МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

_____червня 2023р.____

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерних наук
освітньо-професійної програми "Інформатика"
на тему: "Графічний інтерфейс автоматизованого конфігурування обладнання локальної мережі офісу ІТ-компанії"
здобувача групи ІН-94-1 Вініченка Артема Івановича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Артем ВІНІЧЕНКО

_____

Науковий керівник,
кандидат фізико-математичних наук,
старший викладач кафедри
комп'ютерних наук Дмитро ВЕЛИКОДНИЙ _____

Суми - 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«Затверджую»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»

здобувача групи ІН-94-1 Вініченка Артема Івановича

1. Тема роботи: «Графічний інтерфейс автоматизованого конфігурування обладнання локальної мережі офісу ІТ-компанії»
затверджую наказом по СумДУ від _____

2. Термін здачі здобувачем кваліфікаційної роботи до 09 червня 2023 року___

3. Вхідні дані до кваліфікаційної роботи _____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) огляд технологій та протоколів; 2) постановка завдання для розробки; 3) створення моделей локальної мережі з підтримкою технологій VLAN та протоколів DHCP та NAT; 4) створення графічного інтерфейсу для конфігурації локальної мережі офісу ІТ-компанії.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	Огляд технологій та протоколів		
2	Постановка завдання для розробки		
3	Створення моделей локальної мережі з підтримкою технологій VLAN та протоколів DHCP та NAT		
4	Створення графічного інтерфейсу для конфігурації локальної мережі офісу IT-компанії		

Здобувач вищої освіти _____ Керівник _____

АНОТАЦІЯ

Записка: с. 57, рис. 19, додаток 1, джерел 9.

Обґрунтування актуальності теми роботи – тема кваліфікаційної роботи є актуальною, так як присвячена створенню веб-інтерфейсу для налаштування мережевих приладів без спеціальних знань їх використання у професійній сфері, використовуючи сучасні технології та протоколи роботи у локальній мережі офісу.

Об’єкт дослідження — огляд технологій VLAN та протоколів DHCP та NAT.

Мета роботи — розробка локальної мережі офісу IT-компанії с використанням технологій VLAN та протоколів DHCP та NAT, та створення графічного інтерфейсу для автоматизованого конфігурування обладнання локальної мережі.

Методи дослідження — моделювання комп'ютерної мережі в симуляторі Cisco Packet Tracer.

Результати — створено локальну мережу офісу IT-компанії на основі протоколів DHCP та NAT та технології VLAN на базі програмного забезпечення Cisco Packet Tracer. Спроектовано графічний інтерфейс автоматизованої конфігурації системи мережі для спрощення налаштування мережі з вказанням лише декількох параметрів у самому інтерфейсі

КОМП'ЮТЕРНА МЕРЕЖА,
VLAN, DHCP, NAT, CISCO, PACKET TRACER,
ГРАФІЧНИЙ ІНТЕРФЕЙС.

ЗМІСТ

ВСТУП

Постійний розвиток технологій у сфері мереж зумовлений попитом користувачів для прискорення та автоматизації роботи компаній різноманітних направлень, починаючи з роботи банків, та закінчуючи сферою хмарних технологій, що працюють з великою кількістю інформації, та її транслювання.

На кваліфікаційній роботі ілюструється вирішення класичних проблем роботи великої кількості пристроїв завдяки технологіям та протоколам комутації.

DHCP – протокол, що вирішує проблему статичного налагодження IP адрес на кожен пристрій, а саме прискорює даний процес в геометричній прогресії, в залежності від кількості, як наприклад, комп’ютерів.

При підключенні пристроїв один до одного необхідно звернути увагу на технологію, що дозволить раціонально використати кожен пристрій. Для налагодження локальної мережі та становлення комунікації між ними компанії використовують свічі. Велику кількість комп’ютерів, сканерів, принтерів і так далі не слід під’єднати до свіча у вільний порт Ethernet, бо по-перше, це не є безпечним, так як даний трафік передається кожному користувачу мережі, по-друге, велика кількість під’єднаних приладів створюють широкомовний домен, у якому весь трафік всіх приладів змішується та створює затримки у роботі свіча. Технологія VLAN дозволяє більш точно відправляти трафік, та розмежовувати домени для полегшення роботи локальної мережі.

Інколи існує потреба у вихід до Інтернету з локальної мережі, але з локальними IP адресами вихід в глобальну систему неможливий, тому існує технологія NAT, що дозволяє сірі IP маскувати під білими IP.

Дані технології можливо використати вручну налагоджуючи кожен свіч та маршрутизатор, але краще їх реалізовувати через графічний інтерфейс генерації налаштувань, які потім можна використати безпосередньо на емуляторі мережевих систем, або на реальному обладнанні компанії Cisco.

1 ОГЛЯД ТЕХНОЛОГІЙ ТА ПРОТОКОЛІВ

1.1 Протокол DHCP

DHCP (Dynamic Host Configuration Protocol) – це протокол мережі, який використовується для автоматичного налаштування параметрів для пристроїв, які є частиною цієї мережі, такі як IP-адреса, маска, шлюз за замовчуванням та DNS-сервер [1].

DHCP дозволяє адміністраторам мережі забезпечити автоматичне налаштування мережевих параметрів для великої кількості пристроїв, спрощуючи їх налаштування як у питанні безпеки, так і у швидкості, бо протокол дозволяє пристроям автоматично отримувати мережеві параметри при підключенні до мережі і забезпечує ефективне використання доступних IP-адрес в мережі.

DHCP протокол мережевого управління базується на основі IP (Internet Protocol), протокол, що об’єднує сегменти мережі у єдину мережу, і завдяки якому автоматично назначаються IP-адреси та інші комунікативні параметри до пристроїв, приєднаних до мережі, яка використовує архітектуру “клієнт-сервер” (рис. 1.1).

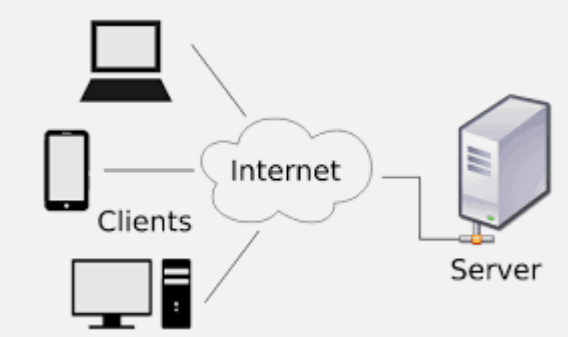

Рисунок 1.1 – архітектура “клієнт-сервер”

Технологія DHCP звільняє від потреби індивідуального конфігурування мережі для кожного пристрою вручну та складається з двох мережевих компонентів, центрального встановленого мережевого DHCP-сервера та

клієнтів-екземплярів стека на кожному компоненті мережі. Клієнт самостійно запитує набір параметрів сервера за допомогою DHCP під час підключення до мережі та періодично після приєднання до неї.

Протокол DHCP може реалізовуватись у різних мережах різних розмірів, починаючи від домашнього використання, закінчуючи компаніями-гігантами та регіональних мереж ISP (Internet service provider). Технологія DHCP входить до можливостей маршрутизаторів та домашніх шлюзів. Маршрутизатори домашніх мереж отримують унікальну IP-адресу в мережі ISP, а локальний DHCP-сервер призначає локальну, або сіру IP-адресу кожному пристрою мережі.

DHCP технологія працює з протоколом IP як версії 4 (IPv4), так і з 6 (IPv6), що має свою назву DHCP – DHCPv6 [2].

Поява протоколу DHCP була вперше у жовтні 1993 року, а до нього першим був RARP (Reverse Address Resolution Protocol), що був впроваджений у 1984 році для конфігурування нескладних пристроїв, наприклад, бездротові робочі станції з IP-адресою. Даний протокол працював на рівні каналу даних, що ускладнювало впровадження на багатьох серверних платформах. Дана технологія вимагала присутність серверу на кожному мережевому каналі. Після RARP був впроваджений протокол Bootstrap (BOOTP) у вересні 1985 року, який дозволяє автоматично отримувати IP-адреси пристроями, як і DHCP. DHCP – надбудова BOOTP, що на відміну від другого дозволяла динамічно визначати IP-адреси з пулу виділяти їх при відсутності в них потреби. Він також використовується для відправлення широкого діапазону додаткових параметрів конфігурації до клієнтів та спеціальні параметри специфічних платформ [3].

З 1997 був введений тип DHCPINFORM, який залишається основою для IPv4 мереж.

Як і IP протокол, DHCP працює на базі клієнт-серверної архітектури. Коли комп’ютер, або інший мережевий пристрій підключається до мережі, програмне забезпечення клієнта DHCP відсилає широкомовний трафік DHCP із запитом важливої інформації. Будь-який DHCP сервер в мережі може обробити даний

запит. DHCP-сервер управляє пулом IP-адресів та інформацією про параметри конфігурації клієнта, наприклад, default gateway (шлюз по-замовчуванню), ім’я домену, ім’я серверів та сервери часу. Після того, як було отримано DHCP-запит, DHCP-сервер може відповісти конкретною інформацією кожному клієнту, завчасно налаштованою адміністратором, або з конкретною адресою та іншою різною інформацією, що дійсна для всієї мережі та часу для якого оренда є дійсною. Зазвичай, DHCP-клієнт запитує цю інформацію одразу після завантаження та інколи після цього до закінчення терміну працездатності інформації. Коли DHCP-клієнт оновлює призначення, спочатку воно створює запит на ті самі параметри, але DHCP-сервер може призначити нову адресу на основі політики призначення, що встановлена адміністратором.

У великих багатоканальних мереж, один DHCP-сервер може обслуговувати всю мережу агентами ретрансляції DHCP, що розташовані на з’єднаних маршрутизаторах. Подібні агенти передають повідомлення між DHCP-клієнтами та DHCP-серверами, що розташовані в різних підмережах.

В залежності від реалізації, DHCP-сервер має три методи розподілу IP-адрес [2]:

- Динамічний розподіл:

  Адміністратор мережі резервує IP-адреси для DHCP, і кожен DHCP-клієнт у локальній мережі налаштований на запит IP-адреси від DHCP-сервера під час ініціалізації мережі. Процес запиту та надання використовує концепцію оренди з підконтрольним часовим періодом, що дозволяє DHCP-серверу відновлювати і потім перерозподіляти IP-адреси, які не оновлюються.

- Автоматичний розподіл:

  DHCP-сервер назавжди призначає IP-адреси клієнту, що створив запит з діапазону, заданим адміністратором. Це схоже динамічний розподіл, але DHCP-сервер тримає таблицю попередніх призначень IP-

адрес, щоб він міг переважно призначити клієнту той самий IP-адрес, який клієнт до цього мав.

- Ручний розподіл:

Даний метод також інколи називають статичним DHCP розподілом, розподілом фіксованої адреси, резервуванням та зв'язуванням MAC/IP-адрес. Адміністратор відбирає унікальний ідентифікатор (client-id або MAC-адрес) для кожного клієнта з IP-адресою, яка пропонується для клієнта-запитувача. DHCP-сервери можуть бути налаштовані на повернення до інших методів, якщо даний не вдається.

DHCP-служби використовуються для IPv4 та IPv6. Інтернет протоколи версій 4 та 6 дуже відрізняються, їх можна вважати різними протоколами. Наприклад, для IPv6 операції, пристрої можуть альтернативно використовувати автоматизовану конфігурацію адрес без використання збереження стану. IPv6 хости також можуть використовувати шлюз локальної адресації для виконання операцій, що обмежені локальною мережею.

DHCP використовує модель обслуговування без з'єднання, разом з UDP (User datagram protocol). Він реалізований з двома номерами портів UDP для його операцій, які є ті самі, що і для BOOTP. Сервер прослуховує число порту UDP 67, а клієнт – порт UDP номер 68.

DHCP операції поділяються на 4 етапи: виявлення сервера, пропозиція оренди IP-адреси, запит на оренду IP-адреси та підтвердження оренди IP-адреси (рисунок 1.2). Ці етапи зазвичай називають абревіатурою DORA (Discovery, Offer, Request, Acknowledgement) [4].

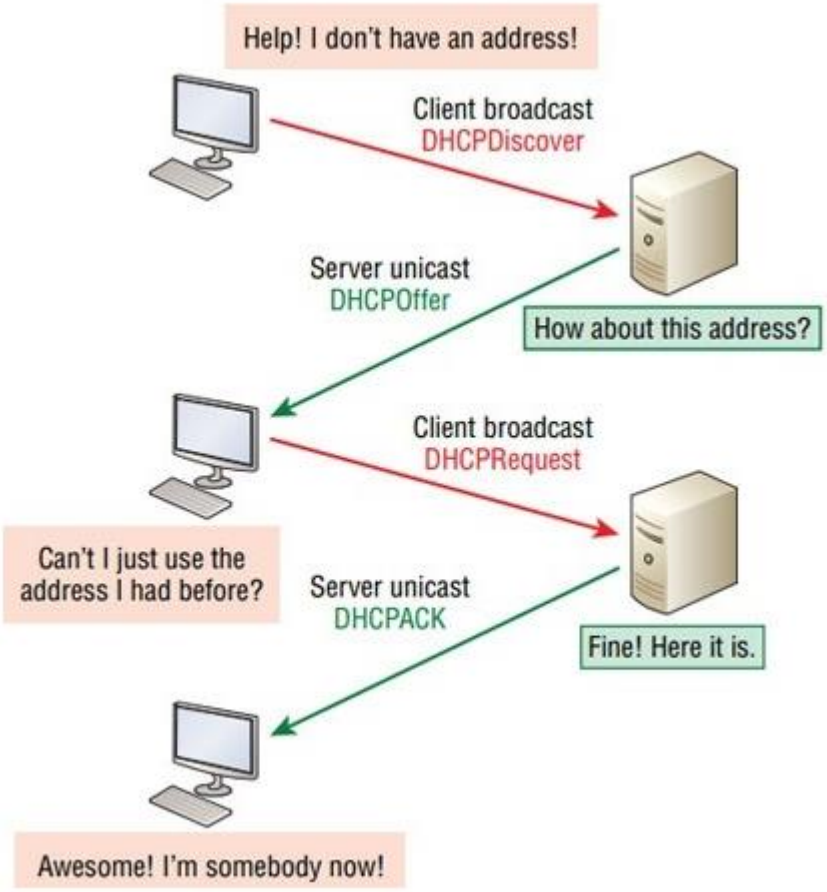

Рисунок 1.2 – етапи DHCP-операцій

DHCP-операція починається з клієнтської трансляції запиту. Якщо клієнт та сервер знаходяться в різних широкомовних доменах, можна використати DHCP Helper, або DHCP Relay Agent. Клієнти, які створюють запит на поновлення існуючої оренди, можуть спілкуватися прямо через UDP unicast, оскільки клієнт вже має встановлений IP-адрес на той момент. Додатково, є BROADCAST flag (1 біт в 2-байтовому полі прапорів, де всі інші біти зарезервовані, тому встановлюється 0), який клієнт може використати, щоб вказати, яким способом (широкомовним, або одноадресним) він може отримати DHCPOFFER: 0:8000 для широкомовної, 0:0000 для одноадресної передачі. Зазвичай, DHCPOFFER надсилається через одноадресну передачу. Для тих хостів, що не можуть прийняти одноадресні пакети до того, як IP-адреси налаштовані, цей прапорець можна використати для роботи з цією проблемою.

1.2 Технологія VLAN

VLAN (Virtual Local Area Network) – віртуальна локальна мережа, будь-який широкомовний домен, який розділений на ізольований у комп'ютерній мережі на канальному рівні (а саме другий рівень OSI) (рис. 1.3). У цьому контексті віртуальний відноситься до фізичного об'єкту, відтвореного і зміненого додатковою логікою в локальній мережі. VLAN працюють шляхом застосування тегів у мережевих системах, створюючи зовнішній вигляд і функціональність трафіку, який фізично знаходиться в одній мережі, але діє так, ніби він розділений між окремими мережами. Таким чином, VLAN можуть тримати мережеві програми відокремленими, незважаючи на те, що вони підключені до однієї фізичної мережі, і не вимагаючи розгортання кількох комплектів кабелів і мережевих пристроїв [5].

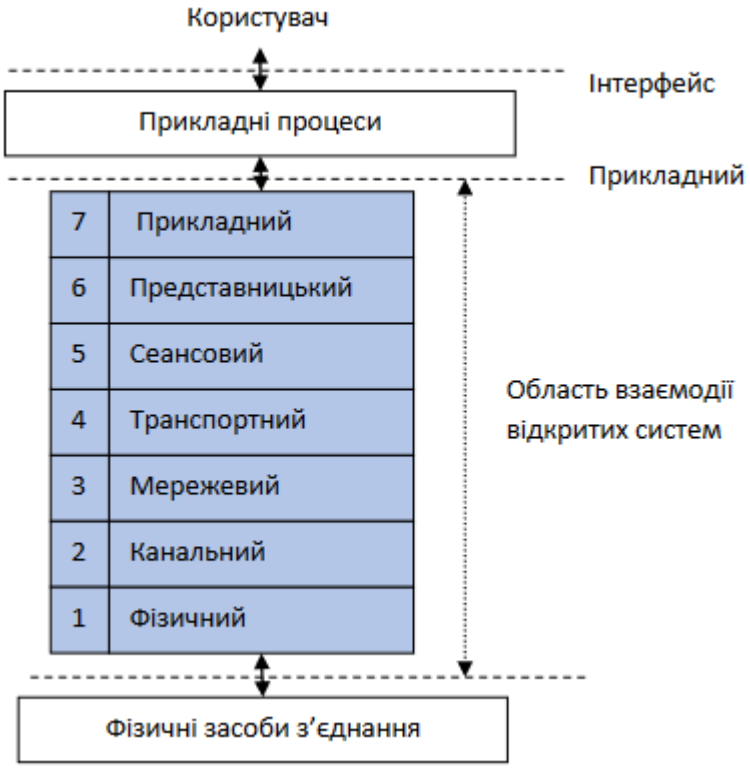

Рисунок 1.3 – рівні моделі OSI

VLAN дозволяють адміністраторам мережі групувати хости разом, навіть якщо хости не підключені напряму до одного свіча. Оскільки членство у VLAN можна налаштувати за допомогою програмного забезпечення, це може значно спростити проектування та розгортання мережі. Без VLAN групування хостів відповідно до їхніх ресурсів потребує праці з переміщення вузлів, або заміни каналів передачі даних. Мережі VLAN дозволяють пристроям, які повинні знаходитись окремо, спільно використовувати кабелі фізичної мережі, але при цьому вони не можуть безпосередньо взаємодіяти один з одним. Такий керований обмін дає переваги в простоті, безпеці, управлінні трафіком і економії. Наприклад, VLAN можна використовувати для розділення трафіку всередині підприємства на основі окремих користувачів, або груп користувачів, або їхніх ролей (наприклад, мережевих адміністраторів), або на основі характеристик трафіку (наприклад, трафік з низьким пріоритетом, якому запобігає вплив на решту функціонування мережі). Багато служб Інтернет-хостингу використовують VLAN для відокремлення приватних зон клієнтів одна від одного, що дозволяє групувати сервери кожного клієнта в одному сегменті мережі незалежно від того, де окремі сервери розташовані в центрі обробки даних. Потрібні певні запобіжні заходи, щоб запобігти “виходу” трафіку з даної VLAN, експлойт, відомий як VLAN Hopping.

Щоб розділити на VLAN, необхідно налаштувати мережеве обладнання. Більш просте обладнання може розділяти лише кожен фізичний порт, і в цьому випадку кожна VLAN працює через виділений мережевий кабель. Більш складні пристрої можуть позначати кадри за допомогою тегів VLAN, так що одне з’єднання (транк) може використовуватися для передачі даних для кількох VLAN. Оскільки VLAN спільно використовують пропускну здатність, транк VLAN може використовувати агрегацію каналів, пріоритет якості обслуговування, або обидва для ефективної маршрутизації даних.

VLAN вирішують такі проблеми, як масштабованість, безпека та керування мережею. Мережні архітектори налаштовують VLAN для

забезпечення сегментації мережі. Маршрутизатори між мережами VLAN фільтрують широкомовний трафік, підвищують безпеку мережі, виконують узагальнення адрес і зменшують перевантаження мережі.

У мережі, яка використовує широкомовні передачі для виявлення послуг, призначення та вирішення адрес та інших послуг, у міру збільшення кількості однорангових користувачів у мережі частота широкомовних розсилок також збільшується. Мережі VLAN можуть допомогти керувати широкомовним трафіком шляхом формування кількох широкомовних доменів. Розбиття великої мережі на меші незалежні сегменти зменшує обсяг широкомовного трафіку, який повинен нести кожен мережевий пристрій і сегмент мережі. Комутатори можуть не перемикати мережевий трафік між VLAN, оскільки це порушить цілісність широкомовного домену VLAN.

VLAN також можуть допомогти створити кілька мереж третього рівня в одній інфраструктурі. VLAN – це структура канального рівня (2-рівень OSI), аналогічні підмережам IP (Інтернет-протоколу), який є конструкціями мережевого рівня (3 рівня OSI). У середовищі, що використовує VLAN, між VLAN та IP-підмережами часто існує зв’язок “один-до-одного”, хоча в одній VLAN можна мати кілька підмереж.

Без можливості використовувати VLAN, користувачі призначаються до мереж на основі географії та обмежені фізичними топологіями та відстанями. VLAN можуть логічно групувати мережі, щоб відокремити мережеве розташування користувачів від їх фізичного розташування. Використовуючи VLAN, можна контролювати шаблони трафіку та швидко реагувати на переміщення співробітників, або обладнання. Мережі VLAN забезпечують гнучкість адаптації до змін у вимогах мережі та дозволяють спрощувати адміністрування.

VLAN можна використовувати для поділу локальної мережі на кілька окремих сегментів, наприклад:

- Виробництво;

- VoIP;
- Управління мережею;
- Мережа зберігання даних (SAN);
- Гостьовий доступ до Інтернету;
- Демілітаризована зона (DMZ).

Загальна інфраструктура, яка спільно використовується між магістралями VLAN, може забезпечити певний рівень безпеки з великою гнучкістю за порівняно низьку вартість. Схеми якості обслуговування можуть оптимізувати трафік на магістральних лініях для вимог у мережі реального часу (наприклад, VoIP), або з низькою затримкою (наприклад, SAN). Однак VLAN як рішення безпеки слід впроваджувати з великою обережністю, оскільки вони можуть бути зруйновані, якщо не впроваджувати їх ретельно.

У хмарних обчисленнях VLAN IP-адреси та MAC-адреси в хмарі є ресурсами, якими можуть керувати кінцеві користувачі. Щоб допомогти зменшити проблеми безпеки, розміщення хмарних віртуальних машин у VLAN може бути кращим, ніж розміщення їх безпосередньо в Інтернеті.

Мережеві технології з можливостями VLAN включають:

- Режим асинхронної передачі (ATM);
- Оптоволоконний інтерфейс розподілених даних (FDDI);
- Ethernet;
- HiperSockets;
- InfiniBand.

Після успішних експериментів з голосом через Ethernet з 1981 по 1984 рік У. Девід Сінкоскі приєднався до Bellcore і почав вирішувати проблему розташування мереж Ethernet. При швидкості 10 Мбіт/с Ethernet був швидшим за більшість альтернатив на той час. Однак Ethernet був широкомовною мережею, і не було гарного способу з’єднати декілька мереж Ethernet разом. Це обмежило загальну пропускну здатність мереж Ethernet до 10 Мбіт/с і максимальну відстань між вузлами до кількох сотень футів.

Напроти, незважаючи на те, що швидкість існуючої телефонної мережі для окремих з'єднань була обмежена 56 кбіт/с (менше однієї сотої швидкості Ethernet), загальна пропускна здатність цієї мережі була оцінена в 1 Тбіт/с (у сто тисяч разів більше, ніж Ethernet).

Хоча можна було використовувати IP-маршрутизацію для з'єднання кількох мереж Ethernet, це було дорого та відносно повільно. Сінкоскі почав шукати альтернативи, які вимагали менше обробки на пакет. У процесі він самостійно винайшов прозорі мости, технологію, яка використовується в сучасних комутаторах Ethernet. Однак використання комутаторів для з'єднання кількох мереж Ethernet у відмовостійкий спосіб потребує надлишкових шляхів через цю мережу, що, у свою чергу, вимагає конфігурації охоплюючого дерева. Це гарантує наявність лише одного активного шляху від будь-якого вихідного вузла до будь-якого пункту призначення в мережі. Це призводить до того, що комутатори, розташовані в центрі, стають вузькими місцями, обмежуючи масштабованість, оскільки все більше мереж з'єднані між собою.

Щоб полегшити цю проблему, Сінкоскі винайшов VLAN, додавши тег до кожного кадру Ethernet. Ці теги можна розглядати як кольори, скажімо, червоний, зелений або синій. У цій схемі кожному перемикачу можна призначити обробку кадрів одного кольору та ігнорувати решту. Мережі можуть бути з'єднані між собою за допомогою трьох остовних дерев, по одному для кожного кольору. Надсилаючи комбінацію різних кольорів рамки, можна покращити сукупну пропускну здатність. Сінкоскі назвав це мультидеревним мостом. Він і Чейз Коттон створили та вдосконалили алгоритми, необхідні для того, щоб зробити систему здійсненною. Цей колір тепер відомий у кадрі Ethernet як заголовок IEEE 802.1Q або тег VLAN. Хоча VLAN зазвичай використовуються в сучасних мережах Ethernet, вони не використовуються у спосіб, який спочатку передбачено тут.

У 1998 році Ethernet VLAN були описані в першому виданні стандарту IEEE 802.1Q-1998. Воно було розширено за допомогою IEEE 802.1ad, щоб

дозволити використання вкладених тегів VLAN у службі мосту провайдера. Цей механізм було вдосконалено за допомогою IEEE 802.1ah-2008.

Протокол, який на сьогодні найчастіше використовується для підтримки VLAN – це IEEE 802.1Q [6]. Робоча група IEEE 802.1 визначила цей метод мультиплексування VLAN, намагаючись забезпечити підтримку VLAN від багатьох постачальників. До введення стандарту 802.1Q існувало кілька власних протоколів, таких як Cisco Inter-Switch Link (ISL) і 3Com`s Virtual LAN Trunk (VLT). Cisco також реалізувала VLAN через FDDI, переносячи інформацію VLAN у заголовок кадру IEEE 802.10, що суперечить меті стандарту IEEE 802.10.

І ISL, і IEEE 802.1Q виконують явне тегування – сам кадр позначається ідентифікаторами VLAN. ISL використовує зовнішній процес тегування, який не змінює фрейм Ethernet, у той час як 802.1Q використовує внутрішнє поле фрейму для тегування, і тому змінює основну структуру фрейму Ethernet. Це внутрішнє тегування дозволяє IEEE 802.1Q працювати як на каналах доступу, так і на магістральних каналах за допомогою стандартного обладнання Ethernet.

IEEE 802.1Q

Відповідно до IEEE 802.1Q максимальна кількість VLAN у певній мережі Ethernet становить 4094 (4094 значень, наданих 12-бітним полем VID мінус зарезервовані значення на кожному кінці діапазону, 0 і 4095). Це не накладає однакове обмеження на кількість IP-підмереж у такій мережі, оскільки одна VLAN може містити кілька IP-підмереж. IEEE802.1ad розширює кількість підтримуваних мереж VLAN, додаючи підтримку кількох вкладених тегів VLAN. IEEE 802.1aq (Shortest Path Bridging) розширює ліміт VLAN до 16 мільйонів. Обидва вдосконалення входять до стандарту IEEE 802.1Q.

Cisco Inter-Switch Link

Inter-Switch Link (ISL) – це власний протокол Cisco, який використовується для з’єднання комутаторів і підтримки інформації про VLAN під час проходження трафіку між комутаторами по магістральних каналах. ISL

надається як альтернатива IEEE 802.1Q. ISL доступний лише на певному обладнанні Cisco і вже не підтримується.

Транковий протокол Cisco VLAN

VLAN Trunking Protocol (VTP) – це власний протокол Cisco, який поширює визначення VLAN у всій локальній мережі. VTP доступний у більшості продуктів сімейства Cisco Catalyst Подібним стандартом IEEE, який використовують інші виробники, є GARP VLAN Registration Protocol (GVRP), або новіший Multiple VLAN Registration Protocol (MVRP).

Протокол реєстрації декількох VLAN

Multiple VLAN Registration Protocol – це програма Multiple Registration Protocol, яка дозволяє автоматично налаштовувати інформацію VLAN на мережевих комутаторах. Зокрема, він надає спосіб динамічного обміну інформацією VLAN і налаштування необхідних мереж VLAN.

1.3 Технологія NAT

Network address translation (NAT) – це метод відображення одного простору IP-адрес в інший шляхом заміни інформації про мережеву адресу в IP-заголовку пакетів, коли вони передаються через пристрій маршрутизації трафіку. Спочатку ця техніка використовувалася для обходу необхідності призначати нову адресу кожному хосту під час переміщення мережі, або заміни постачальника послуг Інтернету, але не могла маршрутизувати адресний простір мережі. Він став популярним і важливим інструментом для збереження глобального адресного простору в умовах виснаження адрес IPv4. Одна IP-адреса шлюзу NAT, яку можна маршрутизувати в Інтернеті, може використовуватися для всієї приватної мережі (рис. 1.4) [7].

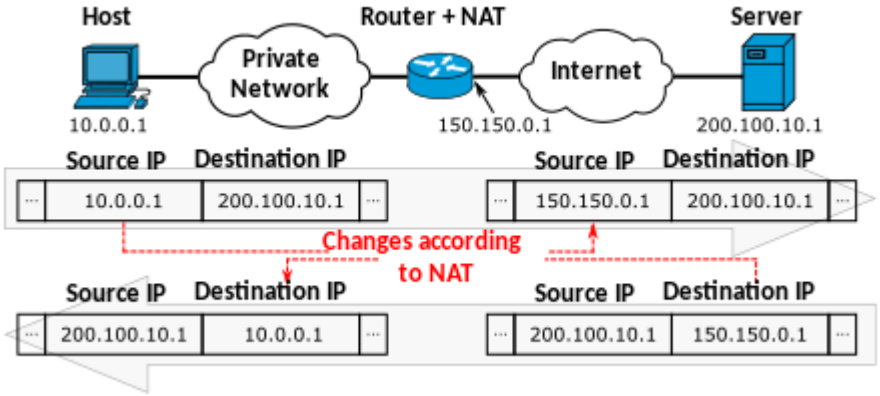

Рисунок 1.4 – NAT між локальною та глобальною мережами

Оскільки трансляція мережевої адреси змінює інформацію IP-адреси в пакетах, реалізація NAT може відрізнятися за своєю конкретною поведінкою в різних випадках адресації та своїм впливом на мережевий трафік. Специфіка поведінки NAT зазвичай не документується постачальниками обладнання, що містить реалізації NAT.

Більша частина трансляторів мережевих адрес відображають кілька приватних хостів на одну загальнодоступну IP-адресу (рис. 1.5).

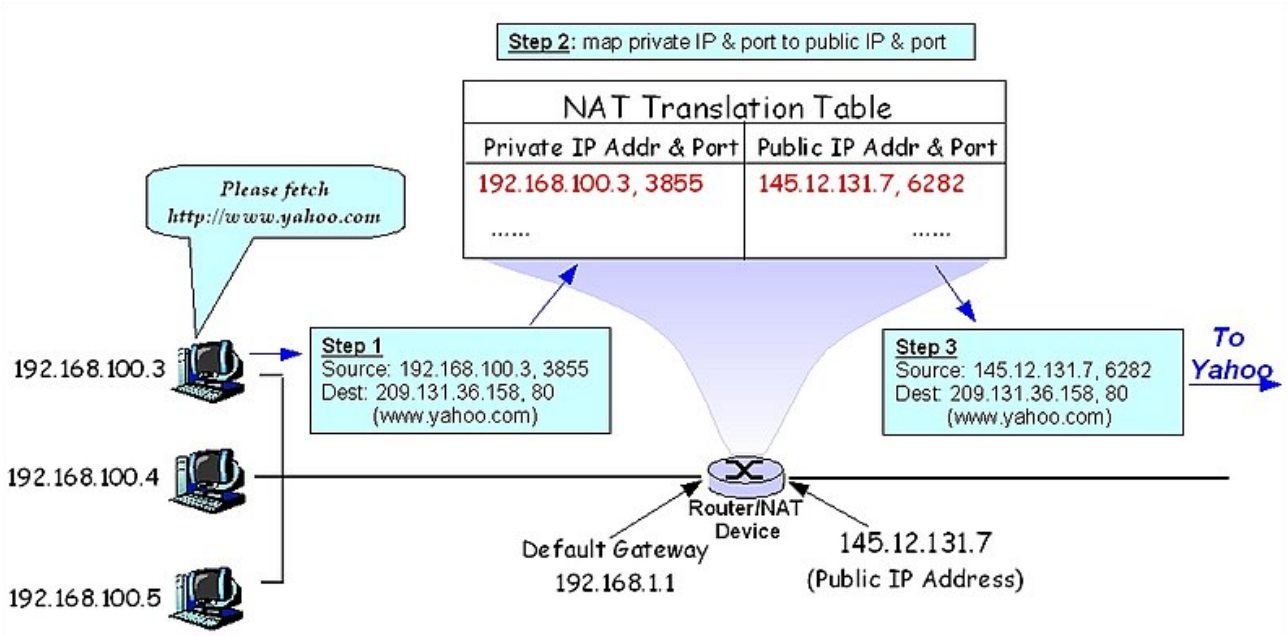

Рисунок 1.5 – відображення мережевих адрес

Типова конфігурація:

1. Локальна мережа використовує одну з призначених приватних підмереж IP-адрес

2. У мережі є маршрутизатор, який має як приватну, так і публічну адресу. Приватна адреса використовується маршрутизатором для зв'язку з іншими пристроями в приватній локальній мережі. Загальнодоступна адреса (зазвичай призначається постачальником послуг Інтернету) використовується маршрутизатором для зв'язку з рештою Інтернету.

3. Коли трафік проходить з мережі в Інтернет, маршрутизатор перетворює адресу джерела в кожному пакеті з приватної адреси на власну публічну адресу маршрутизатора. Маршрутизатор відстежує основні дані про кожне активне з'єднання (зокрема адресу та порт призначення). Коли маршрутизатор отримує вхідний трафік з Інтернету, він використовує дані відстеження з'єднання, збережені під час вихідної фази, щоб визначити, на яку приватну адресу (якщо така є) йому слід переслати відповідь.

Усі IP-пакети мають IP-адресу джерела та IP-адресу призначення. Як правило, адреса джерела пакетів, що переходять із приватної мережі до загальнодоступної, буде змінена, тоді як адреса призначення пакетів, що переходять із загальнодоступної мережі назад у приватну, буде змінена. Щоб уникнути неоднозначності в тому, як перекладаються відповіді, потрібні подальші модифікації пакетів. Переважна частина Інтернет-трафіку використовує протокол TCP, або UDP. Для цих протоколів номери портів змінюються таким чином, щоб комбінація IP-адреси (у заголовку IP) і номера порту (у заголовку транспортного рівня) у повернутому пакеті могла бути однозначно зіставлена з відповідним одержувачем приватної мережі. RFC 2663 використовує термін трансляція мережевих адрес і портів (NAPT) для цього типу NAT. Інші назви включають трансляцію адреси порту (PAT), IP-маскування, перевантаження NAT і багато-до-одного NAT. Це найпоширеніший тип NAT, який став синонімом терміну "NAT" у загальному вживанні.

Цей метод дозволяє спілкуватися через маршрутизатор лише тоді, коли розмова походить у приватній мережі, оскільки початкова вихідна передача є тим, що встановлює необхідну інформацію в таблицях перекладу. Таким чином, веб-браузер у приватній мережі зможе переглядати веб-сайти, які знаходяться за межами мережі, тоді як веб-браузери за межами мережі не зможуть переглядати веб-сайт, розміщений у ній. Протоколи, що не базуються на TCP та UDP, вимагають інших методів перекладу.

Додатковою перевагою NAT типу “один-до-багатьох” є те, що він зменшує виснаження адреси IPv4, дозволяючи цілій мережі підключатися до Інтернету за допомогою однієї загальнодоступної IP-адреси.

Трансляцію мережевих адрес і портів можна реалізувати кількома способами. Деяким програмам, які використовують інформацію про IP-адресу, може знадобитися визначити зовнішню адресу транслятора мережевих адрес. Це адреса, яку виявляють його однорангові зв’язки у зовнішній мережі. Крім того, може знадобитися вивчити та класифікувати тип відображення, що використовується, наприклад, коли потрібно налаштувати прямий шлях зв’язку між двома клієнтами, обидва з яких знаходяться за окремими шлюзами NAT.

З цією метою в 2003 році RFC 3489 визначив протокол від назвою Simple Traversal of UDP over NAT (STUN). Він класифікував реалізації NAT як повний конус NAT, обмежений портом конус NAT, або симетричний NAT, відповідно запропонував методологію тестування пристрою. Однак з тих пір ці процедури були вилучені зі статусу стандартів, оскільки методи не підходять для правильної оцінки багатьох пристроїв. RFC 5389 стандартизував нові методи в 2008 році, а акронім STUN тепер представляє нову назву специфікації: Session Traversal Utilities for NAT.

Класифікація реалізації NAT [8]:

1. Первинний конус NAT, також відомий як NAT один-до-одного (рис. 1.6). Коли внутрішня адреса (iAddr:iPort) зіставляється із зовнішньою адресою (eAddr:ePort), усі пакети з iAddr:iPort надсилаються через eAddr:eProt. Будь-який

зовнішній хост може надсилати пакети на iAddr:iPort, надсилаючи пакети на eAddr:ePort.

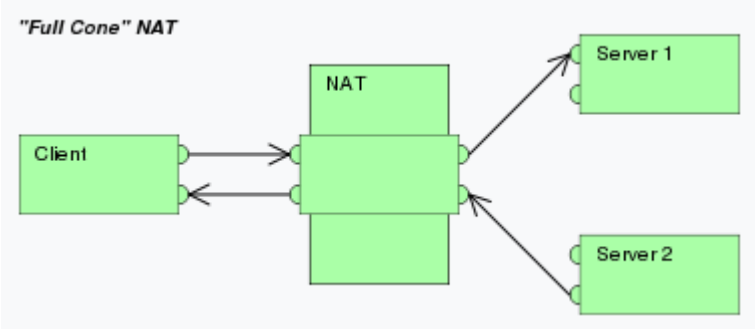

Рисунок 1.6 – первинний конус NAT

2. Конус NAT з обмеженою адресою (рис. 1.7). Коли внутрішня адреса (iAddr:iPort) зіставляється із зовнішньою адресою (eAddr:ePort), усі пакети з iAddr:iPort надсилаються через eAddr:ePort. Зовнішній хост (hAddr:any) може надсилати пакети на iAddr:iPort, надсилаючи пакети на eAddr:ePort, лише якщо iAddr:ePort раніше відсилав пакети на hAddr:any. “Any” означає, що номер порту не має значення.

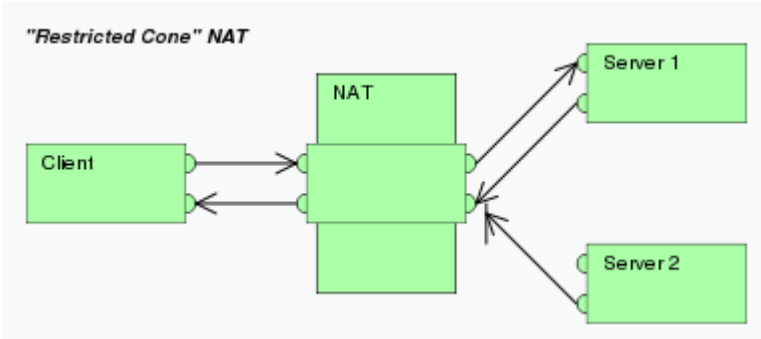

Рисунок 1.7 – конус NAT з обмеженою адресою

3. Конус NAT з обмеженням порту як і конус NAT з обмеженням адреси, але обмеження включає номери портів (рис. 1.8). Коли внутрішня адреса (iAddr:iPort) зіставляється із зовнішньою адресою (eAddr:ePort), усі пакети з iAddr:iPort надсилаються через eAddr:ePort. Зовнішній хост (hAddr:hPort) може

надсилати пакети на iAddr:iPort, надсилаючи пакети на eAddr:ePort, лише якщо iAddr:iPort раніше надіслав пакет на hAddr:hPort.

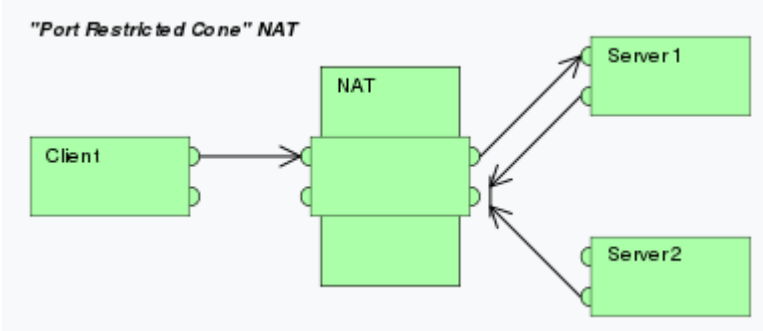

Рисунок 1.8 – конус NAT з обмеженими адресою та портом

4. Симетричний NAT (рис. 1.9). Комбінація однієї внутрішньої IP-адреси плюс IP-адреси призначення та порту відображається на одну унікальну зовнішню IP-адресу та порт джерела; якщо той самий внутрішній хост надсилає пакет навіть їх тією самою адресою джерела та портом, але іншому адресату, використовується інше відображення. Лише зовнішній хост, який отримує пакет від внутрішнього хоста, може надіслати пакет назад.

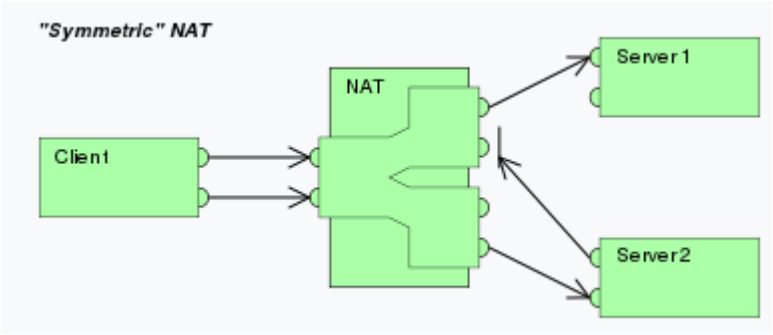

Рисунок 1.9 – симетричний NAT

Багато реалізацій NAT поєднують ці типи, тому краще посилатися на конкретну окрему поведінку NAT замість використання термінології “конус/симетрія”. RFC 4787 намагається зменшити плутанину, вводячи стандартизовану термінологію для спостережуваної поведінки. Для першого

пункту в кожному рядку наведеної вище таблиці RFC характеризуватиме повний конус, обмежений конус і порт-обмежений конус NAT як такі, що мають незалежне від кінцевої точки відображення, тоді як симетричний NAT характеризуватиме як такий, що має адресну та портову залежність відображення. Для другого пункту в кожному рядку наведеної вище таблиці RFC 4787 також позначає первинний конус NAT як такий, що має незалежну від кінцевої точки фільтрацію, конус NAT обмежений адресою як такий, що має адресно залежну фільтрацію, порт-залежний конус NAT як такий, що має адресу і фільтрацію, залежну від порту, та симетричний NAT, що має адрес-залежну фільтрацію, або адрес-залежну та порт-залежну фільтрацію. Інші класифікації поведінки NAT, згадані в RFC, включають те, чи зберігають вони порти, коли та як оновлюють відображення, чи можуть зовнішні відображення використовуватися внутрішніми хостами (тобто його зачіпна поведінка), а також рівень детермінізму, який NAT демонструють від час застосування всіх цих правил. Зокрема, більшість NAT поєднують симетричний NAT для вихідних з’єднань зі статичним відображенням портів, де вхідні пакети, адресовані зовнішній зовнішній адресі та порту, перенаправляються на певну внутрішню адресу та порт.

1.4 Постановка задачі

На базі зібраної та проаналізованої інформації можна визначити етапи виконання практичної частини кваліфікаційної роботи.

1. Створити модель офісної мережі на основі емулятора Cisco Packet Tracer з використанням технології VLAN та протоколів DHCP та NAT.
2. Створити програмний додаток з графічним інтерфейсом для автоматичної генерації конфігураційних даних. Дана програма допоможе у подальшому налаштовувати офісну мережу без спеціальних знань ручного налаштування мережевих пристроїв компанії Cisco. Даний інтерфейс має бути легким і зрозумілим у використанні.

3. Використати графічне програмне забезпечення на практиці, ввівши згенеровані дані у мережеві пристрої емулятора Cisco Packet Tracer.

2 СТВОРЕННЯ МОДЕЛІ ЛОКАЛЬНОЇ МЕРЕЖІ З ПІДТРИМКОЮ ТЕХНОЛОГІЙ VLAN ТА ПРОТОКОЛІВ DHCP ТА NAT

2.1 Створення локальної мережі в емуляторі Cisco Packet Tracer

Перед тим, як налаштовувати реальну локальну систему мережевого обладнання з певною логікою, можна використати емулятор конфігурування подібних мереж на базі різного обладнання: маршрутизаторів, комутаторів, різних видів кабелів, комп'ютерів тощо.

Завдяки програмі, що емулює роботу та логіку мережевого обладнання, можна завчасно визначити розміри проектованої системи, її функціональні можливості, питання з точки зору безпеки. Емулятор дозволяє створити декілька варіантів певної мережевої системи, які потім можна проаналізувати, виявити проблеми та переваги кожної, та обрати найкращій. Таких підхід дозволяє компаніям економити великі гроші на уникненні в створенні помилок, бо розміри систем бувають різними, і якщо виникає фатальна помилка у локальній мережі компанії-мільйонера – це фінансовий крах, який був непомічений при проектуванні мережі.

Cisco Packet Tracer – одна з таких програм-емуляторів фірми Cisco Systems [9]. Вона дозволяє створювати робочі моделі мереж, налагоджувати маршрутизатори та комутатори завдяки командам, які являються частиною програмного забезпечення Cisco IOS, мати взаємозв'язок між декількома користувачами через хмару.

Packet Tracer дозволяє також проектувати складні та дуже великі мережі, які за часту являються неможливими для того, щоб проектувати їх на реальному обладнанні через великі затрати. Packet Tracer являється безкоштовною програмою, яка у своєму основному орієнтирі використовується як програма, що дозволяє студентам вивчати основні концепції Сертифікації Cisco. Програма надає додаткові компоненти для вивчення, в тому числі авторську систему, моделювання мережевого протоколу, покращення знань та систему оцінювання. Packet Tracer можу бути корисним для розуміння абстрактних мережевих

концепцій, таких як Enhanced Internal Gateway Routing Protocol шляхом використання цих елементів у візуальній формі.

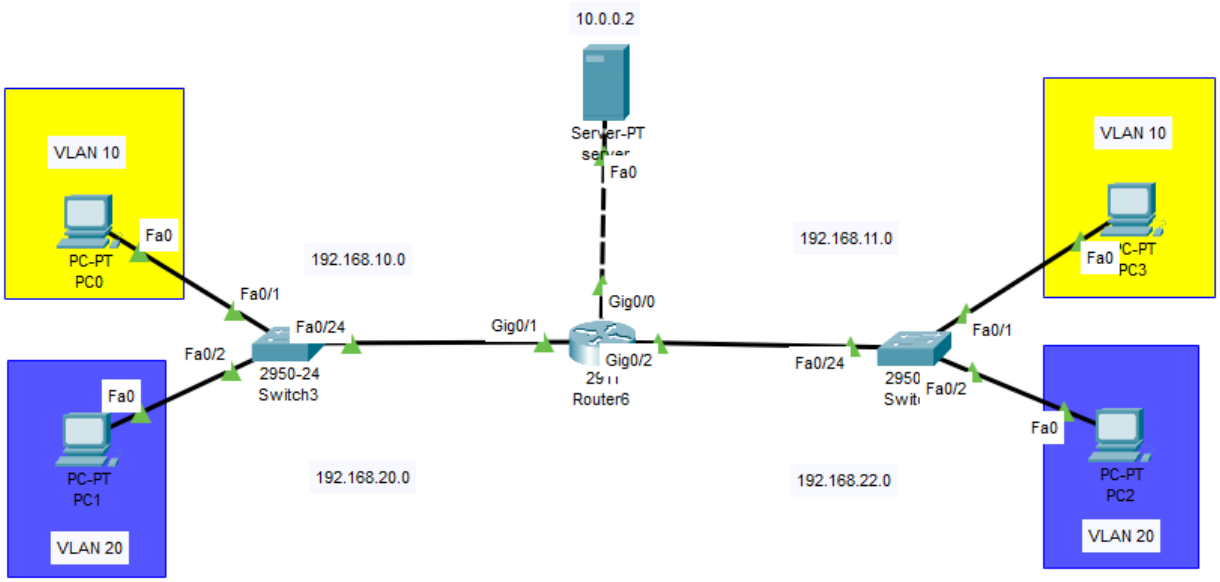

Рисунок 2.1 – локальна офісна мережа в емуляторі Cisco Packet Tracer

На базі емулятора Cisco Packet Tracer було побудовано локальну офісну мережу з використанням технологій VLAN, DHCP та NAT. За даною моделлю (рис. 2.1) буде створений графічний інтерфейс конфігурування.

Для побудови системи використано:

1) 2 комутатора;
2) 1 маршрутизатор;
3) 1 сервер;
4) 4 комп’ютера.

Налаштування обладнання:

Для роботи мережі необхідно задати певні параметри для правильного функціонування системи.

При конфігуруванні комутаторів треба пам’ятати, що система має у своєму розпорядженні технологію VLAN, на схемі зображені 2 VLAN(рис. 2.1), тому

конфігурація буде полягати у налаштуванні портів на режим “trunk” та “access” зі створенням самих VLAN.

Конфігурація комутаторів виглядатиме наступним чином:

Перший комутатор:

en

conf t – *вхід до режиму конфігурації.*

int fa 0/24 – *вхід до конфігурації інтерфейсу FastEthernet0/24.*

switchport mode trunk – *приведення інтефейсів FastEthernet0/24 у режим “trunk”.*

exit – *вихід с режиму налаштування інтерфейсу.*

int range fa 0/1-23 – *вхід до конфігурації списку інтерфейсів FastEthernet0/1-23.*

switchport mode access – *приведення списку інтерфейсів FastEthernet0/1-23 у режим “access”.*

exit – *вихід с режиму налаштування списку інтерфейсів.*

int fa 0/1 – *вхід до конфігурації інтерфейсу FastEthernet0/1.*

switchport access vlan 10 – *надання інтерфейсу FastEthernet0/1 властивості VLAN 10.*

int fa 0/2 – *вхід до конфігурації інтерфейсу FastEthernet0/2.*

switchport access vlan 20 – *надання інтерфейсу FastEthernet0/2 властивості VLAN 20.*

Другий комутатор:

en

conf t – *вхід до режиму конфігурації.*

int fa 0/24 – *вхід до конфігурації інтерфейсу FastEthernet0/24.*

switchport mode trunk – *приведення інтерфейсу FastEthernet0/24 у режим “trunk”.*

exit – *вихід с режиму налаштування інтерфейсу.*

int range fa 0/1-23 – *вхід до конфігурації списку інтерфейсів FastEthernet0/1-23.*

switchport mode access – *приведення списку інтерфейсів FastEthernet0/1-23 у режим "access".*

exit – *вихід с режиму налаштування списку інтерфейсів.*

int fa 0/1 – *вхід до конфігурації інтерфейсу FastEthernet0/1.*

switchport access vlan 10 – *надання інтерфейсу FastEthernet0/1 властивості VLAN 10.*

int fa 0/2 – *вхід до конфігурації інтерфейсу FastEthernet0/2.*

switchport access vlan 20 – *надання інтерфейсу FastEthernet0/2 властивості VLAN 20.*

Для конфігурації маршрутизатора треба використати декілька технологій одразу. По-перше, треба пори, що направлені на VLAN'и, розбити на їх кількість (на 2) і для кожного створити саб-порти для правильної комутації різних VLAN. По-друге, маршрутизатор має сам надавати параметри кожному клієнту мережі, тобто треба налаштувати DHC протокол. По-третє, має бути технологія NAT, яка дозволить клієнтам звертатись до серверу, який у свою чергу не буде мати можливості звернутися до клієнта.

Конфігурація маршрутизатора виглядатиме наступним чином:

en

conf t – *вхід до режиму конфігурації.*

int gig 0/0 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/0.*

ip add 10.0.0.1 255.0.0.0 – *вказання IP-адреси та маски IP-адреси обраного інтерфейсу.*

no sh – *увімкнення інтерфейсу роутера.*

int gig 0/1 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/1.*

no sh – *увімкнення інтерфейсу роутера.*

int gig 0/2 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/2.*

no sh – *увімкнення інтерфейсу роутера.*

int gig 0/1.10 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/1.10.*

encapsulation dot1Q 10 – *оголошення віртуальної локальної мережі з назвою 10 для саб-інтерфейсу gigabitEthernet0/1.10.*

ip add 192.168.10.1 255.255.255.0 – *вказання IP-адреси та маски IP-адреси саб-інтерфейсу.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

int gig 0/1.20 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/1.20.*

encapsulation dot1Q 20 – *оголошення віртуальної локальної мережі з назвою 20 для саб-інтерфейсу gigabitEthernet0/1.20.*

ip add 192.168.20.1 255.255.255.0 – *вказання IP-адреси та маски IP-адреси саб-інтерфейсу.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

exit – *вихід з режиму налаштування саб-інтерфейсу.*

int gig 0/2.10 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/2.10.*

encapsulation dot1Q 10 – *оголошення віртуальної локальної мережі з назвою 10 для саб-інтерфейсу gigabitEthernet0/2.10.*

ip add 192.168.11.1 255.255.255.0 – *вказання IP-адреси та маски IP-адреси саб-інтерфейсу.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

int gig 0/2.20 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/2.20.*

encapsulation dot1Q 20 – *оголошення віртуальної локальної мережі з назвою 20 для саб-інтерфейсу gigabitEthernet0/2.20.*

ip add 192.168.22.1 255.255.255.0 – *вказання IP-адреси та маски IP-адреси саб-інтерфейсу.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

exit – *вихід з режиму налаштування саб-інтерфейсу.*

ip dhcp pool poolleftone – *створення DHCP пулу з назвою poolleftone.*

network 192.168.10.0 255.255.255.0 – *налаштування DHCP пулу на можливість видачі пристроям вказаної IP-адреси та маски IP-адреси.*

default-router 192.168.10.1 – *вказання шлюзу, для якого створений DHCP пул.*

exit – вихід з режиму налаштування DHCP пулу.

ip dhcp pool poollefttwo – *створення DHCP пулу з назвою poollefttwo.*

network 192.168.20.0 255.255.255.0 – *налаштування DHCP пулу на можливість видачі пристроям вказаної IP-адреси та маски IP-адреси.*

default-router 192.168.20.1 – *вказання шлюзу, для якого створений DHCP пул.*

exit – вихід з режиму налаштування DHCP пулу.

ip dhcp pool poolrightone – *створення DHCP пулу з назвою poolrightone.*

network 192.168.11.0 255.255.255.0 – *налаштування DHCP пулу на можливість видачі пристроям вказаної IP-адреси та маски IP-адреси.*

default-router 192.168.11.1 – *вказання шлюзу, для якого створений DHCP пул.*

exit -– вихід з режиму налаштування DHCP пулу.

ip dhcp pool poolrighttwo – *створення DHCP пулу з назвою poolrighttwo.*

network 192.168.22.0 255.255.255.0 – *налаштування DHCP пулу на можливість видачі пристроям вказаної IP-адреси та маски IP-адреси.*

default-router 192.168.22.1 – *вказання шлюзу, для якого створений DHCP пул.*

exit – вихід з режиму налаштування DHCP пулу.

int gig 0/0 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/0.*

ip nat outside – *переведення інтерфейсу маршрутизатора до режиму "outside" для налаштування протоколу NAT із вказанням глобальної мережі.*

int gig 0/1 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/1.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

int gig 0/2 – *вхід до режиму конфігурації інтерфейсу gigabitEthernet0/2.*

ip nat inside – *переведення інтерфейсу до режиму "inside" для налаштування протоколу NAT.*

exit – *вихід з режиму налаштування інтерфейсу.*

ip access-list standard NAT – *створення списку доступу з назвою "NAT".*

permit 192.168.10.0 0.0.0.255 – *внесення мережі до списку з інвертованою маскою.*

permit 192.168.20.0 0.0.0.255 – *внесення мережі до списку з інвертованою маскою.*

permit 192.168.11.0 0.0.0.255 – *внесення мережі до списку з інвертованою маскою.*

permit 192.168.22.0 0.0.0.255 – *внесення мережі до списку з інвертованою маскою.*

exit – *вихід з режиму налаштування списку доступу.*

ip nat inside source list NAT interface gig 0/0 overload – *активація протоколу NAT зі списком доступу NAT на зовнішньому глобальному інтерфейсі.*

Налаштування серверу має бути приведено зі статичною IP-адресою (рис. 2.3)

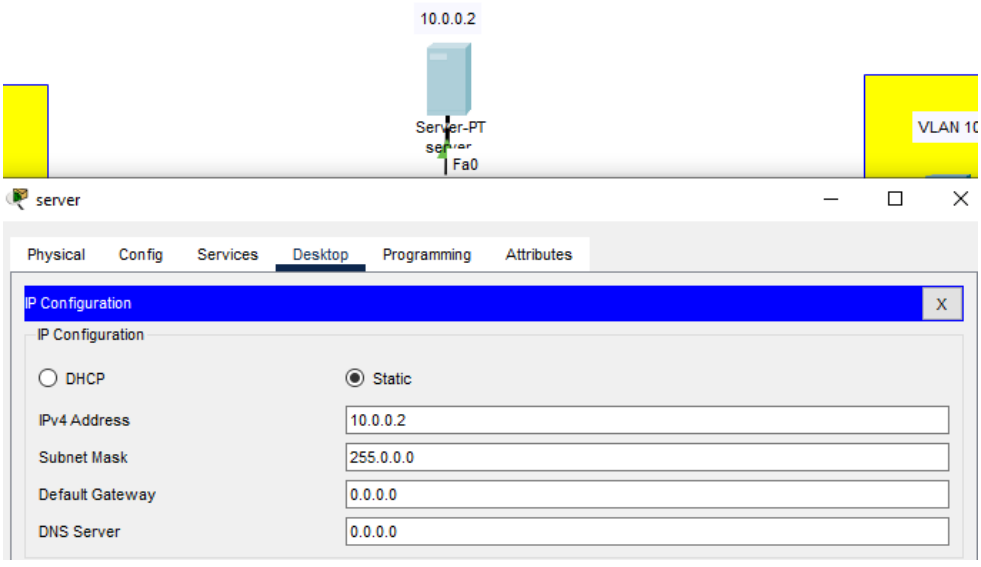

Рисунок 2.2 – налаштування серверу у програмі Packet Tracer

2.2 Створення графічного інтерфейсу генерації налаштування на базі мови програмування JavaScript

Для полегшення налаштування системи локальної мережі треба створити графічний інтерфейс автоматизованого конфігурування локальної мережі, який і являє собою головною частиною дипломного проектування. Для створення інтерфейсу використовується мова програмування JavaScript.

JavaScript – мова програмування, що підтримує різні стилі: об’єктно-орієнтовний, імперативний та функціональний. Дана мова використовується як приєднана мова програмного доступу до об’єктів додатків. Більш за все використовується у браузерах як мова сценарію, щоб надати більшу інтерактивність до веб-сторінок.

Основні риси даної мови програмування:

- динамічна типізація;
- слабка типізація;
- автоматичне керування пам’яттю;
- прототипне програмування.

JavaScript має повну інтеграцію з мовою розмітки HTML та CSS, тому дуже легко створити графічний інтерфейс саме у такій комбінації, що дасть змогу відобразити його на будь-якому браузері.

Для створення графічного інтерфейсу слід використовувати можливості даної мови: здатність проведення операцій із текстовими даними, можливість збереження даних у змінних, можливість завантажувати код на веб-сторінці браузера. Інтерпретатор браузера запускає скрипти JavaScript, таким чином JavaScript працює на стороні клієнта, так як всі операції проводяться на його стороні.

3 СТВОРЕННЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ ДЛЯ КОНФІГУРАЦІЇ ЛОКАЛЬНОЇ МЕРЕЖІ ОФІСУ ІТ-КОМПАНІЇ

Налаштування графічним інтерфейсом мережі буде на обох комутаторах та маршрутизаторі (рис. 3.1).

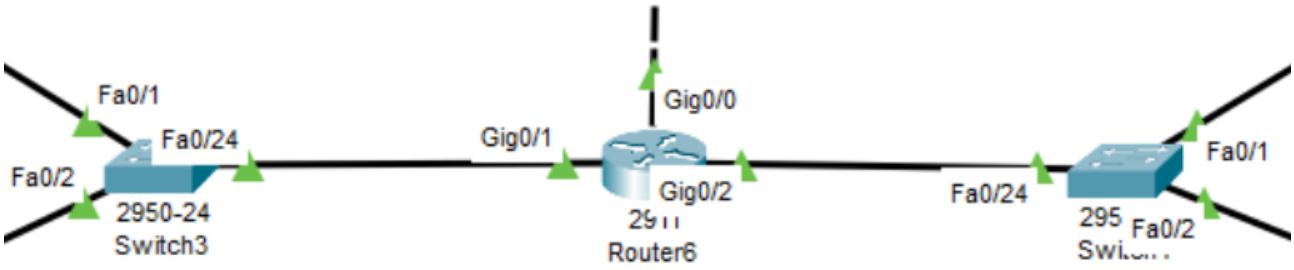

Рисунок 3.1 – налагоджувані прилади мережі

Конфігурація комутаторів буде схожою один до одного через однаковість підключення інтерфейсів до комп’ютерів та їх віртуальних локальних мереж.

Реалізація графічного інтерфейсу для налагодження локальної мережі знаходиться у додатку до кваліфікаційної роботи.

3.1 Огляд створеного графічного інтерфейсу

Створений графічний інтерфейс (рис. 3.2) відображає схему локальної мережі, яка була створена у програмі-емуляторі Cisco Packet Tracer. На схемі зображені інтерфейси маршрутизатора та комутаторів, що під’єднані до комп’ютерів та сервера, також відображені порядки розміщення VLAN мереж, що у сумі дає змогу інтуїтивно зрозуміти логістику роботи даної мережі. При заходженні на веб-сторінку, користувача зустрічає приємна анімація плавного появлення верхньої частини сторінки та полів введення даних, а також плавна поява назв та тексту.

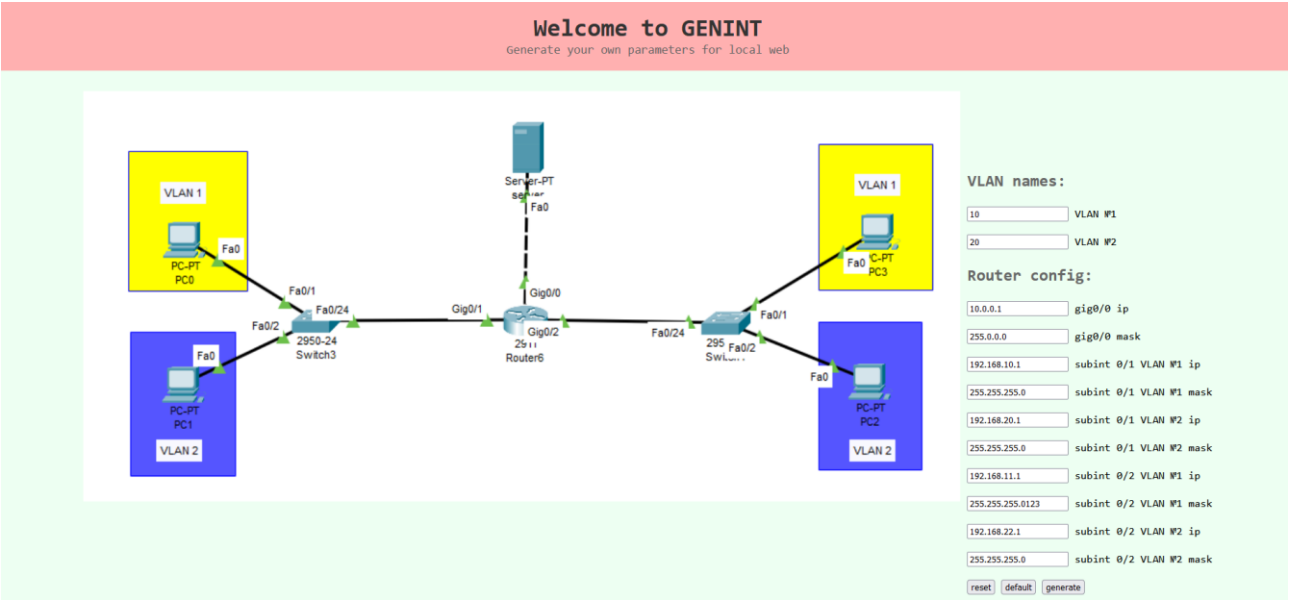

Рисунок 3.2 – графічний інтерфейс налаштування мережі

Поля введення (рис. 3.3) знаходяться у правій частині від зображення самої схеми для зручного прийняття рішень з конфігурації системи. Кожне поле має свої параметри, які потім використаються у створенні коду конфігурації кожного обладнання:

- VLAN №1 – поле введення назви віртуальної локальної мережі (даний параметр відповідає за налагодження комутатора);
- VLAN №2 – поле введення назви віртуальної локальної мережі (даний параметр відповідає за налагодження комутатора);
- gig0/0 ip – поле введення IP-адреси для інтерфейсу gig0/0, що підключений до серверної частини мережі;
- gig0/0 mask – поле введення маски IP-адреси для інтерфейсу gig0/0, що підключений до серверної частини мережі;
- subint 0/1 VLAN №1 ip – поле для введення поле введення IP-адреси для саб-інтерфейсу gig0/1.(VLAN №1);
- subint 0/1 VLAN №1 mask – поле для введення поле введення маски IP-адреси для саб-інтерфейсу gig0/1.(VLAN №1);

- subint 0/1 VLAN №2 ip – поле для введення поле введення IP-адреси для саб-інтерфейсу gig0/1.(VLAN №2);
- subint 0/1 VLAN №2 mask – поле для введення поле введення маски IP-адреси для саб-інтерфейсу gig0/1.(VLAN №2);
- subint 0/2 VLAN №1 ip – поле для введення поле введення IP-адреси для саб-інтерфейсу gig0/2.(VLAN №1);
- subint 0/2 VLAN №1 mask – поле для введення поле введення маски IP-адреси для саб-інтерфейсу gig0/2.(VLAN №1);
- subint 0/2 VLAN №2 ip – поле для введення поле введення IP-адреси для саб-інтерфейсу gig0/2.(VLAN №2);
- subint 0/2 VLAN №2 mask – поле для введення поле введення маски IP-адреси для саб-інтерфейсу gig0/2.(VLAN №2);

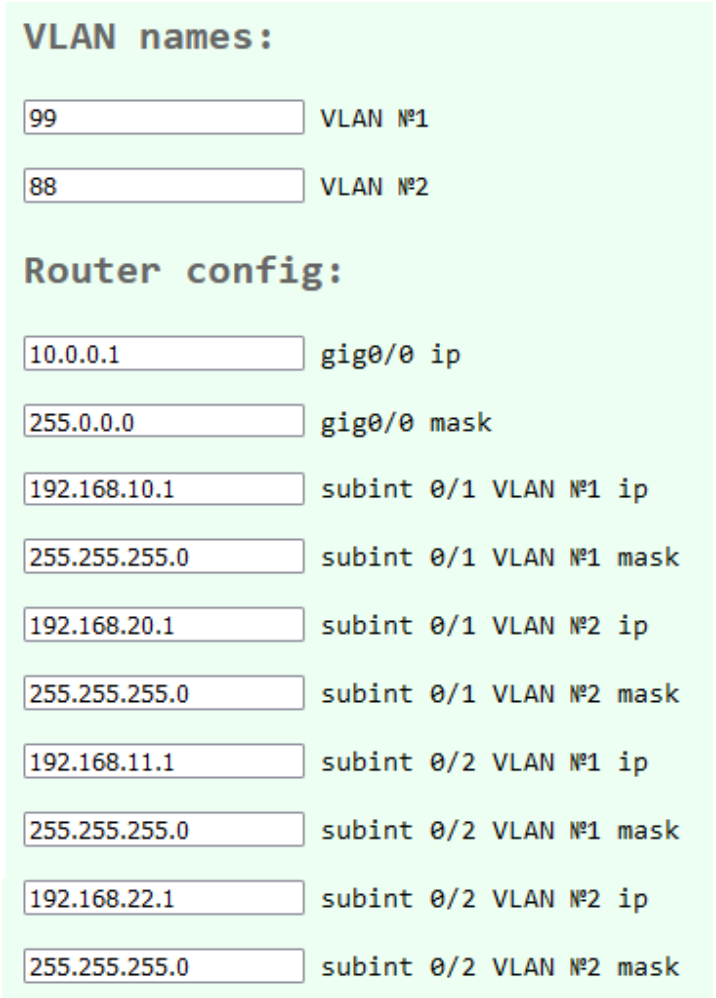

Рисунок 3.3 – поля введення даних для конфігурації мережі

Кнопки даного інтерфейсу мають різну функцію (рис. 3.4):

- reset – кнопка, що відповідає за видалення даних з полів;
- default – кнопка, що відповідає за введення значень по-замовчуванню у поля.
- generate – кнопка, що відповідає за скрипт збору всіх введених даних та створення самої конфігурації для мережі.

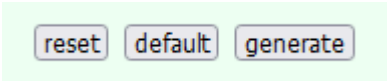

Рисунок 3.4 – кнопки для використання графічного інтерфейсу конфігурування

Поля мають валідаційну перевірку на правильне введення даних згідно з IPv4. Якщо ввести дані, що не включені у набір дозволених символів та порядку їх розміщення, з’явиться повідомлення про введення неправильних даних (рис. 3.5).

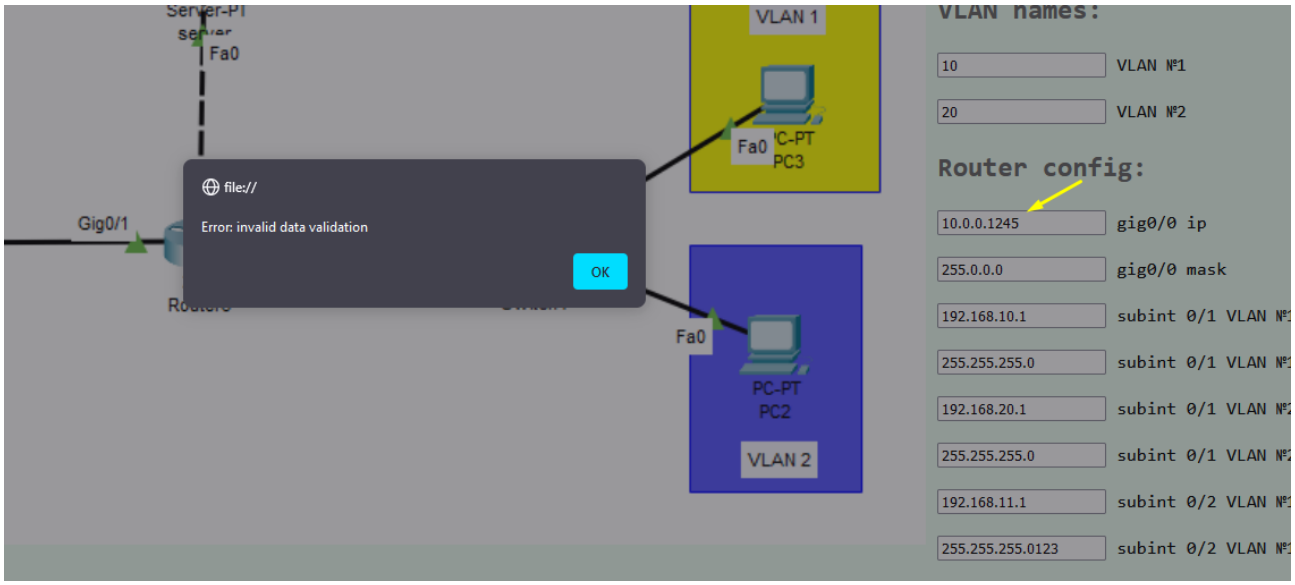

Рисунок 3.5 – повідомлення про неправильні дані

3.2 Практичне використання графічного інтерфейсу

Конфігурація з введеними даними з’являється у нижній частині інтерфейсу. Дані, що з’явились, можна використати у мережу, або зберегти їх у файлі для використання їх у подальшому. Для кожної конфігурації обладнання надається особисте місце розміщення самого налаштування – комутатори (рис. 3.6) та маршрутизатор (рис. 3.7) знаходяться у різних місцях.

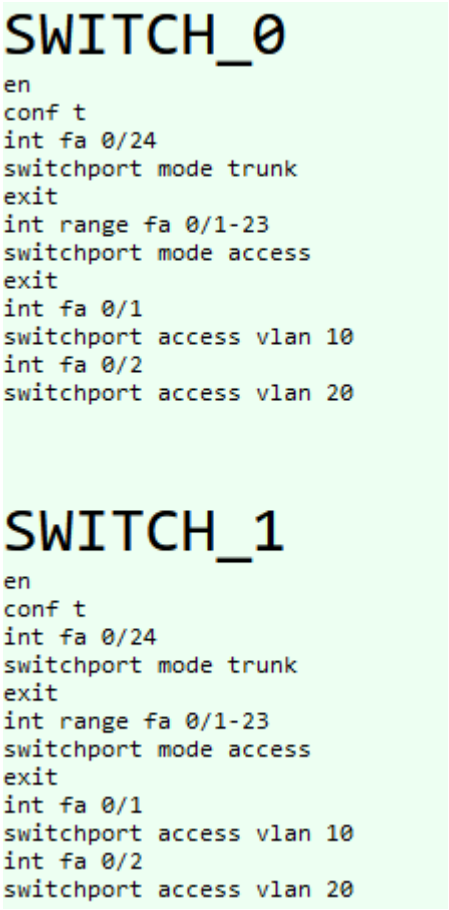

Рисунок 3.6 – приклад згенерованої конфігурації для комутаторів

ROUTER

```
en
conf t
int gig 0/0
ip add 10.0.0.1 255.0.0.0
ip nat outside
no sh

int gig 0/1
ip nat inside
no sh
int gig 0/2
ip nat inside
no sh
int gig 0/1.99
encapsulation dot1Q 99
ip add 192.168.10.1 255.255.255.0
ip nat inside
int gig 0/1.88
encapsulation dot1Q 88
ip add 192.168.20.1 255.255.255.0
ip nat inside
exit

int gig 0/2.99
encapsulation dot1Q 99
ip add 192.168.11.1 255.255.255.0
ip nat inside
int gig 0/2.88
encapsulation dot1Q 88
ip add 192.168.22.1 255.255.255.0
ip nat inside
exit

ip dhcp pool poolleftone
network 192.168.10.0 255.255.255.0
default-router 192.168.10.1
exit
ip dhcp pool poollefttwo
network 192.168.20.0 255.255.255.0
default-router 192.168.20.1
exit
ip dhcp pool poolrightone
network 192.168.11.0 255.255.255.0
default-router 192.168.11.1
exit
ip dhcp pool poolrighttwo
network 192.168.22.0 255.255.255.0
default-router 192.168.22.1
exit

ip access-list standard NAT
permit 192.168.10.0 0.0.0.255
permit 192.168.20.0 0.0.0.255
permit 192.168.11.0 0.0.0.255
permit 192.168.22.0 0.0.0.255
exit

ip nat inside source list NAT interface gig 0/0 overload
```

Рисунок 3.7 – приклад згенерованої конфігурації для маршрутизатора

Згенеровані дані можна одразу ж використати у конфігурації реального обладнання, програмі-емуляторі, або записати у файл для використання конфігурації у майбутньому.

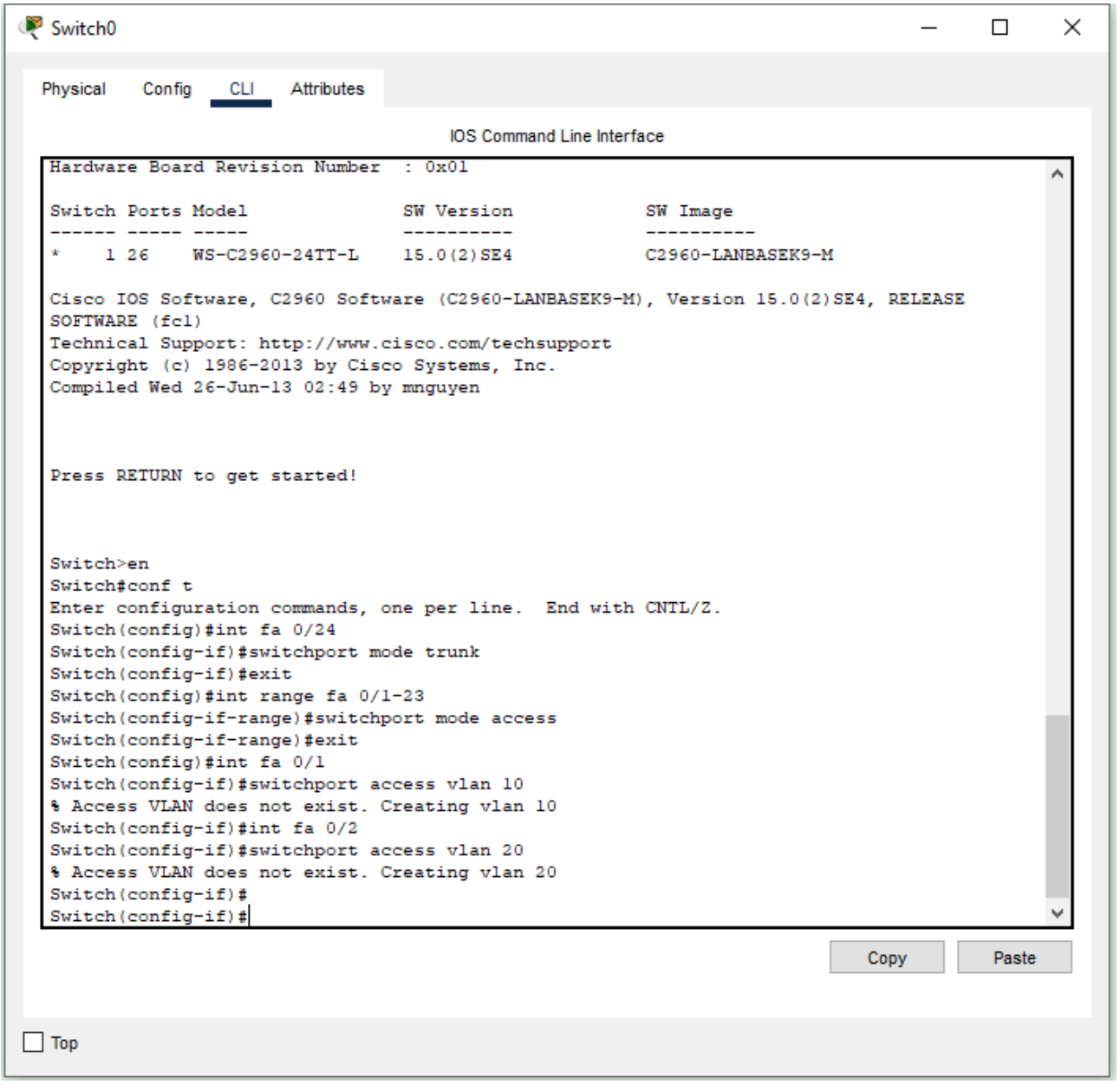

Рисунок 3.8 – приклад конфігурації комутатора у програмі Packet Tracer

Користувачу, що не розуміє, як правильно без вмінь налагоджувати мережу у системі, треба лише увійти у CLI (Command Line Interface) обраного приладу, та скопійовану конфігурацію вставити у відповідну строку. Після цього система прийме введені налаштування та почне свою роботу з конкретною логікою, яка була завчасно спроектована.

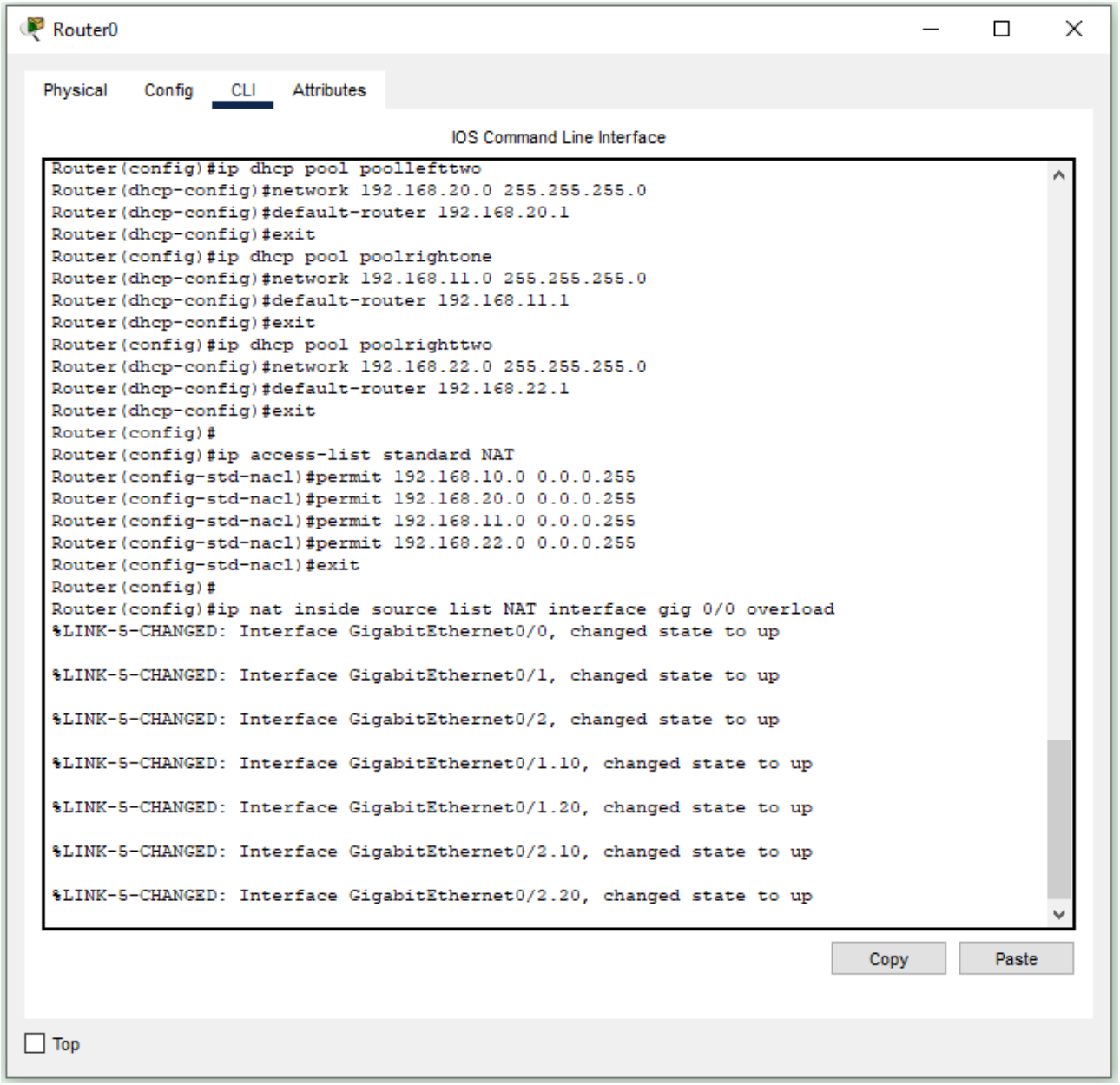

Рисунок 3.9 – приклад конфігурації маршрутизатора у програмі Packet Tracer

Таким чином можливо згенерувати конфігурацію для маршрутизатора та комутаторів з різними значеннями IP-адрес, їх масок, назв віртуальних локальних мереж, та їх саб-інтерфейсів без знань та практики спеціаліста з налагодження локальних мереж, які найчастіше використовуються у сферах IT-бізнесу.

ВИСНОВОК

В ході виконання кваліфікаційної роботи було знайдено та проаналізовано велику кількість джерел інформації, яка стосується проектування систем комп’ютерних мереж на базі програмного забезпечення Cisco Packet Tracer.

Локальна мережа офісу ІТ-компанії була спроектована у програмі-емуляторі Packet Tracer, у якій були використані нативні прилади компанії Cisco Systems: маршрутизатори та комутатори. Їх налаштування було проведено на нативній для даної системи мові конфігурування, що є частиною системи Cisco IOS.

При конфігуруванні мережевих приладів були використані: DHCP – створено чотири пули для кожного шлюзу з видачею певних IP-адрес та масок IP-адрес для кожного комп’ютера. Даний протокол дозволив швидко та автоматично налаштувати кожного клієнта у мережу; технологія VLAN – створено дві віртуальні локальні мережі для більш раціональної роботи комутаторів та для безпечної роботи мережі з уникненням прочитання неважливого трафіку, який переміщується до всіх учасників мережі; NAT протокол – налаштовано маршрутизатор на визначення внутрішньої локальної та зовнішньої глобальної мережі задля створення певної “клієнт-серверної” архітектури, щоб клієнт мав змогу запросити інформацію із серверу, а сервер з клієнта – ні.

Було спроектовано графічний інтерфейс автоматизованої конфігурації системи мережі для спрощення налаштування мережі з вказанням лише базових параметрів у самому інтерфейсі. Дана програма дозволить користувачам, що не мають практичних та теоретичних знань з конфігурацією мережевого обладнання Cisco швидко налаштувати його.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gillis, Alexander S. "What is DHCP (Dynamic Host Configuration Protocol)?". TechTarget: SearchNetworking. [Електронний ресурс] - Режим доступу до ресурсу: https://www.techtarget.com/searchnetworking/definition/DHCP

2. Організація комп'ютерних мереж: конспект лекцій: навчальний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення», спеціалізації «Програмне забезпечення комп'ютерних та інформаційно-пошукових систем» / Л.М. Олещенко ; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 208 с.

3. What is BOOTP? Network Encyclopedia [Електронний ресурс] – Режим доступу до ресурсу: https://networkencyclopedia.com/bootstrap-protocol-bootp/.

4. Режими операцій DHCP. Вікі ЦДУ [Електронний ресурс] - Режим доступу до ресурсу: https://wiki.cuspu.edu.ua/index.php/DHCP_Lyskov.

5. Організація комп'ютерних мереж: конспект лекцій: навчальний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення», спеціалізації «Програмне забезпечення комп'ютерних та інформаційно-пошукових систем» / Л.М. Олещенко ; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 40 - 162 с.

6. Віртуальні локальні мережі VLAN. Методичні вказівки до лабораторних, практичних занять і самостійної роботи з дисциплін «телекомунікаційні та інформаційні мережі», «телекомунікаційні та інформаційні мережі на залізничному транспорті», «мережеві технології», «інтегральні цифрові мережі зв'язку» / проф. С. І. Приходько, доценти О. С. Жученко, М. А. Штомпель, асист. С. В. Сколота; Харків, 2018 – 9 с.

7. Організація комп'ютерних мереж: конспект лекцій: навчальний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення», спеціалізації «Програмне забезпечення комп'ютерних та інформаційно-

пошукових систем» / Л.М. Олещенко ; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 215 с.

8. Full Cone NAT, Restricted Cone NAT and Symmetric NAT. The Cisco Learning Network [Електронний ресурс] – Режим доступу до ресурсу: https://learningnetwork.cisco.com/s/question/0D56e0000CWxJ9sCQF/lets-explain-in-details-full-cone-nat-restricted-cone-nat-and-symmetric-nat-terminologies-vs-cisco-nat-terminologies.

9. Cisco Packet Tracer [Електронний ресурс] — Режим доступу до ресурсу: https://www.netacad.com/.

ДОДАТОК

index.html:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="../css/style.css">

    <script src="https://code.jquery.com/jquery-3.7.0.min.js"></script>
    <script src="script.js"></script>

    <title>GENINT</title>
</head>

<body>
    <header>
        <h1>Welcome to GENINT</h1>
        <p>Generate your own parameters for local web</p>
    </header>
    <div id="background">
        <div id="image">
        </div>
        <div id="headcolor">
            <h3>VLAN names:</h3>
            <input type="text" id="vlan1_number" name="vlan1_number">
            <label for="vlan1_number">VLAN №1</label><br>
            <br>
```

```
<input type="text" id="vlan2_number" name="vlan2_number">
<label for="vlan2_number">VLAN №2</label><br>

<h3>Router config:</h3>
<input type="text" id="router_server_ip" name="router_server_ip">
<label for="router_server_ip">gig0/0 ip</label><br>
<br>
<input type="text" id="router_server_mask" name="router_server_mask">
<label for="router_server_mask">gig0/0 mask</label><br>
<br>

<input type="text" id="vlan1_gig0/1_ip" name="vlan1_gig0/1_ip">
<label for="vlan1_gig0/1_ip">subint 0/1 VLAN №1 ip</label><br>
<br>
<input type="text" id="vlan1_gig0/1_mask" name="vlan1_gig0/1_mask">
<label for="vlan1_gig0/1_mask">subint 0/1 VLAN №1 mask</label><br>
<br>

<input type="text" id="vlan2_gig0/1_ip" name="vlan2_gig0/1_ip">
<label for="vlan2_gig0/1_ip">subint 0/1 VLAN №2 ip</label><br>
<br>
<input type="text" id="vlan2_gig0/1_mask" name="vlan2_gig0/1_mask">
<label for="vlan2_gig0/1_mask">subint 0/1 VLAN №2 mask</label><br>
<br>

<input type="text" id="vlan1_gig0/2_ip" name="vlan1_gig0/2_ip">
<label for="vlan1_gig0/2_ip">subint 0/2 VLAN №1 ip</label><br>
<br>
<input type="text" id="vlan1_gig0/2_mask" name="vlan1_gig0/2_mask">
<label for="vlan1_gig0/2_mask">subint 0/2 VLAN №1 mask</label><br>
<br>
```

```
            <input type="text" id="vlan2_gig0/2_ip" name="vlan2_gig0/2_ip">
            <label for="vlan2_gig0/2_ip">subint 0/2 VLAN №2 ip</label><br>
            <br>
            <input type="text" id="vlan2_gig0/2_mask" name="vlan2_gig0/2_mask">
            <label for="vlan2_gig0/2_mask">subint 0/2 VLAN №2 mask</label><br>
            <br>

            <button id="reset">reset</button>
            <button id="default">default</button>
            <button id="generate">generate</button>
        </div>
    </div>
    <div id="container">
        <label for="switch0_conf">SWITCH_0</label>
        <div id="switch0_conf"><br><br></div>
        <label for="switch1_conf">SWITCH_1</label>
        <div id="switch1_conf"><br><br></div>
        <label for="router_conf">ROUTER</label>
        <div id="router_conf"><br><br></div>
    </div>
</body>

</html>
```

script.js:

```
$(document).ready(function () {
    var switch0_command_list = `
        en<br>
        conf t<br>
        int fa 0/24<br>
        switchport mode trunk<br>
        exit<br>
        int range fa 0/1-23<br>
```

```
    switchport mode access<br>
    exit<br>
    int fa 0/1<br>
    switchport access vlan {vlan1_number}<br>
    int fa 0/2<br>
    switchport access vlan {vlan2_number}<br>
    <br><br><br>
`

var switch1_command_list = `
    en<br>
    conf t<br>
    int fa 0/24<br>
    switchport mode trunk<br>
    exit<br>
    int range fa 0/1-23<br>
    switchport mode access<br>
    exit<br>
    int fa 0/1<br>
    switchport access vlan {vlan1_number}<br>
    int fa 0/2<br>
    switchport access vlan {vlan2_number}<br>
    <br><br><br>
`

var router_command_list = `
    en<br>
    conf t<br>
    int gig 0/0<br>
    ip add {router_server_ip} {router_server_mask}<br>
    ip nat outside<br>
    no sh<br>
```


int gig 0/1
ip nat inside
no sh
int gig 0/2
ip nat inside
no sh

int gig 0/1.{vlan1_number}
encapsulation dot1Q {vlan1_number}
ip add {vlan1_gig0/1_ip} {vlan1_gig0/1_mask}
ip nat inside
int gig 0/1.{vlan2_number}
encapsulation dot1Q {vlan2_number}
ip add {vlan2_gig0/1_ip} {vlan2_gig0/1_mask}
ip nat inside
exit

int gig 0/2.{vlan1_number}
encapsulation dot1Q {vlan1_number}
ip add {vlan1_gig0/2_ip} {vlan1_gig0/2_mask}
ip nat inside
int gig 0/2.{vlan2_number}
encapsulation dot1Q {vlan2_number}
ip add {vlan2_gig0/2_ip} {vlan2_gig0/2_mask}
ip nat inside
exit

ip dhcp pool poolleftone
network {vlan1_gig0/1_net} {vlan1_gig0/1_mask}
default-router {vlan1_gig0/1_ip}
exit

```
    ip dhcp pool poollefttwo<br>
    network {vlan2_gig0/1_net} {vlan2_gig0/1_mask}<br>
    default-router {vlan2_gig0/1_ip}<br>
    exit<br>
    ip dhcp pool poolrightone<br>
    network {vlan1_gig0/2_net} {vlan1_gig0/2_mask}<br>
    default-router {vlan1_gig0/2_ip}<br>
    exit<br>
    ip dhcp pool poolrighttwo<br>
    network {vlan2_gig0/2_net} {vlan2_gig0/2_mask}<br>
    default-router {vlan2_gig0/2_ip}<br>
    exit<br>
    <br>
    ip access-list standard NAT<br>
    permit {vlan1_gig0/1_net} 0.0.0.255<br>
    permit {vlan2_gig0/1_net} 0.0.0.255<br>
    permit {vlan1_gig0/2_net} 0.0.0.255<br>
    permit {vlan2_gig0/2_net} 0.0.0.255<br>
    exit<br>
    <br>
    ip nat inside source list NAT interface gig 0/0 overload<br>

`


var variable_field = {
    'vlan1_number': { label: 'vlan №1', v: '10', t: 'int' },
    'vlan2_number': { label: 'vlan №2', v: '20', t: 'int' },

    'router_server_ip': { label: 'interface ip to server', v: '10.0.0.1', t: 'ip' },
    'router_server_mask': { label: 'interface mask to server', v: '255.0.0.0', t: 'mask' },

    'vlan1_gig0/1_ip': { label: 'subint ip vlan №1 gig0/1', v: '192.168.10.1', t: 'ip' },
```

```
        'vlan1_gig0/1_mask': { label: 'subint mask vlan №1 gig0/1', v: '255.255.255.0', t: 'mask' },


        'vlan2_gig0/1_ip': { label: 'subint ip vlan №2 gig0/1', v: '192.168.20.1', t: 'ip' },
        'vlan2_gig0/1_mask': { label: 'subint mask vlan №2 gig0/1', v: '255.255.255.0', t: 'mask' },


        'vlan1_gig0/2_ip': { label: 'subint ip vlan №1 gig0/2', v: '192.168.11.1', t: 'ip' },
        'vlan1_gig0/2_mask': { label: 'subint mask vlan №1 gig0/2', v: '255.255.255.0', t: 'mask' },


        'vlan2_gig0/2_ip': { label: 'subint ip vlan №2 gig0/2', v: '192.168.22.1', t: 'ip' },
        'vlan2_gig0/2_mask': { label: 'subint mask vlan №2 gig0/2', v: '255.255.255.0', t: 'mask' },
    }


    Object.keys(variable_field).forEach(
        function (name) {
            document.getElementById(name).placeholder = (variable_field[name].t == 'ip') ? 'xxx.xxx.xxx.xxx' : variable_field[name].v.replaceAll(/./gi, 'x');
        });


    $('#default').click(
        function () {
            Object.keys(variable_field).forEach(
                function (name) {
                    document.getElementById(name).value = variable_field[name].v;
                });
            return;
        });


    $('#generate').click(
        function () {
```

```
        if (valid_var(variable_field)) {
            document.getElementById('switch0_conf').innerHTML =
replace_marks(switch0_command_list);
            document.getElementById('switch1_conf').innerHTML =
replace_marks(switch1_command_list);
            document.getElementById('router_conf').innerHTML =
replace_marks(router_command_list);
        } else {
            alert("Error: invalid data validation");
        }
        return;
    });

  $('#reset').click(
    function () {
      Object.keys(variable_field).forEach(
        function (name) {
          document.getElementById(name).value = '';
        });
    });
});

function valid_var(variable_field) {
  var is_valid = true;

  Object.keys(variable_field).forEach(
    function (name) {
      var el = document.getElementById(name);
      if (!el || !el.value) {
        is_valid = false;
        return is_valid;
      }
```

```
        validating = variable_field[name].t;

        if (validating == 'ip' && !/^(?:(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])\.){3}(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])$/.test(el.value)) {
            is_valid = false;
            return is_valid;
        }
        if (validating == 'mask' && !/^(?:(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])\.){3}(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])$/.test(el.value)) {
            is_valid = false;
            return is_valid;
        }
    });

  return is_valid;
}

function replace_marks(command_list) {
  var map = {
    'vlan1_gig0/1_net': ['vlan1_gig0/1_ip', 'vlan1_gig0/1_mask'],
    'vlan2_gig0/1_net': ['vlan2_gig0/1_ip', 'vlan2_gig0/1_mask'],
    'vlan1_gig0/2_net': ['vlan1_gig0/2_ip', 'vlan1_gig0/2_mask'],
    'vlan2_gig0/2_net': ['vlan2_gig0/2_ip', 'vlan2_gig0/2_mask'],
  };

  command_list.match(/{([^}]+)}/gi).forEach(
    function (mark) {
      var html_id = mark.slice(1, -1);
      var v = Object.keys(map).includes(html_id) ? network_vlan(
        document.getElementById(map[html_id][0]).value,
        document.getElementById(map[html_id][1]).value,
      )
```

```
                : document.getElementById(html_id).value;
            command_list = command_list.replaceAll(mark, v);
        })
    return command_list;
}

function convert_ip_int(ip) {
    return ip.split('.').map((octet, index, array) => {
        return parseInt(octet) * Math.pow(256, (array.length - index - 1));
    }).reduce((previous, current) => {
        return previous + current;
    });
}

function convert_int_ip(value) {
    return [
        (value >> 24) & 0xff,
        (value >> 16) & 0xff,
        (value >> 8) & 0xff,
        value & 0xff
    ].join('.');
}

function network_vlan(ip, mask) {
    var ip_as_str = convert_ip_int(ip).toString(2).padStart(32, '0');
    var mask_as_str = (~convert_ip_int(mask)).toString(2).length;
    var net_ip_as_str = ip_as_str.substring(0, 32 - mask_as_str) +
'0'.repeat(mask_as_str);
    return convert_int_ip(parseInt(net_ip_as_str, 2));
}
```

style.css:

```
@keyframes slideIn {
    0% {
```

```
        transform: translateY(-100%);
    }

    100% {
        transform: translateY(0);
    }
}

@keyframes right-left {
    0% {
        transform: translateX(150%);
    }

    100% {
        transform: translateX(0);
    }
}

@keyframes fadeIn {
    0% {
        opacity: 0;
    }

    100% {
        opacity: 1;
    }
}

body {
    margin: 0;
    background-color: #edfff2;
    font-family: monospace;
```

```
}

div#background {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 90vh;
    margin: 30px;
}

div#image {
    background-image: url("../img/system.png");
    background-size: contain;
    background-repeat: no-repeat;
    width: 70%;
    height: 100%;
    animation: fadeIn 1.5s ease-in-out;
}

div#container label {
    font-size: 36px;
}

header {
    background-color: #ffb0b0;
    padding: 20px;
    text-align: center;
    animation: slideIn 1s ease-in-out;
}

header h1 {
    font-size: 36px;
```

```
    color: #333;
    margin: 0;
    opacity: 0;
    animation: fadeIn 1s ease-in-out 0.5s forwards;
}

header p {
    font-size: 18px;
    color: #666;
    margin: 0;
    opacity: 0;
    animation: fadeIn 1s ease-in-out 1s forwards;
}

div#headcolor {
    margin-left: 10px;
    font-size: 16px;
    animation: right-left 1s ease-in-out;
}

div#headcolor h3 {
    font-size: 24px;
    color: #666;
}

div#headcolor label {
    animation: fadeIn 3s ease-in-out 0s forwards;
}
```