Cisco Catalyst серії 4500:

– максимальний захист інвестицій: зворотна сумісність, легкість модернізації керуючого модуля, розширення можливостей на всіх портах;

– висока доступність: відмовостійкість апаратного і програмного забезпечення скорочує планові та позапланові простої мережі;

– легкість у використанні і управлінні: Web-інтерфейс управління дозволяє централізовано налаштовувати обладнання, спрощуючи його експлуатацію;

– комплексна система безпеки: розвинені засоби безпеки дозволяють перешкодити крадіжці інформації та обмежити масштаби збитку від впливу різних вірусів і черв'яків;

– живлення пристроїв по кручений парі: передача по кручений парі до 30 Вт дозволяє забезпечити роботу новітніх потужних кінцевих пристроїв;

– масштабованість: неблокована система з можливістю обробки до 250 млн. Пакетів / с незалежно від числа підключених розширених сервісів;

– гнучкість розгортання: wя серія пропонує багатий вибір шасі, високопродуктивних модулів управління, лінійних карт і джерел живлення [3].

Висновки

В першому розділі розглянуто загальне поняття мережевого обладнання, які бувать його види та які функції воно виконує. Наведено приклади мережевого обладнання двох популярних виробників Juniper та Cisco, представлено рішення для малих та великих підприємств, наведено характеристики та можливості кожного з продуктів. Таке велике різноманіття мережевого обладнання потребує універсального рішення для управління ним, а оскільки на підприємствах кількість пристроїв не обмежується декількома, то подрібно також вести його облік.

РОЗДІЛ 2
АНАЛІЗ СИСТЕМ КЕРУВАННЯ ТА ОБЛІКУ МЕРЕЖЕВИМ ОБЛАДНАННЯМ

2.1. Протоколи управління мережевим обладнанням

Як правило система управління мережею представляє собою додаток високого рівня, що використовує один зі стандартних протоколів мережевого управління – telnet, ssh, snmp.

Головні задачі системи управління наступні:

– забезпечення високої продуктивності мережі;

– забезпечення зручного середовища для управління мережевими ресурсами;

– збір інформації про стан усіх мережевих пристроїв;

– аналіз і зберігання інформації про стан усіх мережевих пристроїв;

– прогнозування збоїв в роботі мережі.

Протоколи управління (або комунікаційні протоколи) відносяться до протоколів прикладного рівня семирівневої моделі взаємодії відкритих систем [4].

2.1.1. Протокол telnet

Telnet – протокол емуляції терміналу, що часто використовується в мережі Інтернет та в мережах, що працюють по протоколах, заснованим на TCP / IP. Це дозволяє користувачеві терміналу або персонального комп'ютера, реєструватися в системі віддаленого комп'ютера і виконувати програми. Протокол telnet був спочатку розроблений для ARPAnet і є важливою складовою протоколу передачі даних TCP / IP.

Хоча більшість комп'ютерів в мережі Інтернет надають доступ за допомогою протоколу telnet тільки для користувачів, у яких є діючий обліковий запис і пароль, існують також деякі системи, які надають доступ до своєї мережі без аутентифікації користувача, щоб запускати і користуватися такими програмами як утиліти пошуку.

Telnet – клієнт-серверний протокол, побудований на TCP, і клієнти зазвичай використовують портом 23 для підключення до віддаленого комп'ютера, про те номер

цього порта завжди можна змінити. Частково через конструкції протоколу і частково через гнучкість, зазвичай програмами, що постачають telnet, можна використовувати програму telnet, щоб встановити інтерактивне підключення TCP з деякою іншою послугою віддаленого комп'ютера. Типовим прикладом такого використання клієнтської частини протоколу може стати передача даних за допомогою функції програми telnet з портом 25 віддаленого комп'ютера, щоб налаштувати поштовий серверх [5].

Потрібно зазначити, що сам протокол telnet має досить низький рівень захисту небезпечний, і потрібно уникати його застосування. Використовуючи цей протокол в загальнодоступних місцях можна зазнати суттєвих проблем безпеки.

Існує три основних проблеми щодо використання протоколу telnet, що робить його поганим варіантом для сучасних систем в плані безпеки:

– демони telnet, що як првило використовуються, мають декілька вразливостей;

– telnet не забезпечує шуфрування жодних даних, які передаються під час встановленого зв'язоку (у тому числі паролі), через це можна отримати дані аторизації для майбутнього використання в зловмисних цілях;

– відсутня система аутентифікації, а тому telnet ніяк не гарантує, що зв'язок, встановлена між двома віддаленими хостами не буде перерваний в середині.

Ці недоліки привели до дуже швидкої відмови від використання протоколу telnet на користь більш безпечного і функціонального протоколу SSH, описаного в 1998р. SSH надає всі ті функціональні можливості, які представлялися в telnet, з додаванням ефектного кодування з метою запобігання перехоплення таких даних, як логіни і паролі. Введена в протоколі SSH система аутентифікації з використанням публічного ключа гарантує, що віддалений комп'ютер дійсно є тим, за кого себе видає.

Експерти комп'ютерної безпеки, як наприклад SANS Institute і члени телеконференції comp.os.linux.security рекомендують відмовитися від використання протоколу telnet для віддаленого доступу при всіх нормальних обставинах.

Коли telnet розвивався в ранніх 1980-их (згідно з деякими джерелами в 1969), більшість призначених для користувача комп'ютерів в мережі було в комп'ютерних відділах академічних установ, або у великих приватних і урядових науково-дослідних

інститутах. У цьому середовищі багато підприємств не були особливого стурбовані захистом до різкого збільшення пропускної здатності 1990-их. З ростом числа користувачів мережі Інтернет, також різко збільшилася кількість людей, які намагаються зламати сервери цієї мережі. Як наслідок протокол telnet не повинен використовуватися в мережах, що мають доступ в Інтернет.

Клієнти telnet до сих пір іноді використовуються для того, щоб вручну «спілкуватися» з деякими іншими сервісами. Наприклад це іноді буває корисним для налагодження мережевих служб типу SMTP або HTTP серверів, тому що цей спосіб робить простим відправку команд на сервер і вивчення відповідей сервера на ті чи інші команди. Telnet може також використовуватися як елементарний IRC клієнт, якщо Ви досить добре знаєте протокол [5].

2.1.2. Протокол SSH

Парадокс мережі Інтернет полягає в тому, що вона ніколи б не розвинулася, якби не була відкритою. Але подібного роду відкритість робить її вразливою для різного роду атак. Протокол SSH був тим рішенням минулого десятиліття, яке створювалося і розвивалося з метою вирішити цей парадокс.

Для непосвячених людей SSH являє собою певну програму, яка дозволяє встановлювати з'єднання з віддаленим комп'ютером, виконувати на ньому деякі команди і переміщати файли між комп'ютерами. Надаючи надійну систему аутентифікації і можливість забезпечення стійкого алгоритму шифрування даних, що передаються по відкритих каналах (таким як мережу Інтернет) дозволила SSH замінити менш безпечні програми емуляції терміналу, такі як telnet, rsh, rlogin та ін.

Будь-який системний адміністратор чудово знайомий з програмами емуляції терміналу (ssh, rsh, telnet та ін.), а також з протоколами, за допомогою яких вони взаємодіють. Насамперед, після установки операційної системи, системний адміністратор вважає своїм обов'язком замінити використовувані за замовчуванням небезпечні програми віддаленого доступу, такі як telnet, rlogin і інші, на SSH.

Протокол SSH не вирішує всіх проблем мережевої безпеки. Він лише зосереджує свою увагу на забезпеченні безпечної роботи таких додатків, як

емулятори терміналу (проте цими лише функціями протокол не обмежується, він дозволяє встановлювати безпечні з'єднання і для інших цілей). Використання протоколу SSH на серверах і в клієнтських додатках дозволяє захистити дані лише в процесі передачі; протокол SSH ніяк не замінює брандмауери, системи виявлення вторгнень, мережевих сканерів, систем аутентифікації та іншихзасобів, що дозволяють захистити інформаційні системи і мережі від атак [5].

Важливо відзначити, що самі по собі протоколи SSH-1 і SSH-2 різні. Тобто якщо клієнт запитує з'єднання з сервером протоколу SSH-1, а сервер підтримує тільки протокол SSH-2, то з'єднання встановлено не буде. Ця особливість пов'язана з технічною реалізацією цих протоколів, які, за великим рахунком, виконують одні і ті ж функції. Більш того використання протоколу SSH-1 швидше за все впаде на системного адміністратора у вигляді зайвого головного болю, ніж надасть реальний захист переданих між системами даних.

Протокол SSH розроблявся для надання безпеки переданих даних шляхом реалізації стійкого алгоритму шифрування даних, надійної системи аутентифікації користувача і сервера, наданням системи контролю цілісності переданих даних, а також інкапсуляцією додатків, що працюють на основі протоколу TCP для встановлення безпечних тунелів.

Опис роботи протоколу SSH-1.

Спочатку клієнт надсилає серверу запит на встановлення SSH з'єднання і створення нового сеансу. З'єднання буде погоджено сервером, якщо він приймає подібні повідомлення і готовий до встановлення нового сеансу зв'язку. Потім клієнт і сервер обмінюються інформацією, щодо підтримуваних версій протоколів. З'єднання буде продовжено, якщо буде знайдено відповідність між протоколами і отримано підтвердження про готовність обох сторін продовжити з'єднання за даним протоколом. Відразу після цього сервер відправляє клієнтові постійний публічний і тимчасовий серверні ключі. Клієнт використовує ці ключі для шифрування ключа сесії. Хоча тимчасовий ключ надсилається у незахищенову вигляді, сесійний ключ і досі безпечний. Після цього сесійний ключ шифрується тимчасовим ключем і публічним ключем сервера і, таким чином, тільки сервер може його розшифрувати.

На цьому етапі і клієнт і сервер мають сесійний ключ і, отже, готові до безпечного сеансу передачі зашифрованих пакетів.

Аутентифікація сервера проходи з урахуванням його можливостей розшифровки сесійного ключа, який зашифрований публічним ключем сервера. Аутентифікація клієнта може відбуватися різними способами, в тому числі DSA, RSA, OpenPGP або за допомогою паролю.

Сесія триває до тих пір, поки і клієнт і сервер здатні аутентифікувати один одного. Встановлене з'єднання по протоколу SSH-1 дозволяє захистити дані, що передаються стійким алгоритмом шифрування, перевіркою цілісності даних і стисненням [5].

Опис технології протоколу SSH-2.

Обидва протоколи, по суті, виконують одні і ті ж функції, але протокол SSH-2 робить це більш елегантно, більш безпечно і більш гнучко. Головна відмінність між протоколами полягає в тому, що протокол SSH-2 розділяє всі функції протоколу SSH між трьома протоколами, в той час як протокол SSH-1 представляє собою один єдиний і неподільний протокол. Модуляцією функцій протоколу SSH в трьох протоколах - протоколі транспортного рівня, протоколі аутентифікації і протоколі з'єднання, робить протокол SSH-2 найбільш гнучким і потужним механізмом створення безпечних тунелів. Нижче надано короткий опис і призначення кожного з трьох протоколів, що складають протокол SSH-2:

– протокол транспортного рівня, за допомогою нього можна шифрувати та стикувати дані, такаож забезпечити систему контролю цілісності даних;

– протокол з'єднання – дає можливість клієнтам встановлювати багатопоточні з'єднання за допомогою оригінального SSH тунелю, зменшуючи при цьому навантаження, яке створюють клієнтські процеси;

– протокол аутентифікації – протокол аутентифікації відокремлений від протоколу транспортного рівня, тому що не завжди потрібно використовувати системи аутентифікації. У разі, необхідності аутентифікації, процес захищається оригінальним безпечним каналом, встановленим через протокол транспортного рівня.

Протокол транспортного рівня є достатнім для встановлення захищеного з'єднання, він є основою протоколу SSH-2 і протоколи узгодження та аутентифікації засновані на ньому. Протокол аутентифікації відділений від протоколу транспортного рівня, тому що іноді виникає ситуація, коли використання аутентифікації не тільки не обов'язково, але і навіть небажано. Наприклад, якась організація надає на своєму FTP сервері анонімний доступ до патчів безпеки для будь-якої людини (або системи), яка захоче їх завантажити. В цьому випадку аутентифікація не буде потрібна, в той час як шифрування, стиснення і контроль цілісності даних будуть забезпечуватися протоколом транспортного рівня. Більш того, при наявності каналу високої пропускної здатності, клієнти зможуть організувати багатопотокове з'єднання через оригінальне SSH з'єднання, використовуючи протокол з'єднання [5].

2.1.3. Протокол SNMP

SNMP відноситься до протоколів прикладного рівня, призначеним для полегшення обміну інформацією управління між мережевими пристроями. Користуючись інформацією SNMP (такий, як показник числа пакетів в секунду і коефіцієнт мережевих помилок), мережеві адміністратори можуть більш просто управляти продуктивністю мережі, виявляти і вирішувати мережеві проблеми.

Програма користувача, звана мережевим менеджером, здійснює віртуальні з'єднання з програмою, яка називається SNMP-агентом. SNMP-агент розташований на віддаленому мережевому пристрої та надає інформацію менеджеру про стан даного пристрою. SNMP-агенти роблять інформацію доступною для систем управління мережами (Network Management Systems, NMS) за допомогою SNMP, ця модель представлена на рис. 2.1.

Керований пристрій, на якому функціонує програма-агент, може бути будь-якого типу - наприклад, сервер доступу в Інтернет, принтери, маршрутизатори, концентратори, комутатори і т.п. У даній ситуації програми управління повинні бути побудовані таким чином, щоб мінімізувати вплив програми-агента на керований пристрій [4].

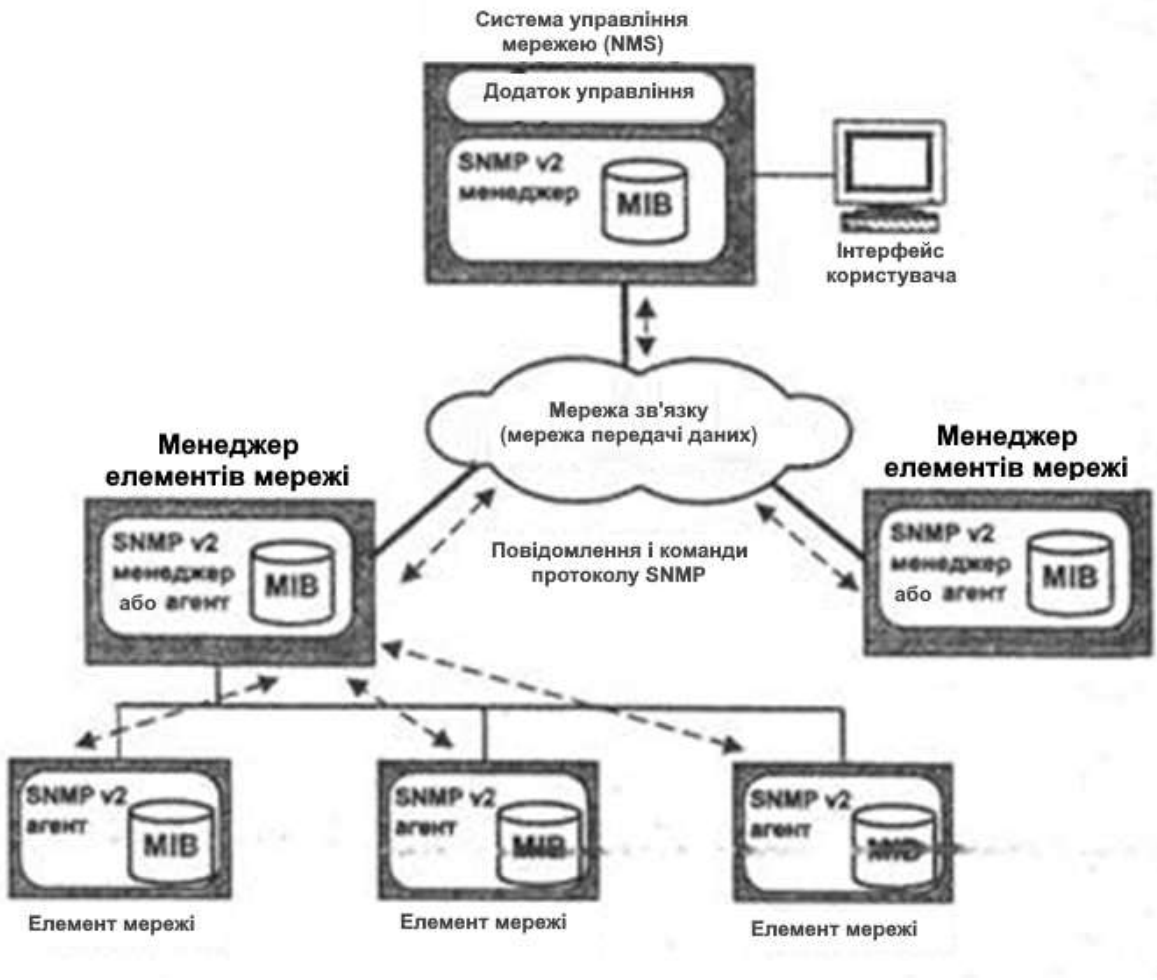

Рис. 2.1. Використання протоколу SNMP

Програми-агенти за завданням менеджера або автоматично можуть відстежувати такі показники роботи мережевого устаткування:

– число і стан своїх віртуальних каналів;

– число певних видів повідомлень про несправності;

– число байтів і пакетів, що входять і виходять з даного пристрою;

– максимальна довжина черги на виході (для маршрутизаторів і інших пристроїв);

– відправлені і прийняті широкомовні повідомлення;

– відмовили і знову пущені в експлуатацію мережеві інтерфейси.

За допомогою протоколу SNMP можна оцінити продуктивність мережевих пристроїв, кількість вільних ресурсів, завантаженість, а також визначити безліч інших

корисних характеристик, необхідних для управління мережевими пристроями. SNMP - це протокол типу «запит-відповідь» (тобто на кожен запит менеджера повинен бути переданий відповідь агента).

SNMP визначає всього п'ять типів повідомлень, якими обмінюються менеджер і клієнт. Команда Get-request застосовується менеджером для отримання від агента значення об'єкта по імені. Команда GetNext-request застосовується менеджером, щоб отримати значення наступного об'єкта при послідовному обході MIB. За допомогою команди Get-response агент SNMP передає менеджеру результати перерахованих вище команд. Команда Set встановлює значення об'єкта, а команда Trap повідомляє менеджеру про виникнення будь-якої нестандартної ситуації. Крім того, в SNMP версії 2 додана команда GetBulk, за допомогою якої менеджер може отримати кілька значень змінних за один запит.

База даних з інформацією про стан елементів мережевого обладнання та керована SNMP-агентом називається базою інформацією управління SNMP (Management Information Base, MIB). Просто кажучи MIB – віртуальний інформаційний масив, який, подібно до класичної базі даних, містить в формалізованому та впорядкованому вигляді всі дані, пов'язані з мережею зв'язку, з мережевим обладнанням в будь-якій частині мережі, і є інформаційною моделлю керованого об'єкта [4].

У протоколі SNMP можна виділити наступні основні стандартні елементи:

– стандартний формат повідомлення (standard message format), який визначається форматом повідомлення UDP. Ця частина стандарту високого рівня має невелике число прикладних користувачів (але викликає інтерес у більшості програмістів, що використовують SNMP);

– стандартний набір керованих об'єктів (standard set of managed objects) являє собою набір стандартних величин (values) - об'єктів SNMP, до яких можна адресувати запити від різних пристроїв. Стандарт включає величини для поточного контролю TCP, IP, UDP і інтерфейсів пристроїв. Кожен керований об'єкт асоціюється з офіційним ім'ям, а також з числовим ідентифікатором, що має в запису імені точку (dot-notation);

– стандартний спосіб додавання об'єктів (standard way of adding objects). Мабуть, наявність цього елемента - одна з причин того, що SNMP став широко відомим і набув статусу фактично промислового стандарту. Причина в тому, що наявність такого способу дозволяє постачальникам мережевих пристроїв розширювати стандартний набір об'єктів управління (що обговорювалося вище) за допомогою специфікації нових об'єктів управління для розгорнутих мереж.

В цілому, починаючи з версії 1 SNMP, були визначені чотири типи стандартних операцій управління:

– операція Get (отримати) застосовується, щоб повернути (отримати) значення поіменованого об'єкта управління;

– операція GetNext (отримати наступний) існує, щоб повернути ім'я (і значення) наступного по порядку керованого об'єкта, який відповідає певному мережевому пристрою і має коректно присвоєне ім'я SNMP;

– операція Set (встановити) застосовується, щоб встановити поіменованим об'єктів специфічні значення (точніше, щоб змінити значення ідентифікаторів або атрибутів керованих об'єктів);

– операція Trap (переривання) використовується пристроями асинхронно; за допомогою даного переривання мережеві пристрої можуть повідомляти адміністратора мережі про проблеми, що виникли в цьому пристрої поза режимом опитування даного пристрою.

Кожній перерахованої команді відповідає PDU певного формату. За допомогою перерахованих типів операцій (команд) можна сформувати відповідні запити від менеджера до агента і від агента до менеджера.

Крім перерахованих, у другій версії SNMP додані нові повідомлення, а саме:

– операція GetBulk використовується для вилучення великого масиву даних, а не одиничних значень;

– операція Inform дозволяє одній NMS виконувати операцію Trap на інший NMS і отримувати відповідь;

– операція Report передає повідомлення агента про стан керованого ресурсу без запиту.

Недоліки протоколу SNMP:

– відсутність засобів взаємної аутентифікації агентів і менеджерів. Єдиний засіб ідентифікації - «рядок спільноти» - «community string». Рядок передається у відкритій формі (до 3-ої версії протоколу) в повідомленні SNMP і служить основою для поділу агентів і менеджерів на «спільноти», так що агент взаємодіє тільки з тими менеджерами, які вказують в поле community string той же символьний рядок, що і зберігається в пам'яті агента. Це, безумовно, не спосіб аутентифікації, а спосіб структурування агентів і менеджерів;

– робота через ненадійний протокол UDP (переважна більшість реалізації агентів SNMP) призводить до втрат аварійних повідомлень (повідомлень trap) від агентів до менеджерів, що може привести до неякісного управління [4].

Опис SNMPv2.

У 1993 році в якості заміни SNMP був запропонований протокол SNMP2.

SNMP2 покращує характеристики продуктивності та безпеки SNMP, визначає додаткові типи даних, забезпечує операції типу «менеджер менеджерів» і визначає, як протокол відображається на транспортні рівні.

SNMP2 підтримує домени управління і більш ефективний збір даних від вузлів мережі. Однак виробники не виявляють особливого ентузіазму щодо його підтримки. Пояснюючи, кажуть, що замовники цього поки не просять. Однак без підтримки стандартів не можна управляти різнорідними конфігураціями.

Втім, може виявитися, що і SNMP2 не вирішить проблем управління. У ньому збережено механізм реєстрації, що вимагає, щоб виділена станція періодично опитувала вузли знаходяться.

Опис SNMPv3.

Як в більш ранніх версіях, в SNMP третьої версії агенти знаходяться на маршрутизаторах, а менеджери займаються їх опитуванням. У агентів і менеджерів тепер є ядро, яке виконує чотири основні функції: контроль доступу, обробка повідомлень, функції безпеки і диспетчеризація. Диспетчер обробляє всі вхідні і вихідні повідомлення і визначає чинну версію SNMP. Потім повідомлення відсилаються відповідного модулю обробки повідомлень, який відправляє

повідомлення системі контролю доступу або системі безпеки. У свою чергу, модуль обробки повідомлень відсилає повідомлення назад диспетчеру, який передає їх додатків SNMP [8].

SNMPv3 приваблює користувачів насамперед своєю безпекою. Розробники передбачили в ньому три рівні безпеки [4]:

– NoAuthNoPriv - паролі передаються у відкритому вигляді, конфіденційність даних відсутня;

– AuthNoPriv - аутентифікація без конфіденційності (більшість користувачів використовує саме цей рівень, так як рівень захищеності в ньому вже досить високий, а мережеві пристрою не перевантажуються шифруванням даних);

– AuthPriv - аутентифікація і шифрування. Максимальний рівень захищеності.

2.2. Аналіз програмних систем керування мережевим обладнанням

Будь-яка об’ємна обчислювальна мережа потребує додаткових методів для керування за винятком базових у стандартних мережевих операційних системах. Це обумовлено з досить значною кількістю обладнання комунікації, функціонування якого обов’язкове для виконання мережею своїх головних функцій.

Як правило великі мережі є досить розподіленими, що ускладнює її обслуговування, без наявності централізованої системи управління, що призначена для контролю мережевого трафіку і управління мережевим обладнанням в автоматичному режимі.

Автоматизована робота системи досягається за рахунок виконання простих дій, а прийняття складних рішень надається людині, на основі зібраної системою інформації.

Як правило кожен пристрій, поставляється з власним програмним забезпеченням для конфігурації та управління, проте з часом коли кількість обладнання зміниться і воно стане більш різноманітним, може виникнути проблема об'єднання розрізнених програм в єдину систему, і для вирішеннятакої проблеми, можливо, доведеться відмовитися від них і замінити їх на власну комплексну систему управління мережею [6].

2.2.1. rConfig

rConfig - веб-додаток, що виконує, по суті, одну важливу задачу: знімок конфігурації мережевих пристроїв і масове розгортання налаштувань. Адміністратор може самостійно налаштувати список команд, які повинні бути виконані на окремому пристрої або певної групи, при цьому команди можна співвідносити з категоріями. Для зручності подальшого відбору і завдання команд в інтерфейсі можна визначати будь-яку кількість категорій, додавати інформацію про виробників, логотипи і довільні поля. Для управління використовується Telnet або SSH. Налаштуйте пристрій вказується стандартна інформація для підключення (логін, пароль, IP-адреса і порт) і додаткова інформація про конкретний пристрої. Команди об'єднуються в завдання, які призначаються для виконання певних категорій в зазначений час, задається інтервал повторення (використовується cron). Список зберігаються на диск, при цьому вони розподіляються по каталогам, що відповідає категорії, влаштуванню і часу. Причому протягом дня можна завантажувати кілька копій конфігурації, все автоматично отримають свою мітку часу. При експериментах з мережевими настройками це виручає. Доступна функція порівняння збережених конфігурацій, статистика, перегляд журналів, пошук потрібного параметра в будь-якої категорії, звіти з налагодження і знайденим змін. Результати можуть відправлятися на email. Функція Configuration Compliance Management дозволяє контролювати конфігурації пристроїв для відповідності політиці.

Розповсюджується за ліцензією GNU GPL. Проект пропонує демоверсію, перед її використанням потрібно реєстрація. Правда, вона реалізована на старій версії 2, в якій відсутній ряд можливостей. Тому, щоб отримати повне уявлення, потрібно самостійно встановити rConfig (рис. 2.2).

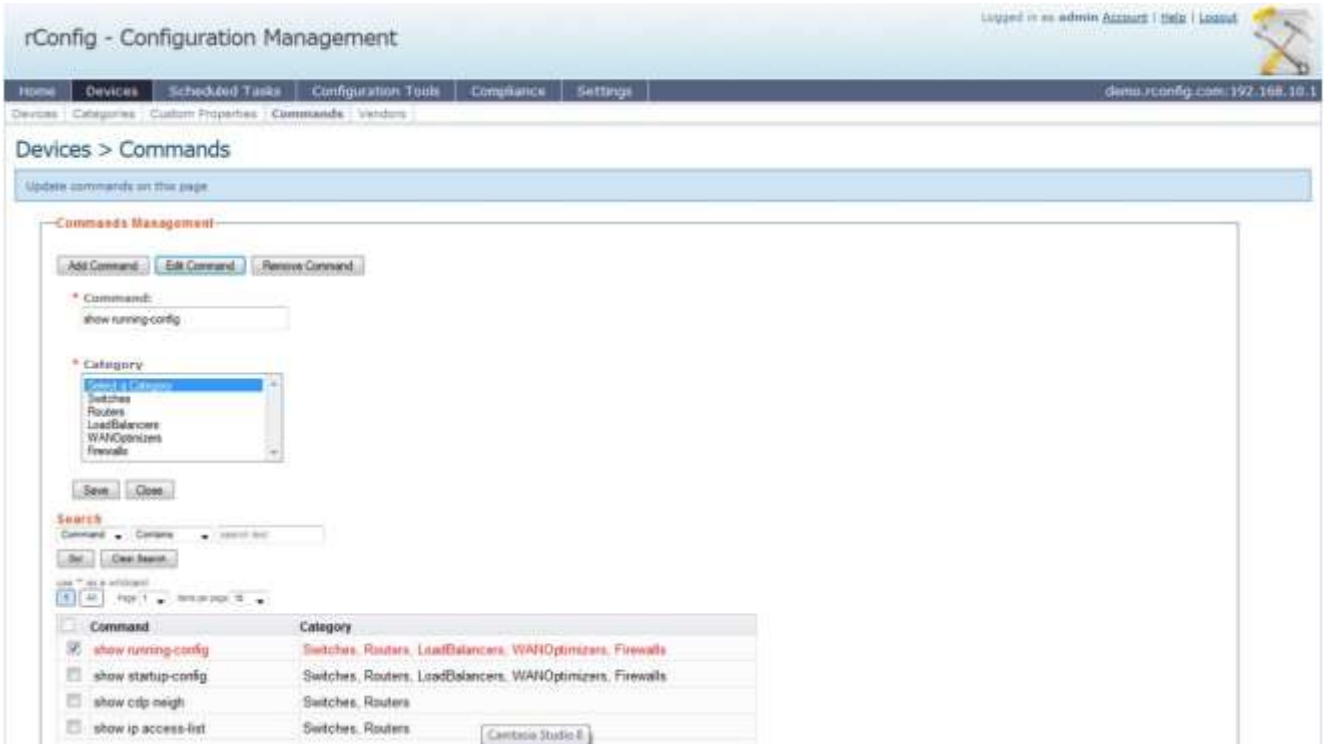

Рис. 2.2. Зовнішній вигляд rConfig

Написаний на PHP, для зберігання інформації задіюється MySQL, файли зберігаються на FTP-сервері. Вимоги до обладнання невеликі. По суті, для розгортання потрібно звичайний LAMP-сервер з окремими специфічними модулями PHP, про наявність яких доведеться потурбуватися. Документації проект особливо не пропонує, лише невелику інструкцію по встановленню в CentOS 6.x. В архіві з вихідними текстами є заготовки саме під цей дистрибутив. Ймовірно, його можна вважати рекомендованим розробниками. Хоча інструкція трохи застаріла навіть для 6.x, тобто просто скопіювати команди в вікно терміналу в розрахунку, що все візьме і почне працювати, чи не вийде. При розгортанні в CentOS 7 випливає ще кілька «дрібниць». Для інших дистрибутивів доведеться вже розбиратися з відмінностями за назвою пакетів і влаштуванню дерева файлової системи. Але після встановлення подальші налаштування цілком зрозумілі, хоча інтерфейс і не локалізований [7].

2.2.2. NeDi - Network Discovery

NeDi – сучасна система для управління, інвентаризації мережного устаткування і візуалізації мережі, яка поширюється під ліцензією GNU GPL. С її допомогою можна проводити в автоматичному режимі пошук і інвентаризацію мережевих пристроїв, досліджувати вузли, які до них підключені, отримувати інформацію (версія ПО, IP, MAC, конфігурація, вендор, дані про завантаження інтерфейсів, процесора, пам'яті, температурний режим) зі збереженням журналів і побудовою наочних графіків за допомогою rrdtool. У тому числі можемо виявляти і моніторити віртуальні машини в віртуальних середовищах. Реалізована підтримка API системи моніторингу Cacti, яка використовується для відтворення графіків. Також виконується резервування конфігурації маршрутизаторів, можливо відстежувати деякі інші мережеві події, наприклад витрата паперу на мережевому принтері. Крім власне виявлення, ведеться моніторинг, і при виході з ладу пристрою, відключення порту і деяких інших подіях (поява в мережі нового пристрою, створення бекапа) адміністратор отримує email або SMS. Доступні деякі звіти: пристрої, стан, події, інциденти та інші. Адміністратор може налаштувати фільтри відбору пристроїв за потрібною критерієм. Доступні кнопки, що дозволяють одним натисканням відібрати потрібну інформацію: наприклад, найбільш завантажені, найшвидші, найгарячіші, що не моніторять. Підтримується обладнання різних виробників, модульна архітектура у вигляді def-файлів дозволяє легко додати підтримку пристроїв будь-якого виробника (за допомогою утиліти Defgen).

NeDi також вельми корисний інструмент для моніторингу безпеки мережі. Він легко виявляє новедротове або бездротове пристрій, зміни в IP-адресації, поява в мережі пристрої, позначеного як зникле. Про події адміністратора попереджає спливаюче вікно і звуковий сигнал. Графіки показують проблемні перевантажені місця в мережі. Функція ідентифікації хоста дозволяє відстежувати відкриті порти Telnet або SSH.

Для виявлення та збору даних використовується SNMP, підтримуються протоколи CDP (Cisco Discovery Protocol), FDP (Foundry Discovery Protocol) і майкрософтовського LLTD (Link Layer Topology Discovery). Для отримання MAC-

адресації та іншої інформації з роутерів на базі Cisco IOS (і деяких інших) може бути використаний привілейований CLI. Такий спосіб швидше, ніж сканування VLAN, але його вже потрібно буде налаштовувати персонально, вказуючи паролі для доступу до Telnet або SSH. Основним методом залишається все-таки SNMP. Перевіряються також DNS- і NTP-сервіси. Конфігурації зберігаються в БД (MySQL або PostgreSQL) або у вигляді текстових файлів. Після виявлення адміністратор може самостійно додати опис і призначити більш наочний значок для конкретного пристрою [7].

Можлива одночасна робота декількох адміністраторів, організований простий чат для швидкого обміну інформацією. Крім власної бази користувачів, залучається LDAP і RADIUS. Є кілька корисних інструментів на кшталт IP-калькулятора і конвертера тексту. Реалізований простий бекап і відновлення налаштувань самого NeDi.

Фактично NeDi є кілька Perl-скриптів для конкретного завдання: виявлення, моніторингу, запису подій в БД, SNMP і генерування статистики. Для їх запуску використовується консоль, GUI і cron.

Вся інформація виводиться через веб-інтерфейс, наявні шаблони дозволяють його однаково комфортно переглядати як на ПК з великим екраном, так і на мобільному пристрої. Вид можна міняти за допомогою тем. Інтерфейс, правда, доступний тільки англійською, італійською та німецькою. Локалізація реалізована у вигляді HTML-файлів, і при бажанні можна перевести самостійно в міру впровадження. Після реєстрації в першому вікні Welcome to NeDi (знаходиться в User -> Profile) видаються загальні підказки щодо подальших дій: уточнення параметрів, встановлення відправної точки для сканування і сканування мережі, налаштування завдань і моніторингу SNMP. Просто натискаєш на значок і переходиш до пункту, а можна сам значок пошукати в меню. Бекап пристрою може бути проведений в CLI безпосереднім запуском nedi.pl або в GUI Devices -> List -> Status. Інформація зберігається зазвичай у / var / nedi / config. Відновлення налаштувань лежить виключно на адміністратора.

У більшості випадків установки через інтерфейс виробляються за допомогою редагування конфігураційних файлів, для зручності всі вони зібрані в список, що

розкривається (рис. 2.3.). В основному файлі nedi.conf слід прописати правильні параметри для сканування SNMP (обов'язково), опціонально політики доступу до CLI, маску IP, налаштування підключення до БД, відправку email і SMS, що виводяться повідомлення, інтеграцію з Cacti і інше. Відступи слід робити табом.

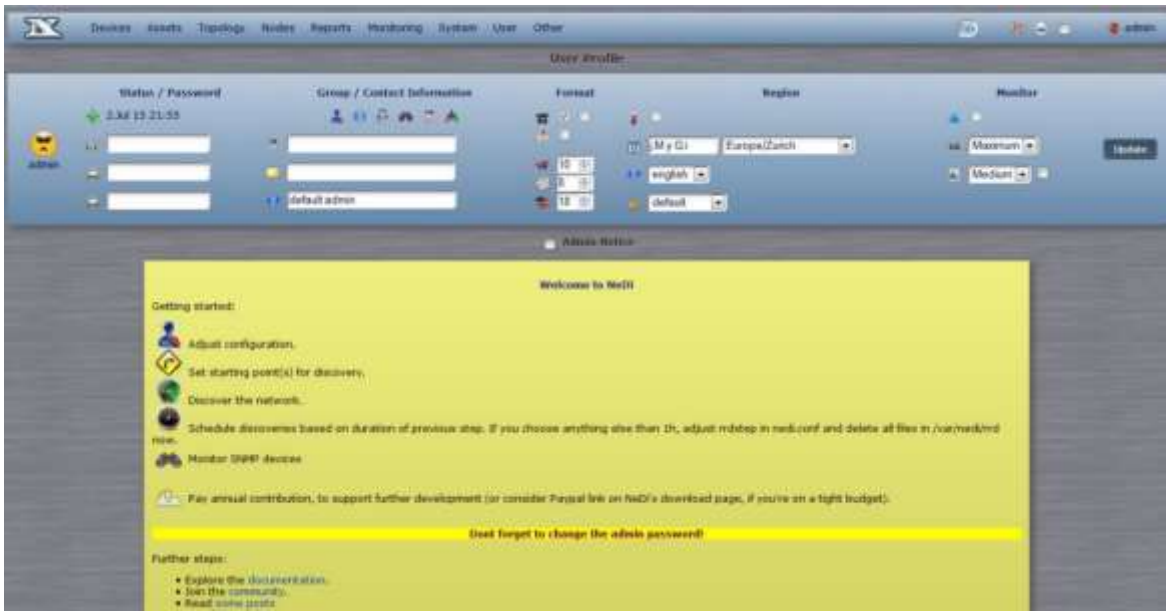

Рис. 2.3. Зовнішній вигляд NeDi

Інтерфейс написаний на PHP, тому для роботи буде потрібно звичайний LAMP-сервер. Проект пропонує інструкції по установці на FreeBSD, OS X і SUSE. Ймовірно, можна їх вважати рекомендованими, хоча NeDi прекрасно працює і в будь-якому іншому Linux. Керівництва для інших ОС можна пошукати в Мережі і адаптувати під поточний реліз. Також відповіді на деякі питання є на форумі. Крім архіву з вихідними текстами, пропонується готовий ISO-образ, зібраний на базі OpenBSD, і образ для VMware (пароль в CLI - root / root, веб - admin / admin). Образ для VMware повністю налаштований, при наявності DHCP-сервера отримує IP автоматично, після чого можна заходити через веб і починати роботу. Знання особливостей OpenBSD не потрібно. Підготовлено кілька команд, що дозволяють швидко виконати основні операції, всі вони показуються при завантаженні (ne запускає NeDi, wa - snmpwalk, er виводить помилки веб-сервера і так далі). Проект пропонує документацію плюс відео на своєму каналі. В принципі, матеріалу достатньо, щоб при бажанні розібратися [7].

2.2.3. Ansible

Ansible – розробник продукту Майкл ДеХаан. Продукт був розроблений для автоматизації одноманітних завдань адміністрування серверів у великих середовищах. Майкл був у новоствореній технологічної групі RedHat, де заснував різні проекти (такі як Cobbler), а після відходу з RedHat заснував Ansible (хоча тепер Ansible і належить RedHat).

Дизайн.

Ansible простий, що є його головною якістю (це відразу стане зрозумілим, якщо подивитися на інші дві якості). У ньому немає ні демонів, ні баз даних, а вимоги для установки мінімальні. Ansible просто встановлюється на Linux-машині і все. Можна визначити цільові сервери в статичному файлі, згрупувати їх в змістовні розділи або використовувати якийсь динамічний модуль виявлення хостів (такий як Amazon EC2 або OpenStack) для пошуку віртуальних машин на основі виклику API. Після того як була проведена інвентаризація, можна виділити специфічні для хоста або групи змінні для використання в плейбук. Вони, знову ж таки, зберігаються в статичних текстових файлах.

Потім Ansible підключиться до вибраного хосту або групі і запустить плейбук. Збірник сценаріїв (плейбук) являє собою послідовність модулів Ansible для виконання на віддалених хостах, написаних на YAML.

Підключення до віддаленого хосту трохи схоже на добре сплановані військові навчання: увійшов, зробив свою роботу і вийшов.

Ansible працює за наступним алгоритмом: підключення до сервера по SSH (або WS-Man / WinRM для Windows), копіювання коду на мові Python, виконання і видалення самого себе.

Архітектура.

Архітектура Ansible прямолінійна: існує додаток, що виконується на локальному комп'ютері, і виконувані на віддалений хост завдання, які підтримують між собою зв'язок по SSH і через файли, що передаються по SCP / SFTP. У Ansible немає архітектури «сервер-клієнт», на відміну від двох інших продуктів, тому

розпаралелювання завдань відбувається на локальній машині, але не масштабується на кілька серверів (якщо не використовувати Tower) [8].

Ansible при управлінні віддаленими машинами не залишає на них свого коду, тому питання, як оновити Ansible при переході на нову версію, в дійсності не постає.

Корпоративна підтримка.

Ansible Tower - це корпоративна версія (Enterprise), яка перетворює командний рядок Ansible в сервіс з веб-інтерфейсом, планувальником і системою повідомлень.

Він також має користувальницький інтерфейс для плейбуків, за допомогою якого можна автоматизувати розгортання хмарної інфраструктури, а потім автоматично додати віртуальні машини до переліку.

Підтримка мереж.

Розділ про підтримку мереж в Ansible охоплює всіх основних вендорів мережевого обладнання і платформи. У Ansible можливо:

– автоматизувати настройку всього, що пов'язано з мережевим обладнанням, - від вказівки систем для підключення до конфігурації основних сервісів, і все це за допомогою модулів і плейбук для конкретної мережевої платформи;

– тестувати і перевіряти поточний стан мережі, виконувати або відкладати процес збору фактів про існуючої конфігурації;

– безперервно перевіряти відповідність змін мережевим налаштувань.

Ansible підтримує Arista, Cisco (всі програмовані платформи), F5, Juniper та інших виробників. Тільки в Ansible підтримка в основному надається і здійснюється вендорами, а не співтовариством. На даний момент це найкращі інтерфейси API, найбільша функціональність і робота з найостаннішими платформами (оскільки підтримуються більш нові версії).

Сильні сторони:

– дуже швидко і просто почати роботу;

– безліч прикладів, документації і модулів від спільноти;

– Ansible Tower реалізує функції для великих корпоративних впроваджень;

– мережеві модулі, підтримувані вендорами.

Слабкі сторони:

– при відсутності належного контролю оператори можуть зберігати плейбук і ключі SSH на власні ноутбуки. Не зовсім недолік в Ansible, але потрібен постійний контроль;

– ніякої історії подій автоматизації, контроль над цільовим хостом зберігається тільки на час виконання плейбук, не більше. Не можна використовувати для довгострокових завдань [8].

2.3. Аналіз програмних систем обліку мережевого обладнання

Облік комп'ютерів і пристроїв у мережі є невід'ємною частиною обслуговування парку техніки в організації будь-яких розмірів. Облік комп'ютерів та обладнання дозволяє підтримувати техніку в актуальному стані і уникнути зайвих витрат на її обслуговування.

Збір інформації про кожному пристрої вручну вже давно не актуальний зважаючи на свою трудомісткість. Вирішити поставлене завдання і отримати максимально точну інформацію в найкоротші терміни дозволяють спеціальні програми для інвентаризації мережі [9]. Один раз налаштувавши таку програму, ви зможете в будь-який час отримати детальну і актуальне зведення по кожному пристрою, а також дізнаєтеся про будь-які зміни в мережі, які потребують вашої уваги.

2.3.1. Total Network Inventory 4

Стандартна версія Total Network Inventory 4 надає повний функціонал для інвентаризації різних мережевих пристроїв. Програма легка в налаштуванні і інтуїтивно зрозуміла з першого запуску. Вам доступно сканування систем Windows, OS X, Linux, FreeBSD і ESX / ESXi, а також сканування мережевих пристроїв по протоколах SNMP і Telnet.

Вся отримана при скануванні інформація може бути виведена в звіти, які легко експортуються в будь-який зручний для вас формат.

Total Network Inventory також дозволяє налаштувати сканування і створення звітів за розкладом, що дозволяє автоматизувати рутинні завдання (рис. 2.4.).

Окремо хочеться відзначити Журнал змін - режим, що дозволяє отримати максимум інформації про будь-які зміни, які відбулися на сканованих пристроях з моменту останнього сканування.

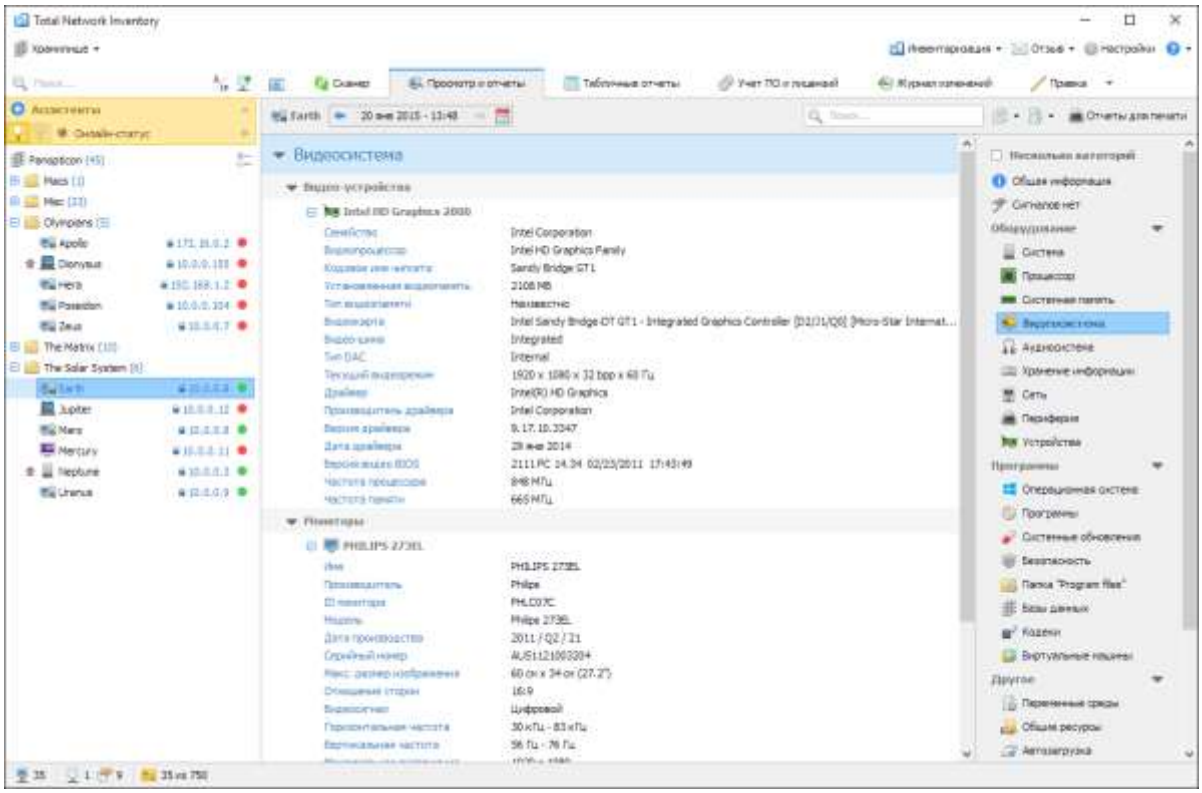

Рис. 2.4 Вікно програми Total Network Inventory

Вартість ліцензії починається від 1000 грн за ліцензію на 25 пристроїв.

Плюси:

– повний функціонал для інвентаризації мережі;
– легко налаштувати;
– дружній інтерфейс;
– низька вартість.

Мінуси:

– може бути встановлена тільки на Windows-системи [9].

2.3.2. Spiceworks

Spiceworks є одним з найбільш популярних безкоштовних рішень для інвентаризації з веб-інтерфейсом. Програма надає зручний і зрозумілий сканер мережі, легку фільтрацію отриманої інформації і механізм порівняння пристроїв.

Наявність веб-інтерфейсу вносить певні обмеження на візуалізацію деяких процесів і настройку програми під свої потреби, але це компенсується легким, а головне розрахованих на багато користувачів, доступом до даних (рис. 2.5.).

Окремо хочемо відзначити, що крім інвентаризації програма також надає інструменти для моніторингу, створення список закупівель та підтримки користувачів (helpdesk).

Рис. 2.5. Вікно програми Spiceworks

Вартість ліцензії: безкоштовно (монетизація за рахунок реклами).

Плюси:

– дружній інтерфейс;

– безкоштовне розповсюдження;

– можливість підключення модулів моніторингу, списку закупівель та підтримки користувачів.

Мінуси:

– доступ тільки через веб-інтерфейс;

– наявність реклами;

– обмежена персоналізація [9].

2.3.3. Network Inventory Advisor

Незважаючи на трохи застарілий інтерфейс, Network Inventory Advisor пропонує повний спектр необхідних інструментів для обліку пристроїв в мережі. Програма дозволяє виконувати сканування систем Windows, Unix-подібних систем і SNMP-пристроїв (рис.2.6). Грунтуючись на результатах сканування, ви можете будувати звіти різної складності як по апаратній, так і програмної частини обладнання.

Окремо хочу відзначити можливість виявлення і збору ключів продуктів Autodesk.

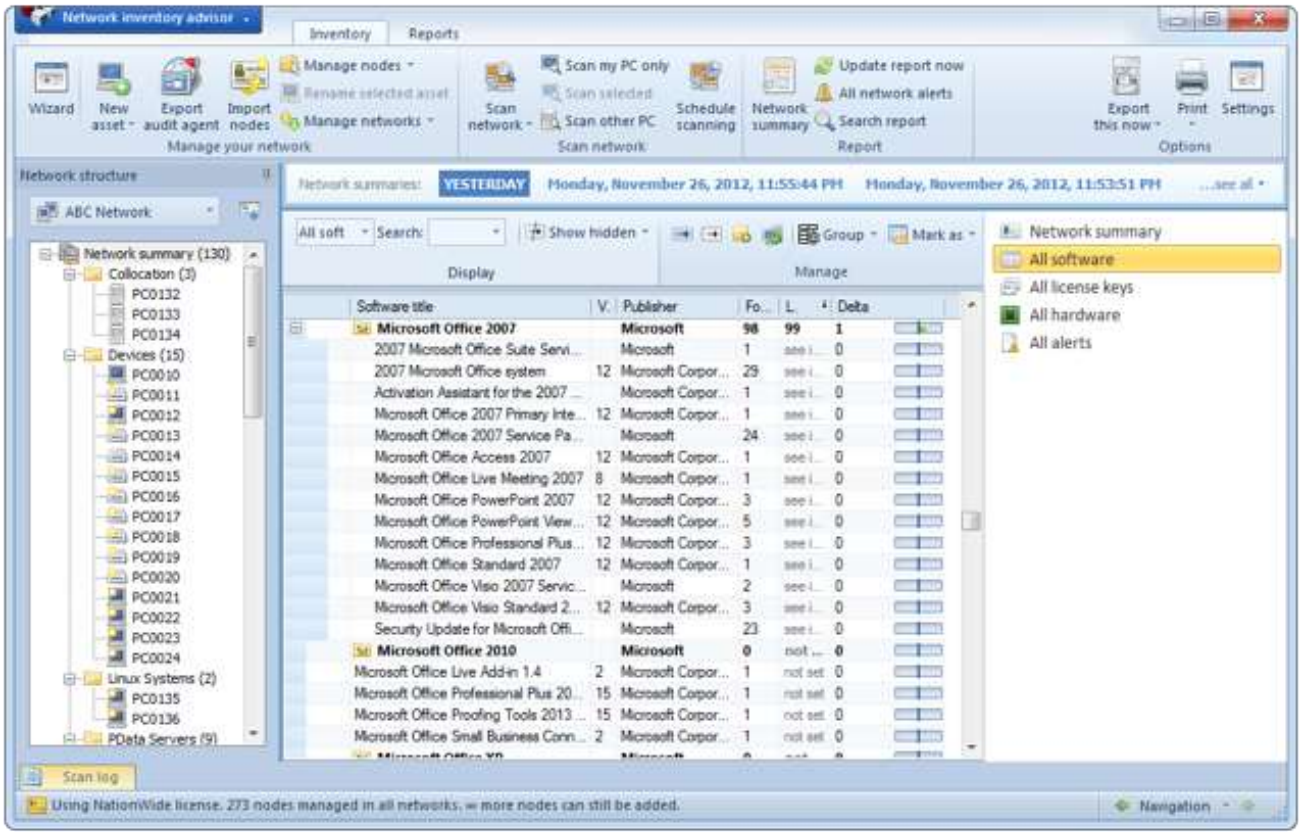

Рис. 2.6. Вікно програми Network Inventory Advisor

Оновлення в рамках старшої версії доступні безкоштовно. Вартість ліцензії починається від 89 доларів за ліцензію на 25 пристроїв.

Плюси:

– повний базовий функціонал для інвентаризації мережі;

– може бути встановлений як на Windows, так і MacOS;

– виявлення ключів продуктів Autodesk.

Мінуси:

– застарілий інтерфейс;

– рідко оновлюється;

– немає Web-інтерфейсу [9].

2.3.4. Solarwinds Network Inventory

Продукт Solarwinds дещо вибивається зі списку розглянутих додатків як за ціною, так і за функціоналом. Проте, це ПЗ досить популярне і використовується в великих корпораціях для вирішення широкого спектра завдань.

Розглянутий продукт включає в себе ряд інструментів для інвентаризації та моніторингу пристроїв в мережі, контролю ліцензій ПЗ, моніторингу активних процесів і сервісів в реальному часі, а також для автоматичного виконання сценаріїв у разі відхилення певних значень від норми (рис. 2.7.).

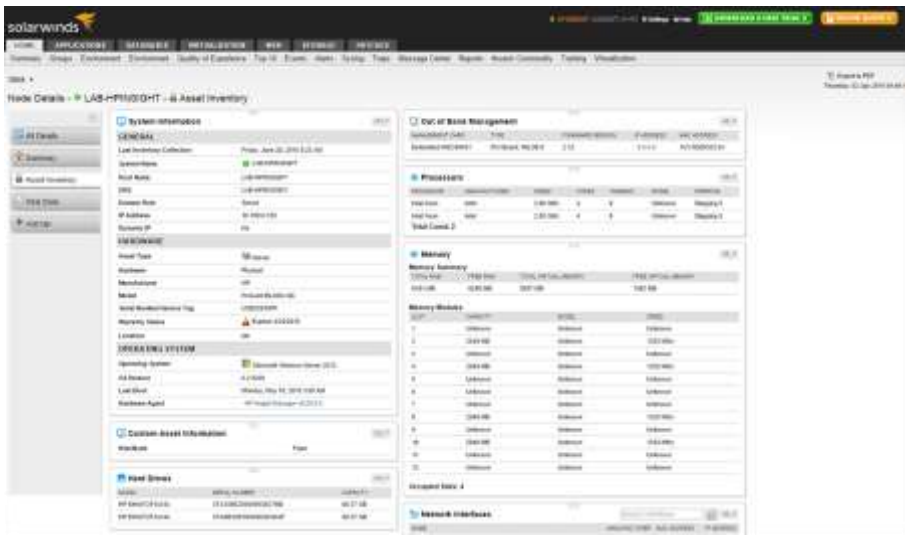

Рис. 2.7. Вікно програми Solarwinds Network Inventory

Solarwinds також надає окремий модуль для створення карти мережі, можливості якого дійсно вражають. Вам доступна масштабована карта світу, на якій ви можете створити локальні карти мережі кожного офісу своєї компанії.

Вартість ліцензії починається від 5995 доларів за ліцензію на 25 пристроїв.

Плюси:

– широкі можливості по інвентаризації та моніторингу;

– велика кількість навчальних відео;

– розрахований на багато користувачів доступ.

Мінуси:

– доступ тільки через веб-інтерфейс;

– висока вартість ліцензії [9].

2.3.5. Open-AudIT

Open-AudIT є програмою для інвентаризації з відкритим вихідним кодом. Цей додаток досить популярний завдяки наявності повністю безкоштовної версії, що пропонує базовий функціонал для інвентаризації.

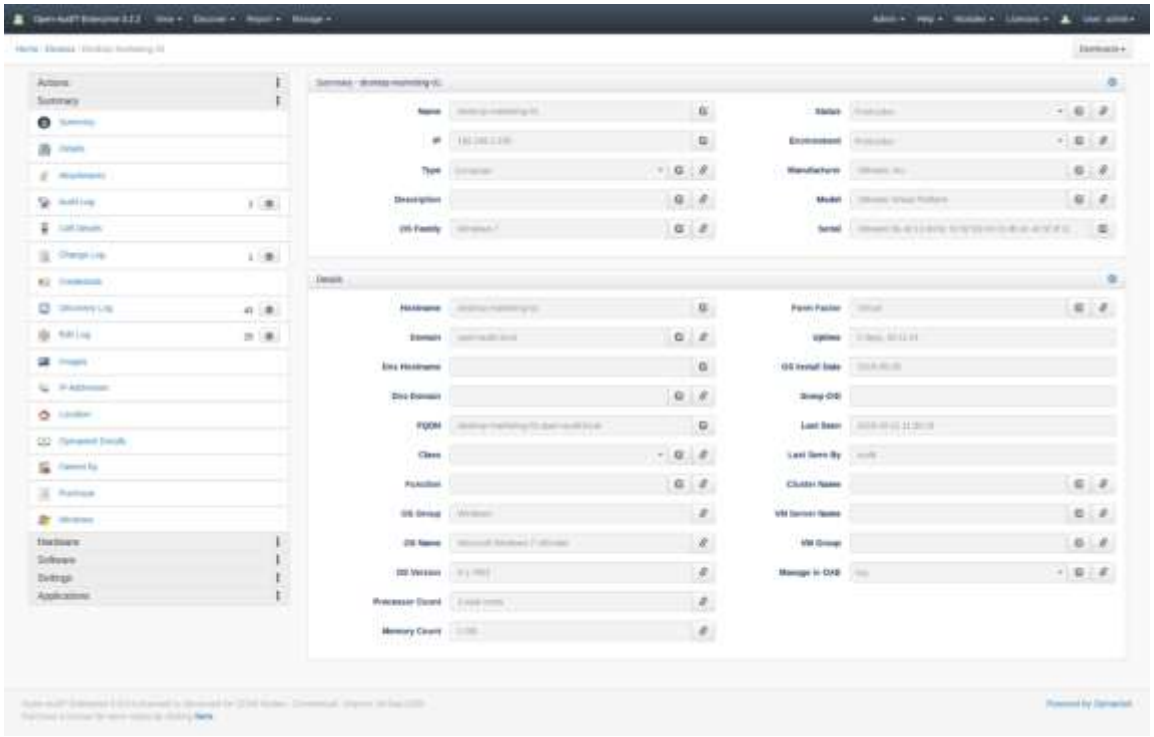

Рис. 2.8. Вікно програми Open-AudIT

В цілому програма має дуже приємний і зручний в роботі інтерфейс, а карта мережі підтримує інтеграцію з Google maps, що є не у всіх конкурентних продуктах.

Збір інформації про Windows-пристроях програма виконує через опитування WMI, що в деяких випадках може давати неповну інформацію в порівнянні з низькорівневим скануванням пристроїв. Проте, таке ПЗ не вимагає тривалого налаштування і дозволяє швидко опитати парк техніки виконуючи мінімальні попередні налаштування.

Розширений функціонал, включаючи карту мережі, експорт пристроїв і персоналізовані звіти доступні тільки в платній версії програми, яка також включає в себе повноцінну комерційну підтримку [4].

Вартість ліцензії: безкоштовно для Community edition. 1199 доларів ліцензію Enterprise на 100 пристроїв (мінімальна ліцензія).

Плюси:

– повний базовий функціонал для інвентаризації мережі;

– наявність безкоштовної версії.

Мінуси:

– доступ тільки через веб-інтерфейс;

– доступний тільки безагентний збір інформації [9].

Висновки

В другому розділі проаналізовані можливі варіанти управління мережевим обладнанням за допомогою протоколів telnet, SSH, SNMP, розглянуті принципи роботи кождного з протоколів. Також проаналізовано переваги та недоліки існуючих окремих систем для керування та окремих систем для обліку обладнання.

РОЗДІЛ 3
ВСТАНОВЛЕННЯ ВИМОГ ТА ПРОЕКТУВАННЯ СИСТЕМИ

3.1. Функціональні та нефункціональні вимоги

Система керування та обліку мережевого обладнання призначена для зберігання інформації про нього і віддаленого налаштування. В процесі використання даного продукту можна отримати повну інформацію про обраний пристрій (ip адресу, mac адресу, модель, назву, фізичне місце розташування), провести автоматизоване налаштування базової конфігурації, внести зміни в існуючу конфігурацію (вимкнути або увімкнути порт, додати vlan глобально або на порт), отримати інформацію про статан портів, кількість помилок на них, таблицю mac адрес.

Цілі створення системи:

– автоматизація процесу налашутвання та керування мережевим обладнанням, зменшення витрат часу на виконання цих завдань;

– упорядкування та структурузація інформації про мережеве обладнання, для зменшення витрат часу на її пошук.

Функціональні можливості розроблюваного програмного забезпечення графічно можна представити, використовуючи use case діаграму (рис. 3.1.), що відображає можливі варіанти використання продукту.

Рис. 3.1. Use case діаграма

Користувачем даної системи можуть бути експерти чи звичайні користувачі, що хочуть отримати інформацію про мереже обладнання або інформацію на мережевому обладнанні або провести його налаштування.

Вимоги до інтерфейсу коритсувача: інтерфейс має бути зручним, простим і зрозумілим, щоб користувач міг з легкістю виконати поставлене завдання. Система має містити довідкову інформацію, вказувати на допущені користувачем помилки та пропонувати варіанти їх вирішення. Має бути достатня комунікація між користувачем та системою.

Вхідними даними будуть дані, введені користувачем у систему, тип яких відповідає розширенню систему та відповідні запити на виконання певних функцій системою (внесення нових данних, редагування та видалення існуючих данних, виконання функцій налаштування та керування мережевим обладнанням).

Вихідними данними будуть відповідно знайдена системою інформація, що запитував користувач або результат виконання операцій налаштування чи керування мережевим обладнанням, представлені в текстовому чи графічному вигляді.

Функціональні вимоги системи:

– система повинна надавати можливість аутентифікації – розпізнавання користувача за логіном та паролем;

– система повинна надавати можливість авторизації – надання прав доступу для окремих користувачів або їх груп;

– система повинна надавати можливість аккаунтингу – ведення звітності дій користувача;

– система повинна надавати користувачу можливість вводити дані у визначеному форматі, та забороняти введення даних іншого формату;

– система має надавати можливість зберігання інформації у вигляді структурованої бази даних;

– система має надавати можливість видаляти та редагувати данні відповідно до прав доступу кожного користувача;

– система має працювати з різними моделями та виробниками мережевого обладнання;

– система має надавати можливість гнучкого пошуку інформації за допомогою різноманітних фільтрів;

– система повинна представляти результати виконання операцій у текстовому або графічному вигляді.

Роботу програми можна представити за допомогою діаграми станів, що показана на рис. 3.2. Дана діаграма відображає поведінку системи та реакцію її на деякі події.

Рис. 3.2. Діаграма станів

Нефункціональні вимоги:

Система повинна відповідати наступним нефункціональним вимогам:

– розроблювана система має бути кросс-браузерним веб-застосунком;

– система має підтримувати багатокористувацький доступ, не менше 30 користувачів одночасно;

– система повинна відповідати на запит менше хвилини;

– система повинна максимально запобігати виникненню помилок;

– система має швидко відновлюватись після збоїв.

3.2. Вибір архітектури

Архітектура інформаційної системи – концепція, яка визначає модель, структуру, виконувані функції і взаємозв'язок компонентів інформаційної системи.

За способом програмно-апаратної реалізації можна виділити декілька видів архітектур інформаційних систем.

За виконуваними функціями компоненти інформаційної системи можна розділити на три рівня: рівень представлення, рівень бізнес-логіки та рівень доступу до даних (рис.3.3.).

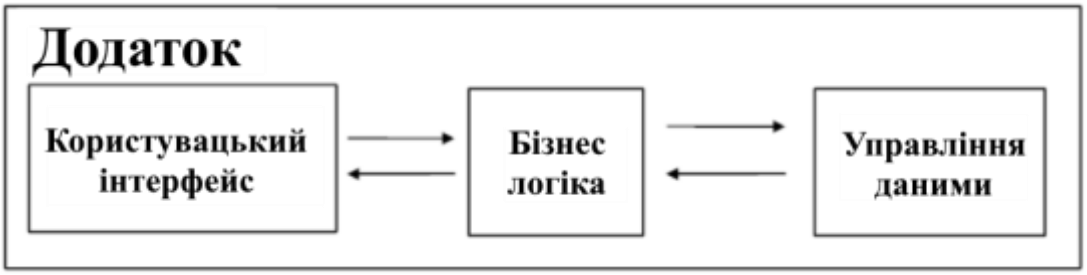

Рис.3.3. Компоненти інформаційної системи

Рівень представлення – все, що пов'язано з взаємодією з користувачем: натисканняклавіш, рух миші, відображення зображення, вивід результатів пошуку і т.д.

Бізнес логіка – правила, алгоритми поведінки програми на дії користувача або на внутрішні події, правила обробки даних.

Рівень доступу до даних – зберігання, вибірка, модифікація і видалення даних, пов'язаних з вирішенням додатком поставленої задачі [10].

3.2.1. Файл-серверна архітектура

З появою локальних мереж файли почали передаватися по мережі. Спочатку були однорангові мережі, де всі комп'ютери мали рівні права.

Потім виникла ідея збереження всіх загальнодуступних файлів на окремому комп’ютері в мережі – файл-сервері.

Файл-серверні додатки – це додатки, схожі за своєю структурою з локальними додатками і використовують мережеві ресурси для зберігання програми і даних.

До функцій сервера входить: зберігання даних і коду програми. Клієнт виконує обробку даних виключно за рахунок власних ресурсів.

Модель файлового сервевера представлена на рисунку 3.4.

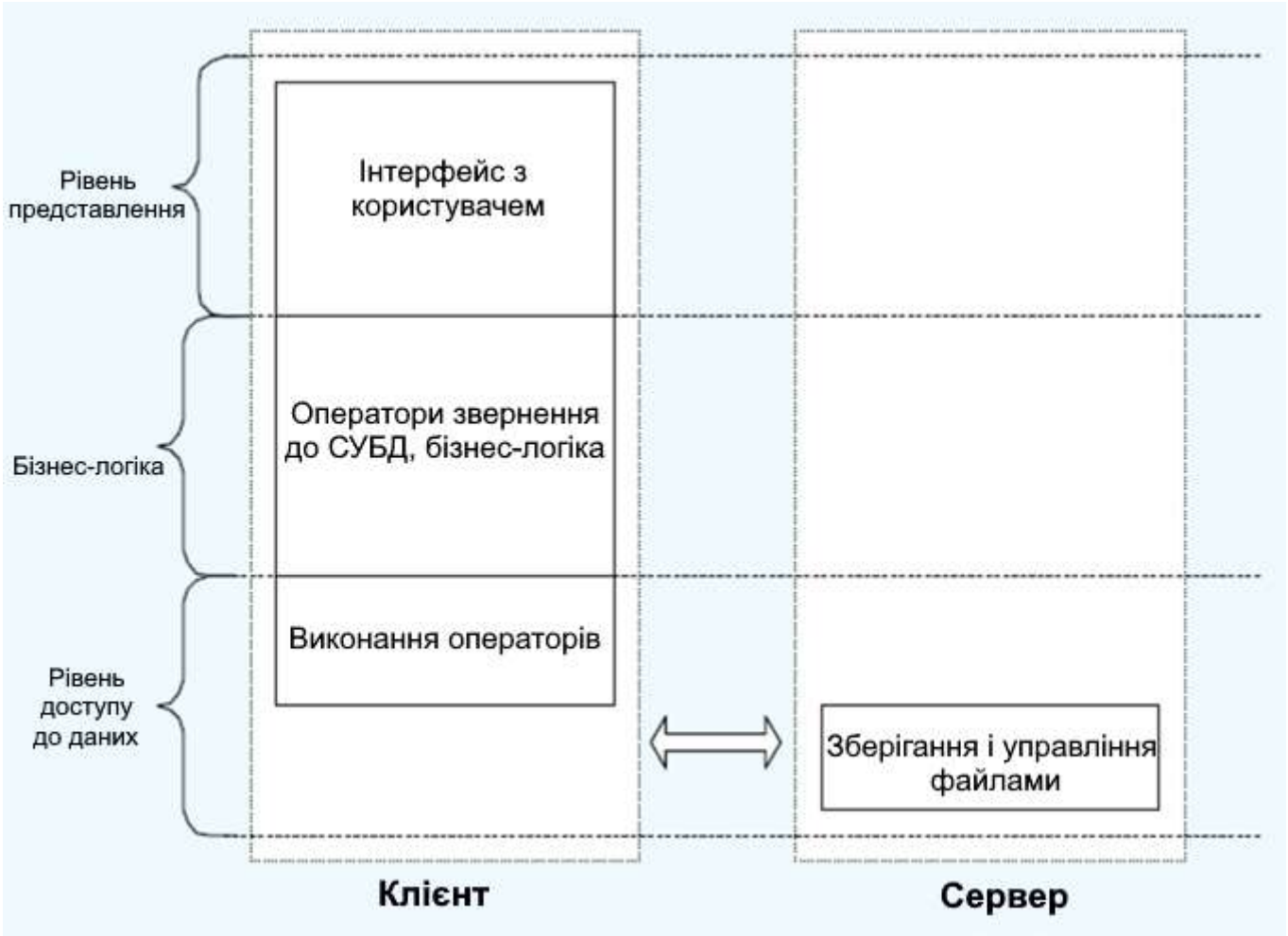

Рис. 3.4. Модель файлового сервера

Переваги:

– розрахований на багатокористувацький режим роботи з даними;

– зручне централізоване управління доступом;

– низька вартість розробки;

Недоліки:

– низька продуктивність;

– низька надійність;

– слабкі можливості розширення;

Недоліки архітектури з файловим сервером обумовлені тим, що дані зберігаються в одному місці, а обробляються в іншому. З цього випливає, що їх потрібно передавати по мережі, щоспричиняє високе навантаження на мережу і, внаслідок цього, різке зниження продуктивності додатка при збільшенні кількості одночасно працюючих клієнтів. Іншим важливим недоліком такої архітектури є децентралізоване рішення проблем цілісності і узгодженості даних і одночасного доступу до них. Таке рішення знижує надійність програми [10].

3.2.2 Клієнт-серверна архітектура

Основною відмінністю клієнт-серверної архітектури від архітектури файл-сервера є відмова від внутрішнього представлення даних (фізичної схеми даних).

Тепер клієнтські програми працюють з даними на рівні логічної схеми (рис. 3.5.).

Отже, використання архітектури клієнт-сервер дозволило створювати надійні (в плані цілісності даних) розраховані навелику кількість користувачів інформаційні системи з централізованою базою даних, незалежні від апаратної (а часто і програмної) частини сервера баз даних і підтримкою графічного інтерфейса користувача на клієнтських станціях, з'єднаних локальною мережею. Також варто зазначити, що витрати на розробку додатків суттєво зменшувались.

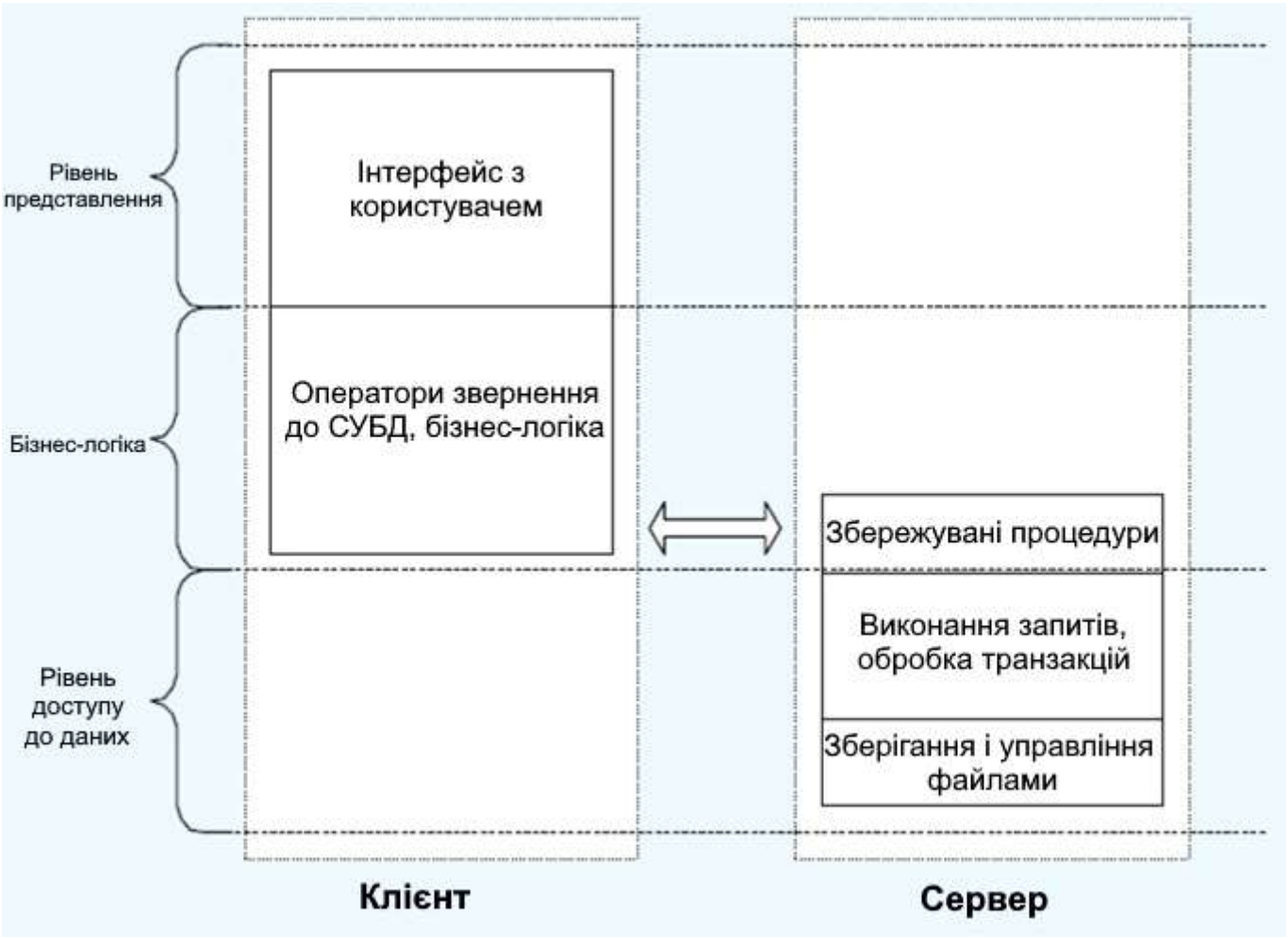

Рис. 3.5. Модель сервера системи управління базами даних

Основні особливості:

– клієнтська програма працює з даними за допомогою запитів до серверного програмного забезпечення;

– базові функції програми розділені між клієнтом і сервером.

Переваги:

– повна підтримка багатокористувацької роботи;

– гарантія цілісності даних.

Недоліки:

– бізнес логіка додатків залишилася наклієнтській стороні. При будь-якій зміні алгоритмів, треба оновлювати призначене для користувачапрограмне забезпечення на кожному клієнті;

– високі вимоги до пропускної здатності комунікаційних каналів з

сервером, що унеможливлює використання клієнтських станцій поза межами локальної мережі;

– слабкий захист даних від злому, особливо від недобросовісних користувачів системи;

– висока складність адміністрування і налашутвання робочих місць користувачів системи;

– необхідність використовувати потужні комп’ютери на клієнтських місцях;

– висока складність розробки системи через необхідність виконувати бізнес-логіку і забезпечувати користувальницький інтерфейс в одній програмі.

Більшість недоліків класичної або дворівневої клієнт-серверної пов’язані з використанням клієнтської станції для виконаня бізнес-логіки інформаціної системи. Тому очевидним етапом розвитку архітектур інформаціних систем стала ідея "тонкого клієнта", тобто розподіл алгоритмів обробки даних на частини пов'язані з виконанням бізнес-функцій і пов'язані з відображенням інформації в зручному для людини вигляді. Втакому разі на клієнтській машині залишається лише друга частина, пов'язана з первинною перевіркою і відображенням інформації, а вся реальна функціональність системи переноситься на серверну частину [10].

3.2.3. Перехідна до трирівневої архітектури (2,5-рівнева)

Використання збережених процедур і обчислення даних на стороні сервера зменшують трафік та збільшують безпеку. Бізнес-логіка все одно частково реалізується клієнтом.

Така організація системи дуже схожа на організацію перших об’єднаних систем з єдиною відмінністю, що на місці користувача стоїть не термінал, а персональний комп'ютер, що надає графічний інтерфейс користувача, наприклад, зараз в якості клієнтських програм часто застосовують стандартні веб-браузери. Використання систем управління базами даних з усіма їх перевагами стало обов'язковим. Програми для серверної частини розробляють, як правило, за допомогою спеціалізованих мов, користуючись механізмом збережених процедур сервера баз даних.

Таким чином, інформаційна система з архітектурою клієнт-сервера та тонкого клієнта , з точки зору логічної організації, поділяється на три рівні:

– рівень даних;

– рівень бізнес-функцій (процедури, що зберігаються);

– рівень представлення.

На жаль, зазвичай, при такій схемі побудови інформаційної системи вся бізнес-логіка додатка пишеться на вбудованих мовах систем управління базами даних, які не призначені для цього. Тому, дуже часто частина бізнесфункцій переноситься на клієнтську сторону, через що її об’єм неодмінно збільшується. Зокрема через це або через те, що фізично такі інформаційні системи складаються з двох компонентів, таку архітектуру часто називають 2.5-рівневий клієнт-сервер.

На відміну від дворівневої архітектури 2.5-рівнева архітектура як правило не вимагає наявності високошвидкісних каналів зв'язку між клієнтською і серверною частинами системи, оскільки мережею передаються вже готові результати обчислень – майже всі обчислення виконуються на стороні сервера. Також значно збільшується і захист інформації – користувачам надаються права на доступ до функцій системи, а не на доступ до її даних. Проте 2.5 архітектура має як переваги об’єднаного підходу так і всі його недоліки, такі як: обмежена масштабованість, залежність від програмної платформи, обмежене використання мережевих обчислювальних ресурсів. Крім того програми для серверної частини розробляються на вбудованих мовах опису процедур систем управління базами даних, що призначені для валідації даних і побудови нескладних звітів, але ніяк не для створення інформаційної системи масштабу підприємства. Все це знижує швидкодію системи, підвищує трудомісткість створення з модифікацій інформаційної системи і вкрай негативно позначається на вартості апаратних засобів, необхідних для її функціонування [10].

3.2.4. Трирівнева клієнт-серверна архітектура

Для вирішення наведених вище проблем була запропонована так звана трирівнева клієнт-серверна архітектура. Основна відмінність якої від архітектури 2.5 – фізичний розділ програм на ті, що відповідають за зберігання даних та програми, де

ці дані обробляються (сервер додатку). Таке розділення програмних компонентів дозволяє оптимізувати навантаження на мережу та на обчислювальне обладнання комплексу (рис. 3.6.).

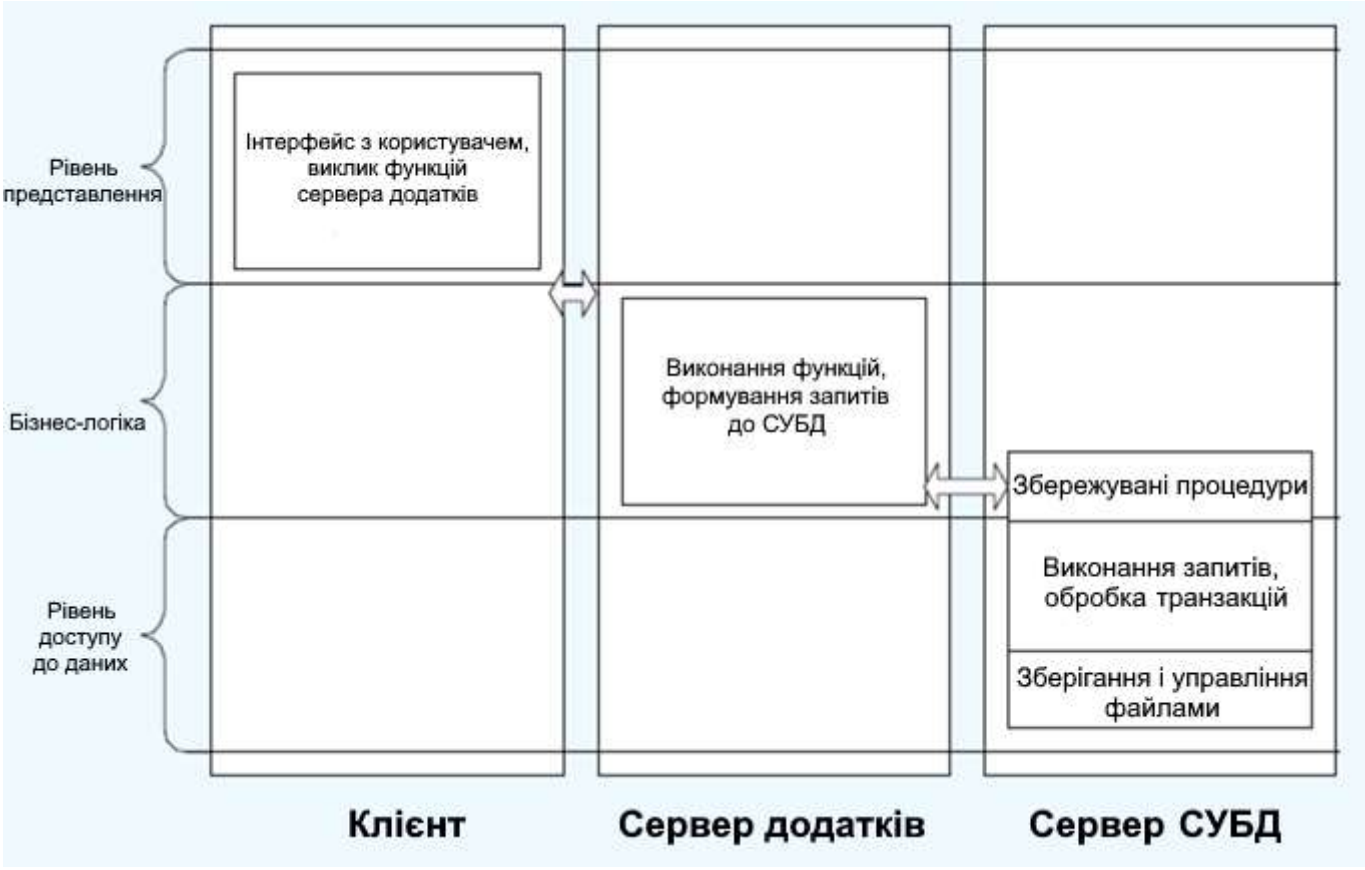

Рис. 3.6. Модель сервера додатків

З точки зору програмного забезпечення компоненти трирівневої архітектури, можуть бути представлені певними серверами баз даних, веб-серверами та браузерами будь-яких виробників.

Переваги:

– тонкий клієнт;

– між клієнтською програмою і сервером додатка відбувається обмін лише мінімально необхідою кількістю даних – аргументи функцій, що викликаються і значення результатів у відповідь. Це теоретична межа ефективності використання

ліній зв'язку, навіть робота з ANSI-терміналами (не кажучи вже про використання протоколу http) вимагає більшого навантаження на мережу;

– сервер додатка інформаційної ситсеми може запускатися в одному або декількох екземплярах на одному або декількох комп'ютерах, що дозволяє адміністратору інформціної системи визначати ефективність і безпеку викростання обчислювальних потужностей організації на власний розсуд;

– дешевий трафік між сервером додатків і системою управління базами даних. Такий трафік може бути і дещо більшим, проте це завжди трафік локальної мережі, а її пропускна здатність, як правило, досить велика і дешева. В разі нагальної потреби, завжди можна запустити сервер додатку і систему управління базами даних на одній машині, що зменшить кількість мережжевого трафіку до нуля;

– підвищення швидкості роботи системи в цілому, за рахунок зниження навантаження на сервер даних на відміну від 2.5-рівневої схеми;

– дешевше збільшувати функціональність і оновлювати програмне забезпечення.

Недоліки:

– більші витрати на адміністрування і обслуговування серверної частини.

Масштабованість систем з використанням трирівневої архітектури архітектури вражає. Одна і та ж система може працювати як на одному окремому комп'ютері, виконуючи на ньому програми системи управління базами даних, сервера додатку і клієнтської частини, так і в мережі, що складається з сотень і тисяч машин. Єдиною причиною, як було вище зазначено, що перешкоджає нескінченній масштабованості, є лише вимога ведення єдиної бази даних.

Окрім вимог, щодо збільшення продуктивності системи із збільшенням масштабів діяльності, важливим фактором є і розширення її функціональності. І тут трирівнева архітектура має перевагу над своїми попередниками. Для розширення функціональності досить встановити новий сервер додатку з необхідною функцією і не обов'язково міняти всю систему як у випадку з 2.5-рівневою схемою.

Більшість проблем пов'язаних з перевстановленням клієнтських частин програми на великій кількості комп'ютерів, можливо досить віддалених, стають

неактуальними, на відміну від 2.5 рівневої архітектури – використання "тонкого" клієнта надає для цього багато можливостей [10].

Отже, проаналізувавши переваги та недоліки представлених типів архітектури інформаційної системи, для створення атоматизованої системи керування та обліку мережевого обладнання було обрано трирівневу клієнт-серверну архітектуру, що дозволить отримати досить швидку роботу системи, при цьому не навантажуючи клієнтську сторону, за рахунок використання “тонкого” клієнта, та досить просто збільшувати функціональність і впроваджувати оновлення з меншими витратами.

3.3. Вибір бази даних

В якості інформації, що зберігає база данх, може бути будь-що: каталог продукції, інформація про клієнтів, контент веб-сайту та інше. Для забезпечення доступу до інформації та для управління нею, використовують систему управління базами даних. Система управління базами даних – це сукупність мовних і програмних засобів, що призначена для створення, ведення і спільного використання бази даних багатьма користувачами. Як правило такі системи поділяють за моделями даних, що вони використовують. Системи, що використовують реляційні моделі даних, називають реляційними. Системи управління базами даних допомагають відсортувати та структурувати інформацію, а також пов'язати бази даних між собою, з наданням звіту про зміни і зареєстровані події.

Попри те, що всі системи управління базами даних мають спільне завдання (надавати можливість користувачам створювати, редагувати і отримувати доступ до інформації, що зберігається в базах даних), сам процес виконання цього завдання суттєво відрізняється. До того ж, різні системи можуть мати досить різні функції і можливості, більш або менш докладну документацію та технічну підтримку.

Під час порівняння різних популярних баз даних, варто враховувати, зручність для користувача і масштабованість окремої системи та впевнитися, що вона може бути вдало інтегрована з іншими продуктами, які вже використовуються. Також, під час вибору варто звернути увагу на вартість системи і підтримки, що надається розробником.

Якщо система обирається для підприємства, потрібно враховувати можливість зростання системи разом з розвитком компанії. Для малого бізнесу може бути достатньо базових функції і можливостей, а також невелика кількість інформації, для розміщення в базі даних. Проте вимоги можуть значно збільшуватись з часом, зміна системи може стати проблематичною.

Існує ряд популярних систем управління базами даних, на платній і безоплатній основі, які можна використовуати в організації [11].

Тенденція популярності систем за даними сайту DB-engines представлена на рисунку 3.7 [12].

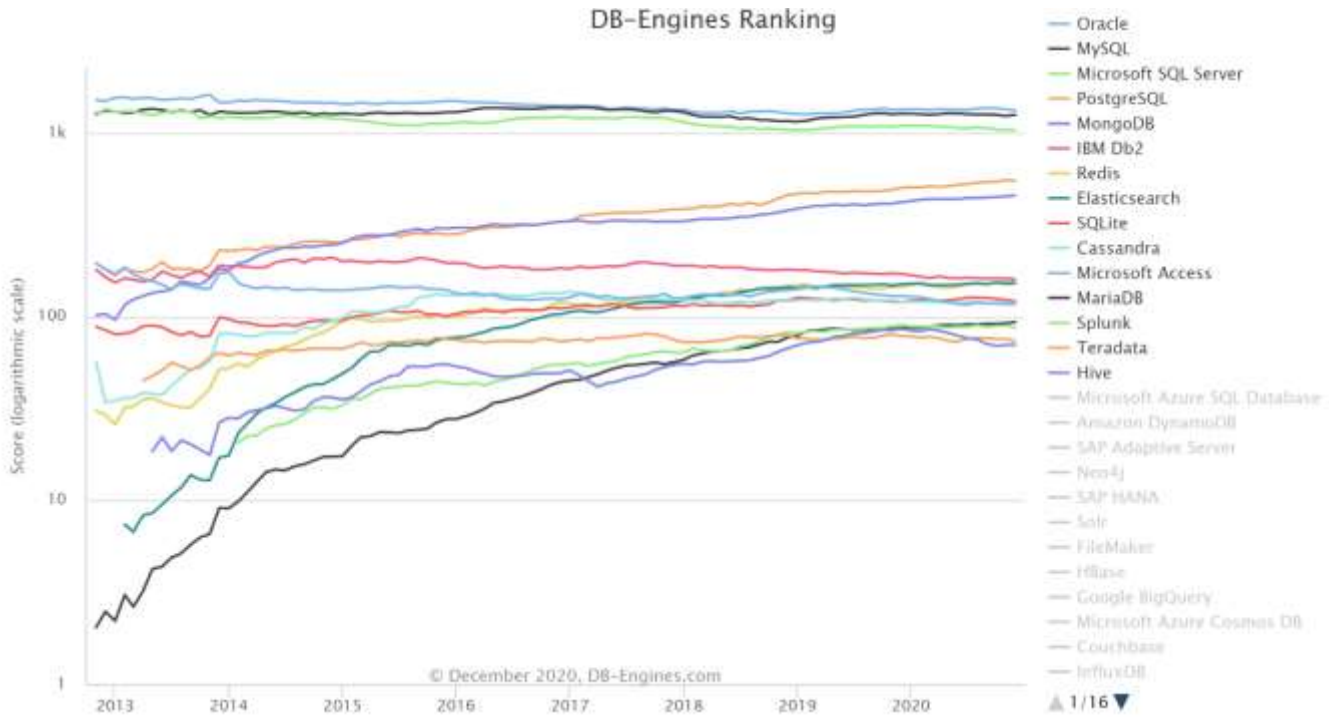

Рис. 3.7. Тенденція популярності систем управління базами даних

3.3.1. Oracle 19c

Не дивно, що компанія Oracle представляє однойменний продукт, що як правило розпочинає перелік варіантів популярних систем управління базами даних. Перша версія Oracle була випущена наприкінці 70-х років, що має до сих пір бездоганну репутацію. Для задоволення потреб конкретних організацій існує декілька версій цього продукту.

Актуальна версія Oracle на даний момент – 19c – призначена для хмарних середовищ і може бути розміщена на одному або декількох серверах, що дозволяє управляти базами даних, які містять мільярди записів.

Це означає, що фізичне управління даними не впливає на доступ до логічних структур. Окрім того, в цій версії кожна транзакція ізольована від інших, що дозволило забезпечити досить високий рівень безпеки.

Переваги:

– оскільки компанія Oracle прагне тримати високу планку у порівнянні з іншими розробниками, то для її системи притаманні найновіші інновації та вражаючий функціонал;

– система від Oracle є дуже надійною, що вважається фактично еталоном надійності серед подібних систем.

Недоліки:

– вартість такої системи, особливо для невеликих організацій, може виявитися значно високою;

– системі може знадобитися значна кількість ресурсів відразу після встановлення, тому можливо потрібно буде оновити обладнання для впровадження Oracle.

Ідеально підходить для великих організацій, які працюють з величезними базами даних і різноманітними функціями [11].

3.3.2. MySQL

MySQL – є однією з найпопулярніших баз даних для веб-додатків. Для веб-серверів, що працюють на операційній системі Linx, фактично є стандартом. MySQL – представляє собою безкоштовний пакет програм, проте має постійні оновлення, що розширюють функціонал і покращують безпеку. Також існують окремі платні версії для використання підприємствами. Головний акцент у безкоштовній версії ставиться на швидку і надійну роботу системи, а не на різноманіття її функцій, що може бути і перевагою і недоліком – в залежності від області застосування.

Розробкою та підтримкою сайта MySQL займається компанія Oracle, яка отримала права на торговельну марку після поглинання Sun Microsystems, яка раніше придбала шведську компанію MySQL AB. Продукт поширюється як під GNU General Public License, так і під власною комерційною ліцензією. Також, розробники додають певний функціонал на замовлення ліцензійних користувачів. Якраз дякуючи такому замовленню майже в перших версіях з'явився механізм реплікації.

Ця система управління базами даних дозволяє обирати різнноманітні движки для системи зберігання, що надають можливість змінювати функціонал інструменту і виконувати обробку даних, які зберігаються в різних типах таблиць. Підтримка великої кількості типів таблиць робить систему MySQL досить гнучкою, що дозволяє користувачам вибирати як таблиці типу MyISAM, які підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Окрім того, MySQL поставляється із спеціальним типом таблиць EXAMPLE, які демонструють принципи створення нових типів таблиць. В MySQL постійно з'являються нові типи таблиць, в наслідок відкритої архітектури і GPL-ліцензуванню. Вона також має простий у використанні інтерфейс, і пакетні команди, які дозволяють зручно обробляти величезні обсяги даних. Система приголомшливо надійна і не намагається використати всі доступні апаратні ресурси.

Переваги:

– безкоштовне розповсюдження;

– має хорошу документацію;

– навіть у безкоштовна версія надає багато функцій;

– простий процес установки на найбільш поширених дистрибутивах операційної системи Linux, оскільки пакет MySQL включений в стандартні репозиторії;

– підтримує набір інтерфейсів користувача;

– може працювати з іншими базами даних, включаючи DB2 і Oracle.

Недоліки:

– потрібно буде витратити багато часу і зусиль, щоб налаштувати MySQL для виконання нескладних завдань, таких як, створення інкрементних резервних копій, хоча інші системи автоматично це виконують;

– відсутність вбудованої підтримки XML або OLAP.

– підтримка для безкоштовної версії здійснюється лише на платній основі.

Відмінно підходить для організацій, яким потрібна надійна система управління базами даних, але безкоштовна [11].

3.3.3. Microsoft SQL сервер

Microsoft SQL-сервер, також є однією з популярних систем управління бащами даних. Ця система працює на движку, що може запускатися як на локальних серверах, так і в хмарному середовищі, до того ж типи використуваних систем можна одночасно поєднувати. Спочатку продукт працював лише на Windows-платформі, але вже після виходу Microsoft SQL сервер 2016, Microsoft адаптувала його для операційної системи Linux.

Функція тимчасової підтримки даних у версії 2016 року, є однією з основних її особливостей, що дозволяє відстежувати зміни даних з плином часу. Остання версія Microsoft SQL-сервер може гарантувати доступ до конфіденційних даних лише для авторизованих користувачів, завдяки підтримці динамічного маскування.

Переваги:

– дуже простий у використанні продукт;

– актуальна версія має швидку і стабільну роботу;

– движок надає можливість знизити використання ресурсів, за рахунок регулювання івідстеження рівнів продуктивності;

– є можливість отримати доступ до візуалізації на мобільних пристроях;

– добре працює в поєднанні з іншими продуктами Microsoft.

Недоліки:

– неприйнятна ціна для більшої частини організацій;

– навіть після ретельного налаштування система може використати всі доступні ресурси;

– повідомляється про проблеми з використанням служби інтеграції для імпорту файлів;

Відмінно підходить для великих організацій, які вже користуються іншими продуктами Microsoft [11].

3.3.4. PostgreSQL

PostgreSQL одна з декількох безкоштовних популярних варіантів систем управління базами даних, часто використовується для ведення баз даних веб-сайтів. Зараз ця система добре розвинена, оскільки була створена однією з перших, вона дозволяє користувачам керувати як структурованими, так і неструктурованими даними. Підходить для використання на більшості основних платформ, включаючи Linux. Завдяки власним інструментам добре справляється із задачами імпорту інформації з інших типів баз даних.

Движок системи може розміщуватись в різних середовищах, в тому числі віртуальних, фізичних і хмарних. Остання версія здійснює обробку великих обсягів даних і дозволяє збільшити кількість одночасно працюючих користувачів. Рівень безпеки був покращей завдяки підтримці DBMS_SESSION.

Переваги:

– добре масштабується та може обробляти великі обсяги даних;

– наявна підтримка формату json;

– існує велика кількість визначених функцій;

– доступно ряд інтерфейсів.

Недоліки:

– документація досить заплутана, тому, можливо, відповіді на деякі питання доведеться шукати в інтернеті;

– конфігурація може збентежити непідготовленого користувача;

– під час проведення пакетних операцій або виконання запитів читання, швидкодія системи може знижуватись.

Ідеально підходить для організацій з обмеженим бюджетом, але кваліфікованими фахівцями, коли потрібна можливість вибрати свій інтерфейс і використовувати json [11].

3.3.5. MongoDB

Ще одна безкоштовна база даних, яка також має і комерційну версію – MongoDB, вона призначена для додатків, що використовують як структуровані, так і неструктуровані дані. Ядро системи дуже гнучке і працює при підключенні бази даних до додатків за допомогою драйверів MongoDB. Можна працювати з будь-якою мовою програмування, оскільки є великий асортимент доступних драйверів і тому знайти потрбіний буде досить легко.

Від початку MongoDB не ставила за мету обробку моделей реляційних даних, хоча і здатна це виконувати, але можуть виникнути проблеми з продуктивністю, якщо спробувати використовувати її таким чином. Втім, движок призначений для обробки різних даних, які не можна віднести до реляційних, все таки може справлятися там, де інші движки працюють повільно або взагалі безсилі.

Актуальна версія, має нову систему движків зберігання. Документи можуть бути перевірені під час оновлення або виконання вставок також була покращена функція текстового пошуку. Нова можливість часткового індексування може призвести до підвищення продуктивності, за рахунок зменшення розмір індексів.

Переваги;

– швидкість і простота у використанні;

– наявна підтримка json та інших традиційних документів NoSQL;

– можна швидко і легко прочитати та зберегти дані будь-якої структури.

Недоліки:

– в якості мови запитів не використовується SQL;

– наявні в MongoDB засоби для перекладу SQL-запитів, варто розглядати саме як доповнення;

– програма встановлення може зайняти багато часу.

Підходить для організацій, що працюють з різнорідними даними, які важко піддаються класифікації. Для впровадження потрібні кваліфіковані спеціалісти.

Зважаючи на зазначені переваги та недоліки представлених систем управління базами даних, для реалізації проекту було прийнято рішення використати MySQL, оскільки функціоналу навіть безкоштовної версії буде достатньо, система досить просто встановлюється та має хорошу документацію, також вона може працювати з іншими базами даних [11].

3.4. Вибір веб-сервера

Web-сервер - це програма, щообробляє повідомлення, які приходять за замовчуванням на 80-й порт, і працює з протоколом HTTP (Hypertext Transfer Protocol). Для мережі Інтернет саме цей протокол є основним. Він заснований на принципі «запит-відповідь» та представляє з себе сукупність правил для обміну даними. Клієнт надсилає до сервера запит, що містить службову інформацію про його тип (дані, заголовок, форма), заголовок запиту (допустимі типи файлів, авторизація, версія клієнта, адреса, де було активовано посилання на даний ресурс, і сам адреса) та запит даних. Сервер надсилає відповідь клієнту, у якій знаходиться службовий код, що показує стан обробки запиту, заголовок відповіді (версію сервера, дату, довжину і тип даних) і самі дані. Так стисло можна поясни як веб-сервер взаємодіє з клієнтом.

Проте останнім часом ринок програмного забезпечення значно виріс. А саме, збільшилася кількість веб-серверів від різних виробників. Як і у випадку з будь-яким продуктом, тут є досить великий вибір за ціною, можливостями, вбудованим розширенням, рівнем технічної підтримки, зручністю встановлення і обслуговування та за багатьма іншими параметрами. Очевидно, будь-який Web-сервер має підтримку базового функціоналу, а саме: підтримка протоколу HTTP,налаштування на різні порти, створення log-файлів, директорії користувача, функції захисту. Наведені вище критерії є найбільш важливим в будь-якій програмі, в тому числі і для веб-серверів. Проте існують деякі нюанси. Як правило проектування системи починається не з

вибору веб-сервера, а з вибору операційної системи. Тут потрібно зауважити, що некожен сервер підримує конкретну операційну систему.

Спершу, перед встановленням серверу, необхідно зрозуміти який функціонал він підтримує. Очевидно, що будь-який сервер може працювати з протоколом HTTP, але не кожен сервер відразу має підтримку баз даних. До цього ж зараз будь-який розробник веб-серверів створює свій API (Application Program Interface) для роботи з сервером. Немале значення при виборі сервера відіграє зручність засобів розробки програм для нього. Варто також зазначити, що, на відміну від багатьох інших продуктів, програмиможуть бути безкоштовними. При цьому «безкоштовне» у випадку веб-серверів, означає якщо не найкраще, то, по принаймні, найпопулярніше. Мається на увазі сервер Apache, який, за оцінками, встановлений приблизно на 60% всіх веб-серверів, і є більш надійним та стабільним в порівнянні з іншими, при тому, що є безкоштовним, вільно розповсюджуваним [13]. Наразі існує досить багато веб-серверів, але популярністю користуються лише декілька, а саме: Apache від ASF (Apache Software Foundation), Nginx від Nginx Inc.та Internet Information Server від Microsoft. Діаграма популярності веб-серверів в 2020 році показана на рисунку 3.8.

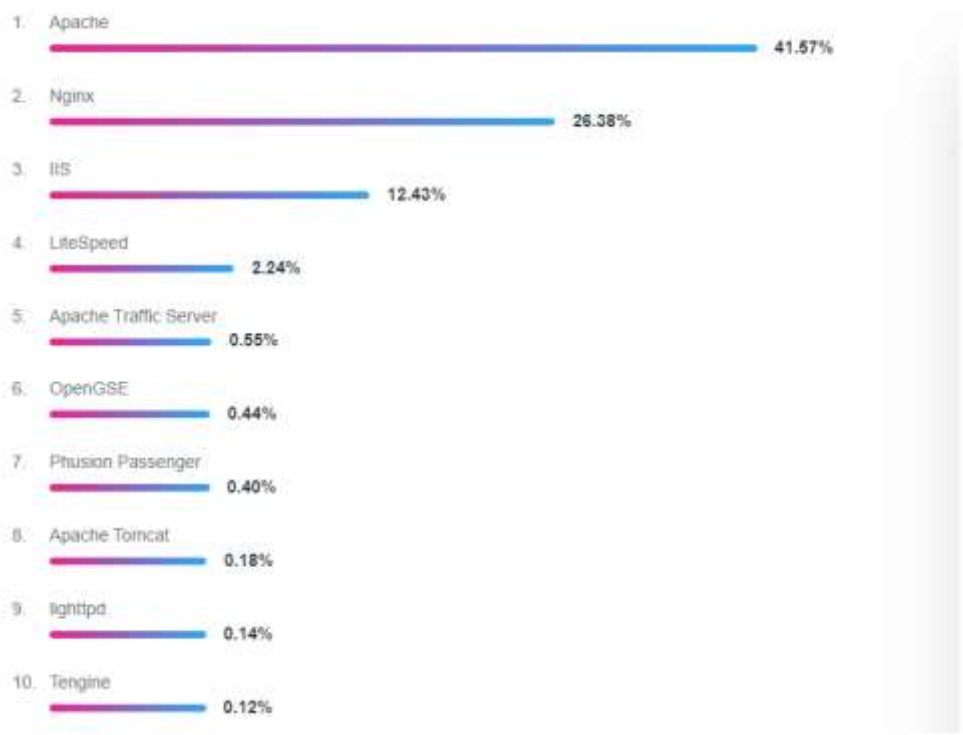

Рис. 3.8. Діаграма популярності веб-серверів в 2020

3.4.1. Apache

Його без перебільшень можна назвати найпоширенішим веб-сервером. Наразі Apache підтримує всі відомі платформи і операційні системи, в тому числі і Windows. Підтримка якої з'явилася лише в останніх версіях, що тільки збільшило популярність веб-сервера. До його беззаперечних переваг відносять надійність, виняткову продуктивність і величезний набір функцій і додаткових модулів. Проте головною особливістю цього сервера є розповсюдження за безкоштовною ліцензією. Окрім заощадження фінансових витрат, це надає можливість, швидко виправляти помилки і вносити в код програми необхідні доповнення. Слід відзначити розробників веб-сервера Apache, – це люди які не тільки безкоштовно, а й головне, дуже швидко виправляють знайдені помилки. Окрім стандартної версії є велика кількість модулів, що значно розширюють можливості Apache. Також вдається створювати найбільш захищені модулі, завдяки вільному розповсюдженню і відкритому коду. В цьому плані будь-якому комерційному конкуренту Apache важко протистояти. Проте, не зважаючи на всі вище зазначені переваги, є чимале упущення, яке все-таки важко назвати недоліком: у Apache немає зручної програми встановлення і управління. Під час встановлення доводиться використовувати командний рядок. Іноді навіть доводиться вносити корективи в include-файли, при установці під UNIX. Як правило, є детальна інструкція того, що і як необхідно змінити, але це все-таки виглядає не дуже зручним на перший погляд. Під час роботи доводиться мати справу з тестовими файлами, а не з програмою у вигляді вікна. Технічна підтримка полягає не в гарних help-файлах або організаціях, що надають фахівців, а в основному завдяки телеконференціям і обміну думками системних адміністраторів. Насправді організації, що надають послуги технічної підтримки, все ж є, але це коштує грошей. Проте такий нетиповий для серйозного продукту стиль не робить Apache менш популярним, і на ринку він є найсерйознішим конкурентом для всіх веб-серверів [13].

Короткі характеристики веб-сервера Apache:

– безкоштовний та має відкритий код;

– має підтримку наступних операційних систем: NetBSD, Digital UNIX, BSDI, AIX, OS / 2, SCO, HPUX, Novell NetWare, Macintosh, Be OS, Windows NT, Linux, Windows 95, FreeBSD, Windows 98, IRIX, Solaris;

– може створювати декілька log-файлів;

– на Windows запускається у вигляді сервісу та/або програми, під UNIX може запускатися з inetd;

– дозволяє налаштування на декілька портів;

– підтримує Windows CGI, HTTP / 1.1, в тому числі і HTTP / 1.1 PUT;

– є функція автоматичної відповіді при внесенні змін до документа, підтримує Microsoft ISAPI;

– можливе блокування доступу із зазначених адрес, до певних документів, заборона запуску CGI скриптів, окремим користувачам, можливо вносити зміни без перезапуску сервера;

– поставляється разом з повним вихідним кодом, має підтримку інших протоколів (ftp, telnet), має налаштування на директорії користувачів, містить модуль proxy [13].

3.4.2. Nginx

Nginx був розроблений для усунення обмежень продуктивності веб-серверів Apache. Продуктивність і масштабованість Nginx обумовлені архітектурою, що керується подіями. Він значно відрізняється від підходу Apache. У Nginx кожен робочий процес може одночасно обробляти тисячі HTTP-з'єднань. Отже, Nginx – це легка, масштабована і високопродуктивна реалізація. Така архітектура робить процес обробки великих і флуктуючих навантажень на дані більш передбачуваною з боку використання оперативної пам’яті та центрального процесора і затримки.

Nginx також має велику кількість функцій і може виступати в якості різних серверів:

– зворотний проксі-сервер для протоколів HTTP, HTTPS, SMTP, POP3 та IMAP;

– балансувальник навантаження і HTTP-кеш;

– інтерфейсний проксі для Apache та інших веб-серверів, що дозволя поєднати гнучкість Apache з хорошою продуктивністю статичного контенту Nginx.

Nginx має складну архітектуру, тому розробка модулів для нього є нелегкою. Розробники модулів Nginx мають бути дуже обережними, щоб створювати ефективний і точний код, без збоїв, і відповідним чином взаємодіяти зі складним ядром, що керується подіями, щоб уникнути блокування операцій.

Nginx підтримує декілька сучасних Unix-подібних систем та Windows, проте більш продуктивно і стабільно він працює платформах UNIX, а ніж на Windows.

Nginx Plus – це програмний балансувальник навантаження, веб-сервер і кеш контенту, що побудований на основі відкритого вихідного коду Nginx, а також використовує модульну архітектуру. За необхідністю, до працюючого екземпляру Nginx Plus, можуть бути підключені нові функції і можливості за допомогою програмних модулів. Динамічні модулі додають в Nginx Plus такі функції, як геолокація користувачів за IP-адресою, зміна розмірів зображень та додавання сценаріїв Lua в модель обробки подій Nginx Plus. Модулі створюються як самою компанією, так і сторонніми розробниками [14].

3.4.3. Internet Information Server

Сервер IIS є найбільш вдалим вибором для платформи Windows, оскільки краще за розробників самої операційної системи навряд чи хтось зможе розробити програму, що повністю використовуватиме її можливості. На процес встановлення та налаштування витрачається не більше десяти хвилин. Варто зазначити, що сервер використовує досить малий об’єм дискового простору та може працювати на досить слабкій машині. Але нажаль, робота сервера підтримується тільки на Windows 2000 Server. Також добре відтворена і система керування сервером. Для часто виконуваних і рутинних дій є декілька корисних програм шаблонів (Wizards). IIS підтримує загальновідомі та нові стандарти захисту, такі як: SSL 3.0, Kerberos 5.0, і нового методу Fortezza. Найбільш цікавою і відмінною особливісю IIS є підтримка WebDAV (Web-based Distributed Authoring and Versioning). Цей нещодавно винайдений стандарт, дозволяє перетворювати внутрішні мережі на спільний простір,

використовуючи ресурси сусідніх комп'ютерів в якості своїх. На практиці це означає, що користувачі IIS можуть доволі зручно розподіляти свої робочі файли і мати при цьому можливість обмежувати до них доступ. Але також існують деякі проблеми несумісності. Не зовсім коректно відбувається робота одночасно з Front Page Server, іноді при внесенні змін до конфігурації доводиться перезапускати систему, зустрічаються помилки при використанні системи віддаленого адміністрування [13].

Основні характеристики:

– входить до складу Windows 2000 Server;

– підтримує операційні системи: Windows 2000 Server, Advanced Server;

– маж підтримку ASP, Microsoft API, ODBS;

– має можливість створювати декілька log-файлів, в тому числі окремо для кожного CGI-скрипта;

– протоколює продуктивніть;

– на Windows запускається у вигляді сервіс та/або програми, під UNIX може запускатися з inetd;

– дозволяє налаштування на декілька портів;

– до поставки включений SNMP-агент; надає доступ до змінних стану сервера з CGI, HTTP / 1.1, в тому числі і HTTP / 1.1 PUT;

– є функція автоматичної відповіді при внесенні змін до документа, підтримує Microsoft ISAPI;

– можливе блокування доступу із зазначених адрес, до певних документів, окремим користувачам;

– наявна підтримка S-HTTP;

– зміни можна вносити без перезавантаження сервера;

– є підтримка SSL другої і третьої версій та авторизації;

– графічна програма встановлення та та управління;

– має підтримку інших TCP-протоколів (ftp, telnet);

– присутня програма для оцінки продуктивності під час роботи;

– підтримка директорій користувачів;

– вбудований алгоритм пошуку;

– можливість віддаленого керування.

Отже розглянувши переваги та недоліки представлених веб-серверів, в якості веб-сервера для розроблюваної системи було обрано Apache, оскільки він має докладну документацію, хорошу підтримку з боку спільноти, завдяки безкоштовному розповсюдженню та однаково швидко і стабільно працює, як на Windows так і на UNIX системах [13].

Висновки

У третьому розділі розглянуто функціональні та нефункціональні вимоги до розроблюваної системи з урахуванням переваг та недоліків існуючих продуктів. Представлено різноманіття архітекур інформаційних систем, проаналізовано їх сильні і слабкі сторони та обрано для реалізації трирівневу клієнт-серверну архітектуру, що забезпечить досить швидку роботу системи та надасть можливість досить просто додавати нові функції. Розглянуто ряд найпопулярніших систем управління базами даних та для реалізації обрано базу даних MySQL, оскільки її функціоналу буде повністю достатньо та вона має хорошу документацію. Також було розглянуто популярні веб-сервери та обрано для використання Apache, оскільки він також має хорошу документацію та підтримку спільноти і досить швидко та стабільно працює.

РОЗДІЛ 4
ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

4.1. Структурна схема системи

Для представлення складових частин та взаємозв'язків між ними було використано структурну схему (рис. 4.1).

Рис. 4.1. Структурна схема

Система керування та обліку мережевого обладнання складається з трьох основних частин: підсистеми керування мережевим обладнанням, підсистеми обліку та підсистеми інтерфейсу користувача.

Підсистема керування – призначена для безпосерельного керування мережевим обладнанням за допомогою спеціалних скриптів та мережевих протоколів.

Підсистема обліку – призначана для зберігання інформації про мережеве обладнання та складається з бази даних.

Підсистема інтерфейсу користувача – призначена для взаємодії користувача з іншими складовими системи, саме через цю підсистему надходить інформація для обробки та за допомогою неї виводиться результат роботи системи користувачу.

4.2. Опис таблиць бази даних

Для зберігання даних системи використовується реляційна база даних MySQL. Дані розподілені між таблицями які використовують зв'язки багато-до-багатьох, та

один-до-багатьох. Детальна інформація про їх структури (ім’я, тип і розмір поля, опис поля) наведено у таблицях 4.1. – 4.3.

Таблиця 4.1.

Структура таблиці “Користувач”

Ім’я поля	Тип і розмір поля	Опис поля
Id	int(10)	Первинний ключ
UserName	varchar(30)	Ім’я облікового запису
LastName	varchar(50)	Призвіще користувача
FirstName	varchar(50)	Ім’я користувача
Email	varchar(20)	Електрона пошта

Таблиця 4.2.

Структура таблиці “Модем”

Ім’я поля	Тип і розмір поля	Опис поля
Id	int(20)	Первинний ключ
Name	varchar(30)	Назва обладнання
Ip	varchar(15)	Ip адреса
Mac	varchar(16)	MAC адреса
Model	varchar(30)	Назва моделі
Serial	varchar(40)	Серійний номер
City	varchar(50)	Місто розташування
Street	varchar(50)	Вулиця розташування
House	varchar(50)	Номер будинку
Floor	int(3)	Поверх
Login	varchar(10)	Назва користувача для віддаленого доступу
Password	varchar(20)	Пароль для віддаленого доступу

Таблиця 4.3.

Стуктура таблиці “Модель модема”

Тип і розмір поля	Опис поля	Тип і розмір поля
Id	int(20)	Первинний ключ
Name	varchar(30)	Назва моделі
Firmware	varchar(30)	Актуальна версія програмного забезпечення
Port_count	int(3)	Кількість портів

В наведених вище таблицях зберігаються всі портібні дані для функціонування системи, а семе: інформація про користувачів (призвіще, ім’я, назва облікового запису та інше), інформація про мережеве обладнання (назва, ip адреса, mac адреса, серійний номер, фізичне місце розташування, модель та інше) та інформація про конкретну модель мережевого обладнання (назва, актуальна версія програмного забезпечення, кількість портів).

4.3. Керування мережевим обладнанням за допомогою розроблюваної системи

Керування мережевим обладнанням в розроблюваній системі здійснюється за допомогою різноманітних скриптів, одним з яких є upload_cfg.py написаний на мові програмування Python, він дозволяє завантажувати з tftp-сервера на мережеве обладнання потрібний файл конфігурації. Цей скрипт викликається з веб-сторінки системи.

Коротко про роботу скрипта.

Запит скрипта до бази даних:

gw = sql.query("SELECT m.ip as ip, m.id as id, m.name as name, mm.model as model, d.name as domain, m.snmp_community as community, p.template as template, m.uplink as uplink,mac as mac FROM modems as m LEFT JOIN modems_models as mm ON mm.id = m.model_id LEFT JOIN domains as d ON d.id = m.domain LEFT JOIN modem_profiles as p ON p.id = m.profile WHERE ip like '" + str(string) + "'")

Виклик іншого скрипта для генерації потрібного файлу конфігурації:

r = requests.get('http://10.0.0.100/scripts.php?cmd=/usr/local/bin/CFG-GEN/init.py {ip} {model} 3'.format(ip=ip, model=model.lower()))

Підключення за допомогою протоколу telnet та стандартних логіну і паролю до мережевого обладнання, на ip адресу яка заздалегідь налаштована в базовій конфігурації, для кожної моделі обладнання ця ip адреса різна.

```
with telnet_class.TelnetDevice(realip, timeout=5) as sw:
    try:
      sw.login('admin','admin')
      except Exception:
      print('Проблема з підключенням до обладнання, первірте доступ до нього')
return
```

Перевірка версії програмного забезпечення обладнання.

```
version = sw.execute('show version')
 if 'Operation Code Version : {}'.format(sql.query('SELECT firmware from modems_models WHERE model="{}"'.format(model))[0]['firmware']) not in version:
      print('<h1><b>На комутаторі використовується неактуальна версія програмного забезпечення!</b></h1>')
      sw.execute('exit')
```

Завантаження згенерованого раніше файлу конфігурації з tftp-сервера.

```
print('<b>Завантажується конфигурація з файлу {}</b>\n'.format(cfg_file_name))
                sw.execute('copy tftp startup-config', wait_until='address:')
                sw.execute('10.0.0.100 ', wait_until='name:')
                sw.execute(cfg_file_name, wait_until=']:')
                sw.execute('auto_upload.cfg', timeout=15)
                sw.execute('reload', wait_until='<y/n>?')
                sw.execute('y')
                print('<b>Виконано. Перевірьте обладнання після перезавантаження (через 2-3 хвилини).</b>')
```

Повний код скрипта наведено в додатку А.

4.4. Робота з графічним інтерфейсом користувача

Для роботи з системою користувач спочатку повинен авторизуватись в ній і тільки тоді він зможе користуватись її функціоналом, відповідно до його прав доступу. Вікно авторизації представлено на рисунку 4.2.

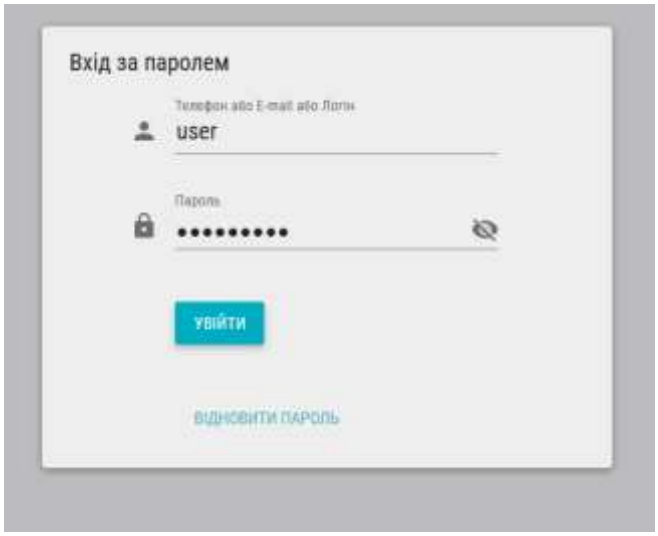

Рис. 4.2. Вікно авторизації

Після успішної авторизації користувач може переглянути список всього обладнання або виконати пошук за допомогою фільтрів (рис. 4.3.).

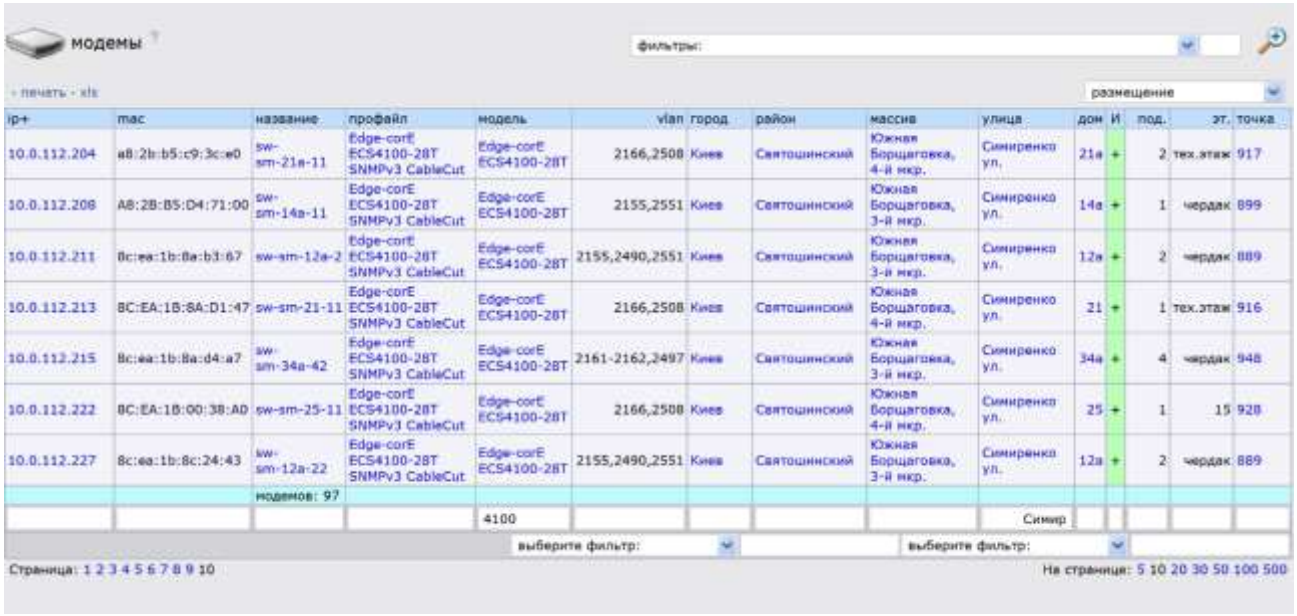

Рис. 4.3. Вікно відображення результатів пошуку

Якщо користувач має відповідні права, він може переглянути більш детальну інформацію про обладнання вибравши його (рис. 4.4.).

Рис. 4.4. Перегляд додаткової інформації про обладнання

Можна отримати інформацію із самого обладнання, таку як: назва портів, статус портів, результат тесту кабеля. Для цього потрібно вибрати будь-яке обладнання із списку, перейти у вкладку Дія та вибрати із випадаючого списку, дію яку потрібно виконати, наприклад “Показати інформацію за портами” (рис. 4.5.).

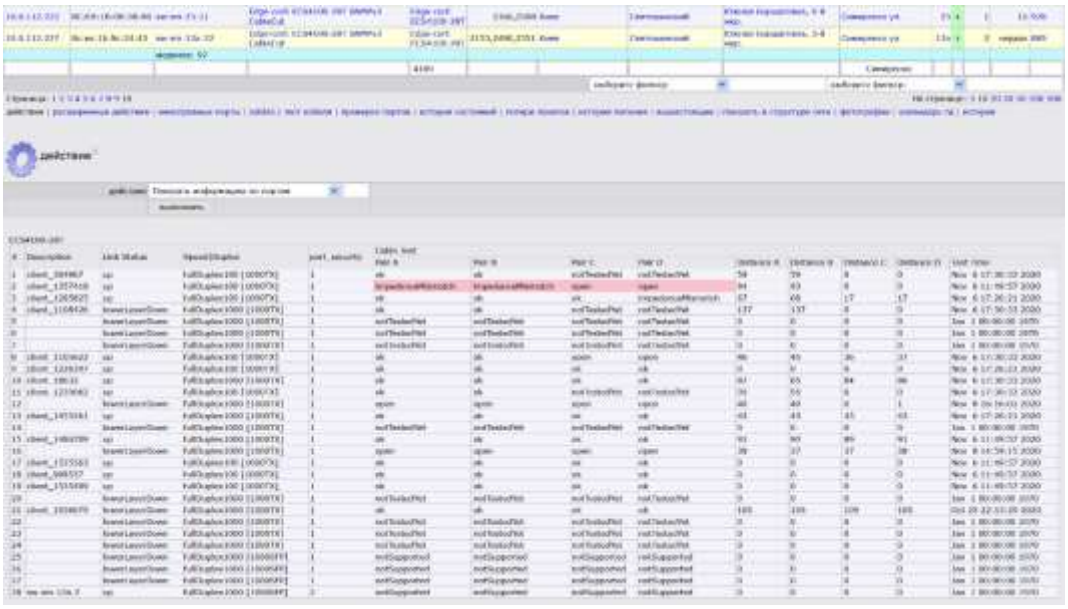

Рис. 4.5. Результат виконання дії “Показати інформацію за портами”

Можна перезавантажити обладнання не заходячи на нього і не користуючись інтерфейсом командного рядка, достатньо обрати потрібне обладнання, перейти до вкладки Дія та у випадаючому списку вибрати відповідну операцію “Reboot Edge-core”, та натиснути клавішу Виконати, обладнання буде перезавантажене за допомогою протоколу SNMP та з’явится повідомлення, що відповідний snmpset запит відправлено до обладнання (рис. 4.6.).

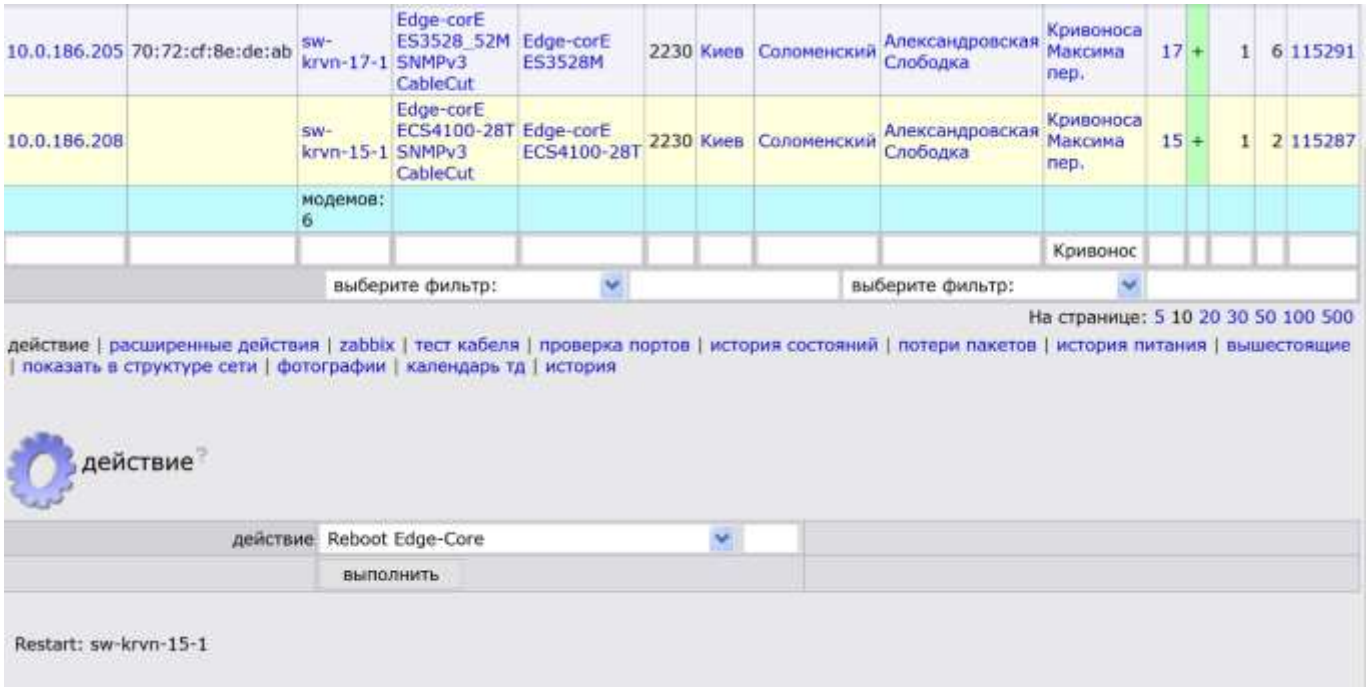

Рис. 4.6. Результат виконання операції “Reboot Edge-core”

Якщо обладнання замінили на іншу модель або встановили нове обладнання і потрібен файл конфігурації, його можна згенерувати за допомогою відповідної операції у вкладці Дії, у випадаючому списку – “Згенерувати конфіг для (модель обладнання)”. Після натиснення клавіші Виконати, на tftp-сервері буде згенеровано файл конфігурації, а для користувача буде показана підказка, які комнди слід ввести на обладнанні, щоб завантажити файл конфігурації на нього (рис. 4.7.).

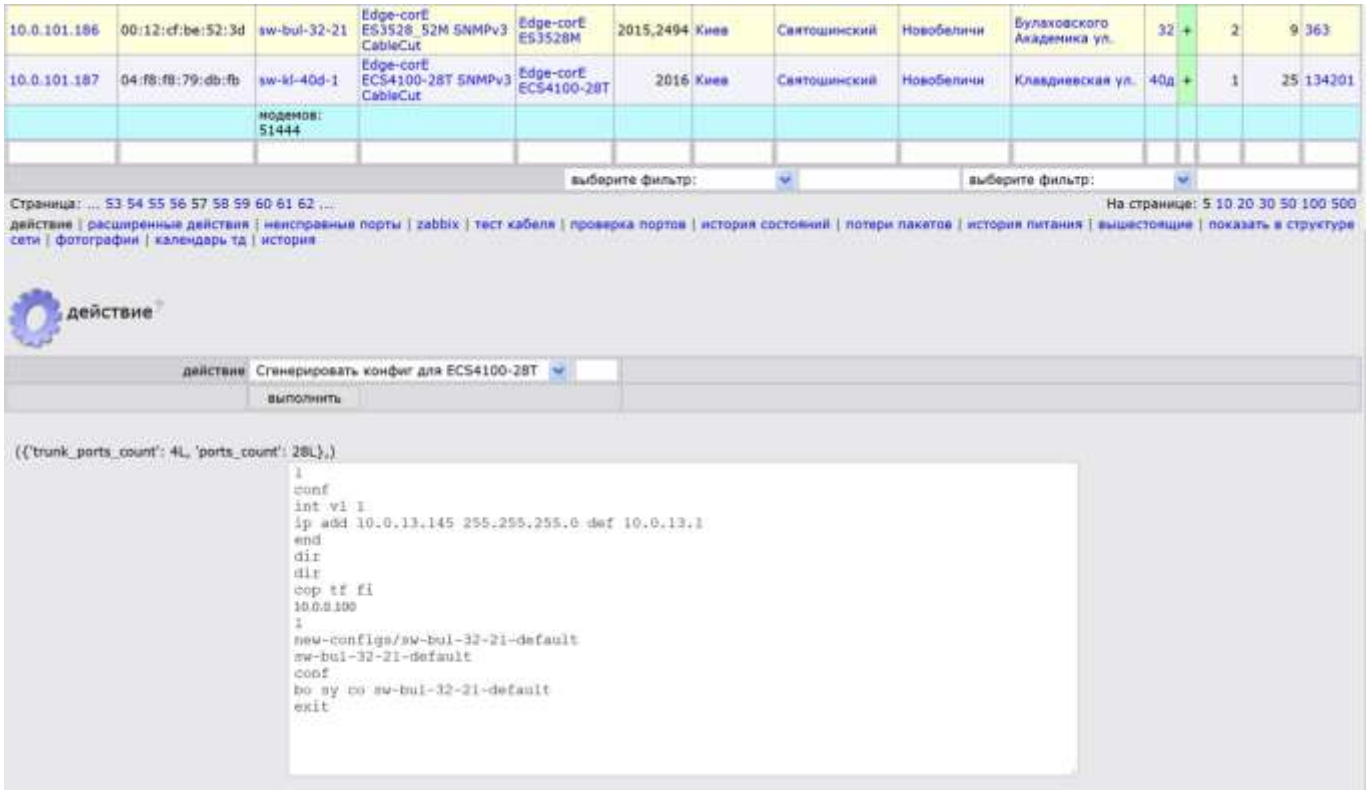

Рис. 4.7. Результат виконання операції “Згенерувати конфіг для ECS4100-28T”

Якщо потрібно буде релізувати пітримку нового функціоналу для управління обладнанням, можна буде внести зміни до існуючих скриптів або створити новий та додати відповідну операцію у випадаючому списку для його виклику.

Висновки

В четвертому розділі представлено структурну схему системи, описано структури таблиць баз даних, та призначення цих таблиць. Представлено частину програмної реалізації системи. Показано роботу системи з використанням графічного інтерфейсу.

ВИСНОВКИ

В даній дипломній роботі розглянуто різноманіття мережевого обладнання та способів керування ним. Проаналізовано ринок існуючих систем керування та обілку мережевого обладнання. Приділено увагу до функціональних те нефункціональних вимог системи. Опрацьовано ряд архітектурних рішень з порівнянням переваг та недоліків кожнного.

У дипломній роботі описано проектування автоматизованої системи керування та обліку мережевого обладнання і засобів, що використовувались. Вдалося реалізувати програмний засіб завантаження файлу конфігурації на пристрій та інтерфейс користувача як частини автоматизованої системи. Це дозволить спростити та зменшити витрати часу на конфігурацію мережевого обладнання.

Даний програмний продукт сприятиме автоматизації роботи з мережевим обладнанням, без використання інтерфейсу командного рядка, який до того ж у кожного виробника різний, дозволить взаємодіяти з обладнанням користувачам, яким не потрібно буде детально вивчати роботу обладнання та його функції. Все це призведе до більш простої взаємодії технічних спеціалістів та менш обізнаних користувачів з мережевим обладнанням. А можливість переглядати та зберігати інформацію про пристрої, дозволить більш зручно керувати їх обігом, і допоможе більш зручно та якісно побудувати свою мережеву архітектуру з наявних пристроїв або полегшить задачу складання переліку для їх закупівель.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Аналіз мережевого обладнання для побудови мережі інтернет-провайдера [Електронний ресурс] / С. С. Волошко, А. Ю. Фролов – Режим доступу до ресурсу: http://reposit.nupp.edu.ua/xmlui/bitstream/handle/PoltNTU/4575/%d0%a1%d1%82%d0%b0%d1%82%d1%82%d1%8f%20%d0%9d%d0%86%d0%a1%d0%a2%20%d0%a4%d1%80%d0%be%d0%bb%d0%be%d0%b2.pdf?sequence=1&isAllowed=y.

2. Обзор и обновления решений Juniper Networks по маршрутизации, коммутации и безопасности [Електронний ресурс] / Muk. – 2016. – Режим доступу до ресурсу: https://habr.com/ru/company/muk/blog/280338/.

3. Сетевое оборудование Cisco [Електронний ресурс] / Akvilona – Режим доступу до ресурсу: http://www.akvilona.ru/serv/cisco.htm.

4. Протоколы и интерфейсы управления проводных сетей доступа [Електронний ресурс] / studbooks – Режим доступу до ресурсу: https://studbooks.net/2179548/informatika/protokoly_upravleniya_setyu_dostupa.

5. Протокол TELNET и SSH [Електронний ресурс] / chin28.narod – Режим доступу до ресурсу: http://www.chin28.narod.ru/d7.3.htm.

6. Функции систем управления [Електронний ресурс] / iptcp – Режим доступу до ресурсу: http://iptcp.net/funktsii-sistem-upravleniya.html.

7. Хранители сети. Open source утилиты для управления, мониторинга и бэкапа настроек сетевого оборудования [Електронний ресурс] / Мартин Пранкевич. – 2015. – Режим доступу до ресурсу: https://xakep.ru/2015/08/10/network-management/.

8. Ansible v.s. Salt (SaltStack) v.s. StackStorm [Електронний ресурс] / Ігор Олемський. – 2017. – Режим доступу до ресурсу: https://habr.com/ru/company/southbridge/blog/330594/.

9. 10 лучших программ для инвентаризации сети 2020 [Електронний ресурс] / softinventive – Режим доступу до ресурсу: https://www.softinventive.ru/best-network-inventory-tools/.

10. Архитектура информационных систем [Електронний ресурс] / it-claim – Режим доступу до ресурсу: http://it-claim.ru/Education/Course/ISDevelopment/Lecture_3.pdf.

11. Сравнение современных СУБД [Електронний ресурс] / Владимир Драч. – 2017. – Режим доступу до ресурсу: https://drach.pro/blog/hi-tech/item/145-db-comparison.
12. Ranking database management systems according to their popularity [Електронний ресурс] / db-engines. – 2020. – Режим доступу до ресурсу: https://db-engines.com/en/ranking.
13. Общее и частное о Web-серверах [Електронний ресурс] / Алексей Кошелев – Режим доступу до ресурсу: https://compress.ru/article.aspx?id=11744.
14. APACHE VS NGINX – СРАВНЕНИЕ И ПРЕИМУЩЕСТВА [Електронний ресурс] / МЕРИОН НЕТВОРКС. – 2020. – Режим доступу до ресурсу: https://wiki.merionet.ru/servernye-resheniya/34/apache-vs-nginx-sravnenie-i-preimushhestva/.

Текст скрипта upload_cfg.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import sys, os, time

sys.path.insert(1, os.path.abspath(os.path.dirname(sys.argv[0]) + '/lib'))
sys.path.insert(1, os.path.abspath(os.path.dirname(sys.argv[0]) + '/templates'))
sys.path.insert(1, os.path.abspath(os.path.dirname(sys.argv[0]) + '/CFG-GEN'))
from snmp import Snmp
from subprocess import Popen, PIPE, STDOUT
import switches
import random
import multiprocessing
import sql_data
import telnetlib
import requests

sql = sql_data.mysql('config', {'config_file': 'config.ini', 'host': '10.0.0.200'})

def build_profile(id):
    query = 'php /var/www/html/stat/tools/call.php -evs modems/modemTool profile ' + str(id) + ' info'
    print query
    print os.system(query)

def daemon(ip, name, id, realip, model):
    if model in ['ECS4120-28F', 'ECS4510-28F', 'ECS4120-28FV2']:
        # Generate configuration
```

```
    r = requests.get('http://10.0.0.100/scripts.php?cmd=/usr/local/bin/CFG-GEN/init.py {ip} {model} 3'.format(ip=ip, model=model.lower()))
    if int(r.status_code)==200:
        cfg_file_name = r.text[r.text.find('new-configs'):r.text.find('\n',r.text.find('new-configs'))].strip()
        if len(cfg_file_name) in range(0,50):
            #Config OK, telnet to switch and upload cfg:
            import telnet_class
            import config
            import logging.config
            logging.config.dictConfig(config.logging)
            with telnet_class.TelnetDevice(realip, timeout=5) as sw:
                try:
                    sw.login('admin','admin')
                except Exception:
                    print('Проблема з підключенням до обладнання, первірьте доступ до неї')
                    return
                version = sw.execute('show version')
                if 'Operation Code Version : {}'.format(sql.query('SELECT firmware from modems_models WHERE model="{}"'.format(model))[0]['firmware']) not in version:
                    print('<h1><b>На комутаторі використовується неактуальна версія програмного забезпечення!</</b></h1>')
                    sw.execute('exit')
                    return

                print('<b>Завантажується конфигурація з файлу {}</b>\n'.format(cfg_file_name))
                sw.execute('copy tftp startup-config', wait_until='address:')
```

```
            sw.execute('10.0.0.100 ', wait_until='name:')
            sw.execute(cfg_file_name, wait_until=']:')
            sw.execute('auto_upload.cfg', timeout=15)
            sw.execute('reload', wait_until='<y/n>?')
            sw.execute('y')

            print('<b>Виконано. Перевірьте обладнання після перезавантаження (через 2-3 хвилини).</b>')
        return
    else:
        p = multiprocessing.current_process()
        print 'Starting:', p.name, p.pid
        print "<font color=green>Loading config. Please check switch in 2 minute</font>"
        sys.stdout.flush()
        os.system("/usr/local/bin/CFG-GEN/init.py " + ip + " " + model.lower() + " 3 >/dev/null")
        time.sleep(5)
        switches.Collective(realip, 'private', 2, '').upload_file(model, 'new-configs/' + str(name), str(name),'config', '10.0.0.100')
        time.sleep(60)
        switches.Collective(realip, 'private', 2, '').set_startup(model, str(name))
        time.sleep(30)
        switches.Collective(realip, 'private', 2, '').reboot(model)
        print 'Exiting :', p.name, p.pid
        sys.stdout.flush()

if __name__ == '__main__':

    if len(sys.argv) < 4:
```

```
    sys.exit('Usage: %s ip/id model op_id' % sys.argv[0])
    pass
else:
    string = sys.argv[1]
    model = sys.argv[2].upper()
    operator_id = sys.argv[3]

gw = sql.query(
    "SELECT m.ip as ip, m.id as id, m.name as name, mm.model as model, d.name as domain, m.snmp_community as community, p.template as template, m.uplink as uplink,mac as mac FROM modems as m LEFT JOIN modems_models as mm ON mm.id = m.model_id LEFT JOIN domains as d ON d.id = m.domain LEFT JOIN modem_profiles as p ON p.id = m.profile WHERE ip like '" + str(string) + "'")
if 'xx' in gw[0]['name']:
    print('<h1>Для коммутатора {name} не має можливості віддалено завантажити конфигурацію!</h1>'.format(name=gw[0]['name']))
    exit()
ip = gw[0]['ip']
mac = gw[0]['mac']
domain = gw[0]['domain']
id = gw[0]['id']

if str(ip) == '':
    print 'no ip'
else:
    checkip = {
        'ES3528M': '172.16.1.10',
        'ES3552M': '172.16.1.11',
        'ES3510': '172.16.1.12',
```

```
        'ES3510MA': '172.16.1.13',
        'ECS4210-28T': '172.16.1.14',
        'ES3528MV2': '172.16.1.15',
        'ES4612': '172.16.1.16',
        'ES4626-SFP': '172.16.1.17',
        'ECS4510-28F': '172.16.1.18',
        'ECS4210-12T': '172.16.1.19',
        'ECS4100-28T': '172.16.1.20',
        'ECS4120-28F': '172.16.1.24',
        'ECS4120-28FV2': '172.16.1.25'
    }
    swip = checkip[model]

    switch = switches.Collective(swip, 'private', '2', '')
    check_model = switch.identify()[1]
    check_mac = Snmp(swip, 'private', 2, '').get('.1.3.6.1.2.1.17.1.1.0').replace(' ',
':').replace(':"',"").replace('"',"")

    if str(check_model) == 'None':
        print "<font color=red>ERROR</font>: UKNOWN MODEL OR SWITCH IS
DOWN"

    elif model != check_model:
        print "<font color=red>ERROR</font>: THE MODEL MISTMATCH"

    else:
        if str(mac).upper() != str(check_mac):
            print "<font color=red>ERROR</font>: MAC MISMATCH, ADD VALID MAC
INTO BILLING: " + str(check_mac)
```

```
        update = sql.query("UPDATE modems SET mac ='" + str(check_mac) + "' WHERE id = '" + str(id) + "';")


        name = domain + "." + str(gw[0]['name']) + "-default"
        d = multiprocessing.Process(name='daemon', target=daemon, args=(ip, name, id, swip, model))
        d.daemon = True
        d.start()
        d.join()
```