МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Графічний інтерфейс налаштування технології VRF-Lite»

Завідувач випускаючої кафедри — **Довбиш А.С.**

Керівник роботи — **Великодний Д.В.**

Студента групи ІН.мдн-91с — **Кейзеров П.В.**

СУМИ 2020

Сумський державний університет

(назва вузу)

Факультет *ЦЗДВН* Кафедра *Комп'ютерних наук*

Спеціальність *122 «Комп'ютерні науки»*

Затверджую:

зав. кафедри _____

"_____" _____ 20___р.

ЗАВДАННЯ

НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Кейзерова Павла Вікторовича

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Графічний інтерфейс налаштування технології - VRF-Lite
затверджую наказом по інституту від "____" _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи) _____

3. Вхідні данні до проекту (роботи) _____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Літературний огляд. 2) Моделювання технології vrf-lite та аналіз з використанням емулятора GNS3. 3) Створення графічного інтерфейсу для побудови VRF-Lite*

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

_____

Керівник _____

(підпис)

Завдання прийняв до виконання _____

(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1	*Літературний огляд*		
2	*Моделювання технології vrf-lite та аналіз з використанням емулятора gns3*		
3	*Створення графічного інтерфейсу для побудови VRF-Lite*		
4	*Оформлення кваліфікаційної магістерської роботи*		

Студент – дипломник _____

(підпис)

Керівник проекту _____

(підпис

РЕФЕРАТ

Записка: 55 ст., 37 рис., 23 джерела, 2 додатки

Мета роботи — створення графічного інтерфейсу для спрощення в налаштуванні на маршрутизаторах Cisco технології віртуальної маршрутизації VRF-Lite.

Об'єкт дослідження — налаштування VRF-Lite в мережах Ethernet на базі маршрутизаторів CISCO.

Предмет дослідження — набір команд для забезпечення безперебійного доступу до мережі Ethernet.

Методи дослідження — моделювання в емуляторі мереж GNS3.

Результати — була створена веб-орієнтована система. Графічний інтерфейс який дозволяє в автоматичному режимі вивести команди для налаштування маршрутизатора, на якому вимагається реалізація віртуальної маршрутизації. Створений графічний інтерфейс дає можливість зменшення часу на конфігурацію маршрутизатора з допомогою генерації команд в автоматичному режимі на маршрутизатор. Інтерфейс має захист від введення помилкових даних. Реалізація була виконана з допомогою мови програмування JavaScript.

ГРАФІЧНИЙ ИНТЕРФЕЙС, VRF-LITE, NAT, EIGRP, GRE, GNS3, CISCO, ВІРТУАЛЬНА МАРШРУТИЗАЦІЯ,ВЕБ-ОРІЄНТОВАНА СИСТЕМА, JAVASCRIPT, HTML, CSS

ЗМІСТ

ВСТУП

В сучасному світі потреба бізнесу в доступі до всесвітній мережі дуже велика. Для нього це запорука фінансового росту. У кожного з підприємців настає такий час коли потрібно розширювати бізнес і збільшувати внутрішню мережу та шукати захист свого бізнесу від перебоїв підключення до інтернету, бо незначні перешкоди в мережі можуть спровокувати проблеми з банківськими операціями, звітністю бухгалтерії т.д.. Тому вони шукають різноманітні рішення для зменшення витрат та збільшенню функціоналу наявної мережі. Одним з можливих рішень може бути використання віртуальна маршрутизація (VRF).

Обираючи яким чином буде реалізоване розширення мережі організації та безперебійний доступ до мережі, використовують навчально-експериментальні симулятори мереж. Вони пропонують більш швидке та гнучке налаштування без потреби придбання обладнання для тестування і витрат на обладнання.

Використання технології VRF-Lite дозволить організувати безперебійний доступ до мережі та захистити від нестабільності, відключення і втрати пакетів в мережі. Тому було прийняте рішення створити зрозумілий та простий в налаштуванні графічний інтерфейс для конфігурації безперебійного підключення до мережі за допомогою технології VRF-Lite на роутерах компанії Cisco. Його використання буде можливе не лише для емуляторів та симуляторів але для конфігурації реального обладнання.

Графічним інтерфейсом зможуть користуватися не лише фахівці з досвідом для пришвидшення своєї роботи, так і користувачів новачків для навчання.

1 ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 Технологія VRF та її основні типи

1.1.1 VRF MPLS

Технологія Virtual Routing and Forwarding (VRF) має широке застосування в мережах MPLS. У мережах мітки MPLS застосовуються для поділу трафіку користувачів, а VRF підтримує таблицю маршрутизації для кожного з них окремо. Для обміну маршрутною інформацією в таких мережах застосовується MP-BGP. Але є можливість обійтися без технології MPLS такий варіант використання VRF називається - VRF-Lite [1].

Віртуальна маршрутизація та переадресація (VRF) дозволяє в маршрутизаторі існувати кільком екземплярам таблиці маршрутизації, представляє можливість мати таблицю VRF за замовчуванням та створені користувачем таблиці. Одна таблиця може обробляти різні типи протоколів маршрутизації, таких як EIGRP, OSPF, BGP, IGRP і т. д.. Кожен протокол маршрутизації в таблиці VRF вказаний як запис. На додаток до обробки декількох типів загальних протоколів маршрутизації, ви можете налаштувати протокол маршрутизації для посилання на інтерфейс з іншого VRF. Це дозволяє сегментувати мережеві шляхи без використання кількох пристроїв (рис. 1.1) [1,2] .

Ви можете створити кілька віртуальних маршрутизаторів для підтримки окремих таблиць маршрутизації для груп інтерфейсів. Оскільки кожен віртуальний маршрутизатор має свою власну таблицю маршрутизації, можна забезпечити чіткий розподіл трафіку, що проходить через пристрій [2].

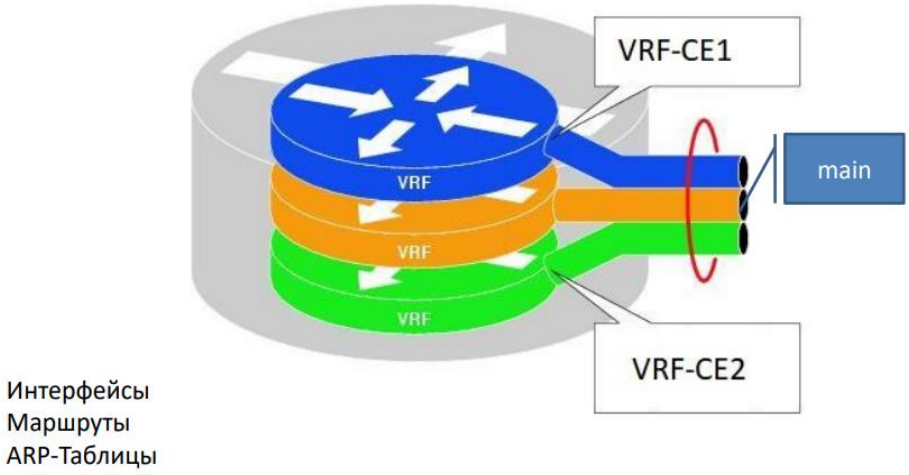

Рисунок 1.1 — Структура віртуалізації

1.1.2 VRF-Lite

У цій реалізації кожен маршрутизатор в мережі бере участь в середовищі віртуальної маршрутизації на рівноправній основі. Незважаючи на те, що VRF-Lite простий в розгортанні і підходить для малих і середніх підприємств і загальних центрів обробки даних, він не масштабується до розміру, необхідного для глобальних підприємств або великих операторів зв'язку, оскільки існує необхідність впроваджувати кожен екземпляр VRF на кожному маршрутизаторі, включаючи проміжні маршрутизатори. Спочатку VRF були введені в поєднанні з багатопротокольною комутацією міток (MPLS), але VRF виявився настільки корисним, що в кінцевому підсумку перетворився в незалежну технологію від MPLS [3].

Оскільки кожен екземпляр віртуального маршрутизатора працює автономно, мережевий трафік на призначених інтерфейсах відділяється від трафіку, керованого іншими віртуальними маршрутизаторами. Це особливий поділ мереж збільшує безпеку без необхідності використання VPN, як в загальній мережі. Оскільки можна використовувати одні і ті ж IP-адреси або діапазони IP на декількох віртуальних маршрутизаторах, які можуть навіть перекриватися без конфлікту один з одним, віртуальні маршрутизатори

також можуть використовуватися для управління мережевим трафіком для декількох мереж з ідентичними конфігураціями одночасно [3, 4].

Віртуальну маршрутизацію можна використовувати для сегментації мережевих шляхів без використання додаткових пристроїв (рис.1.2). Кількість мережевих шляхів обмежена тільки кількістю доступних інтерфейсів. Концепція поділу трафіку на декількох мережевих шляхах може бути скасована шляхом налаштування правил доступу, які перенаправляють певний трафік з однієї колії на іншу. Наприклад, весь трафік з різних мережевих шляхів може бути спрямований на загальний, що веде в Інтернет, тим самим зменшуючи накладні витрати на налаштування правил доступу для маршрутизації трафіку, який одночасно переходить з різних мереж в Інтернет. Комбінуючи різні типи інтерфейсів, такі як мережа, VLAN або пов'язані інтерфейси, різні можливості пропонують величезну різноманітність програм [4].

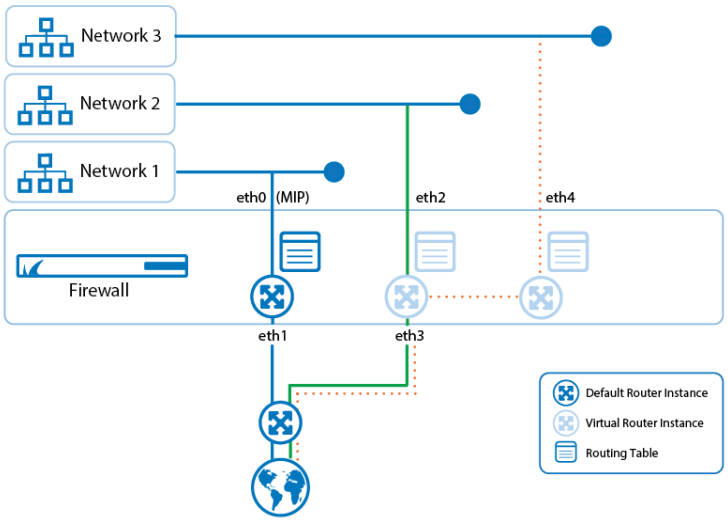

Рисунок 1.2 — Схематична реалізація VRF

1.2 Технологія NAT основні відомості

NAT (Network Address Translation) - технологія трансляції мережевих адрес, тобто заміни адрес в заголовку IP-пакета. Він дозволяє приватним IP-мережам, що використовують незареєстровані IP-адреси, підключатися до Інтернету. NAT працює на маршрутизаторі, зазвичай поєднуючи дві мережі разом, і перетворює приватні (не рідкість в глобальному масштабі) адреси у внутрішній мережі в юридичні адреси, перш ніж пакети будуть перенаправлені в іншу мережу [5, 6].

В рамках цієї можливості NAT може бути налаштований для оголошення тільки однієї адреси для всієї мережі зовнішнього світу. Це забезпечує додаткову безпеку, ефективно приховуючи за цією адресою всю внутрішню мережу. NAT пропонує подвійні функції безпеки і збереження адрес, зазвичай реалізується в середовищах віддаленого доступу.

NAT дозволяє одному пристрою, наприклад маршрутизатору, виступати в якості агента між Інтернетом (чи загальнодоступною мережею) і локальною мережею (або приватною мережею), що означає, що потрібно тільки один унікальний IP-адрес для представлення цілої групи комп'ютерів для чого-небудь за межами своєї мережі [7].

Ви можете дозволити внутрішнім користувачам доступ в Інтернет, але у вас може не вистачити дійсних адрес для всіх. Якщо всі комунікації з пристроями в Інтернеті відбуваються з боку внутрішніх пристроїв, вам знадобиться один дійсний адрес або пул дійсних адресів.

На цьому рис. 1.3 показана проста мережева діаграма з інтерфейсами маршрутизатора, визначеними як внутрішні і зовнішні [8]:

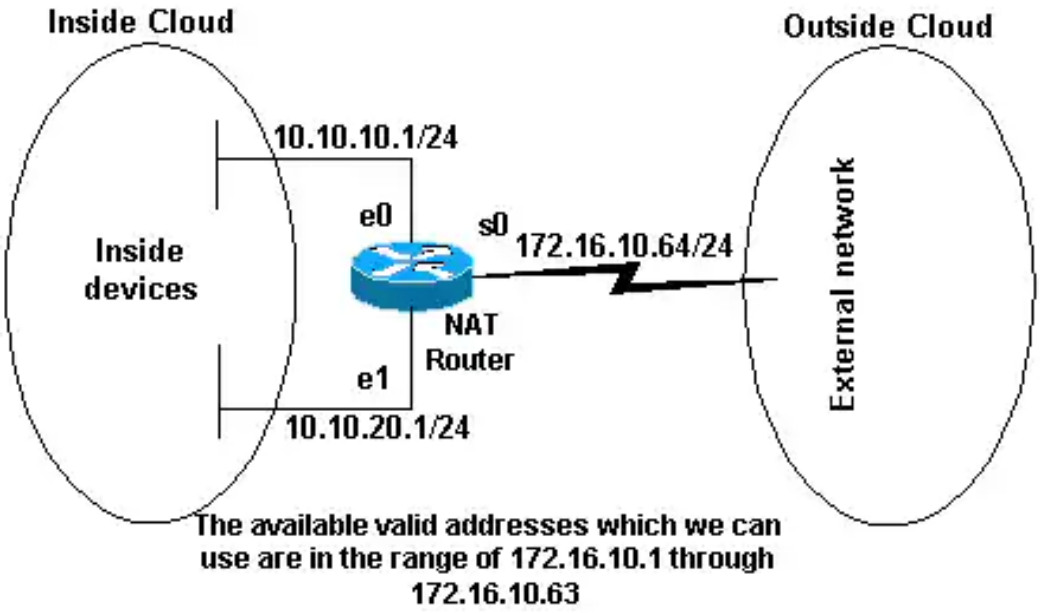

Рисунок 1.2 — Конфігурація NAT

Класифікація NAT

1. Static NAT - статичний NAT задає однозначну відповідність однієї адреси іншому.

2. Dynamic NAT - при проходженні через маршрутизатор, нова адреса вибирається динамічно з деякого пулу адресів. Запис про трансляцію зберігається деякий час, щоб відповідні пакети могли бути доставлені адресату. Якщо протягом деякого часу трафік по цій трансляції відсутній, трансляція видаляється і адреса повертається в пул. Якщо потрібно створити трансляцію, а вільних адрес в пулі немає, то пакет відкидається.

3. Dynamic NAT with overload або PAT. Працює майже також, як dynamic NAT, але при цьому відбувається трансляція багато-в-один, використовуючи при цьому можливості транспортного рівня.

4. IP NAT пули - діапазон IP адрес, які виділяються для NAT трансляції в міру необхідності [9, 10].

1.3 Протокол GRE основні відомості

GRE (Generic Routing Encapsulation) –універсальна інкапсуляція при маршрутизації - це протокол тунелювання, здатний інкапсулювати різні протоколи мережевого рівня між двома об'єктами по загальнодоступній мережі, наприклад, в Інтернеті.

GRE можна використовувати в наступних ситуаціях:

- підключення мережі IPv6 по мережах IPv4;
- багато адресна розсилка пакетів, наприклад OSPF і EIGRP, а також потокового передавання даних.

Протокол GRE інкапсулюється безпосередньо в IP, минаючи TCP або UDP, в IP пакеті є спеціальне поле «Protocol type», в якому міститься число, що позначає протокол, інкапсульований в даний IP пакет, для GRE protocol type дорівнює 47. У зв'язку з цим, якщо в мережі є GRE тунелі, то варто уважно ставитися до написання розширених списків контролю доступу [11].

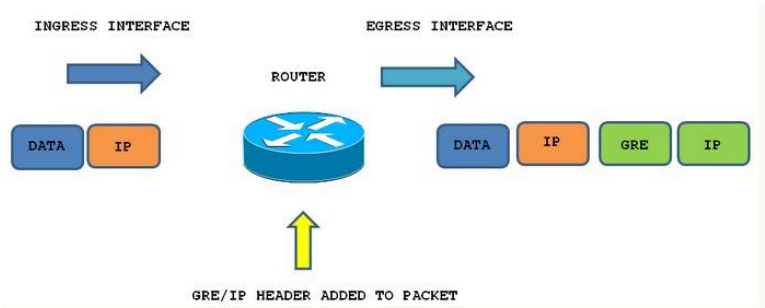

Рисунок 1.4 — Процес інкапсуляції GRE пакета

Як бачимо на (рис. 1.4) , IP інкапсулюється в GRE, який інкапсулюється в IP. При цьому заголовок GRE і заголовок зовнішнього IP пакета додають 24 байта, відповідно, зменшуючи розмір пакета на цю величину. В даному прикладі ми налаштовуємо IPv4 в GRE в IPv4, проте, GRE може успішно працювати з іншими інкапсулюємими і транспортними протоколами, наприклад, IPv6 [12].

1.4 Протокол EIGRP основні відомості

EIGRP — вдосконалений дистанційно-векторний протокол динамічної маршрутизації, розроблений компанією Cisco.

Основні характеристики EIGRP:

- Швидка збіжність (в порівнянні з іншими дистанційно-векторними протоколами)
- Підтримка VLSM
- Часткові оновлення
- Підтримка різних протоколів мережевого рівня (IP, IPX, AppleTalk)
- Однакові налаштування протоколу при використанні різних протоколів канального рівня (наприклад, у OSPF настройки відрізняються для Ethernet і Frame Relay)
- Складна метрика
- Використання multicast (224.0.0.10) і unicast адрес, замість широкомовної розсилки

Метрика протоколу EIGRP використовує мінімальну пропускну здатність на шляху до мережі призначення і загальну затримку для обчислення показників маршрутизації. Можна також налаштувати і інші метрики, однак ми не рекомендуємо робити цього, оскільки в цьому випадку у вашій мережі можуть з'явитися петлі по маршрутизації. Метрики пропускної здатності і затримки визначаються зі значень, що налаштовуються в інтерфейсах маршрутизаторів на шляху до мережі призначення (рис.1.5) [13].

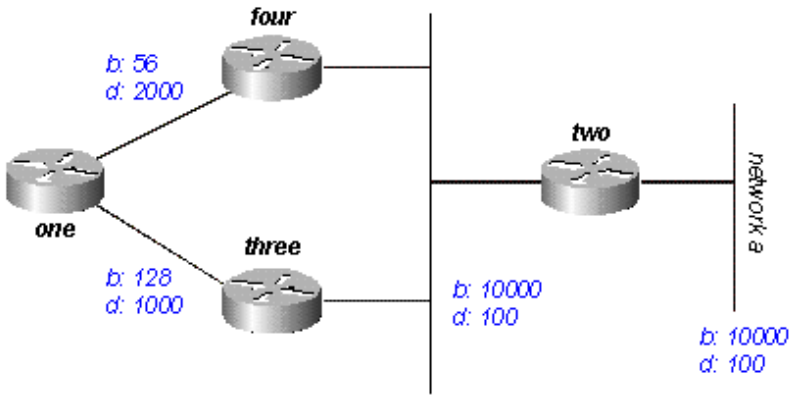

Рисунок 1.5 — Маршрутизатор 1 обчислює найкращий шлях в мережу A

Метрика EIGRP заснована на таких 5 компонентах (за замовчуванням використовуються тільки два):

- Bandwidth - найменша bandwidth між source і destination (використовується за умовчанням);
- Delay - cumulative interface delay всього шляху;
- Reliability - найгірший показник надійності на всьому шляху, на підставі keepalive;
- Loading - найгірший показник завантаження посилання на всьому шляху, на підставі packet rate і налаштованої bandwidth на інтерфейсі;
- MTU - найменше MTU на всьому шляху. MTU включається в оновлення EIGRP, але фактично не використовується для підрахунку метрики [14].

За замовчуванням для підрахунку метрики використовуються bandwidth і delay. Решта критеріїв не рекомендується використовувати, так як це призведе до постійних підрахунків маршрутів.

EIGRP підтримує кілька типів маршрутів, у кожного типу маршруту своє значення administrative distance:

- internal - внутрішні маршрути EIGRP.

- external - маршрути, перерозподілені в процес EIGRP з інших джерел.
- summary - сумарні маршрути EIGRP. (за замовчуванням, може бути будь-який - задається адміністратором при написанні команди підсумовування) [10].

1.5 Постановка задачі

Ознайомившись з літературою, пов'язаною з технологією VRF-Lite, постановку задачі можу сформулювати таким чином: необхідно створити веб-орієнтовану систему, графічній інтерфейс, в якому є можливість автоматично зробити конфігурацію команд для налаштування інтерфейсів роутера вибраної мною технології. Він зможе забезпечити зручний набір згенерованого коду налаштувань в симуляторі GNS3 і на реальному обладнанні Cisco копіюванням з головного екрану веб-орієнтованої системи.

Програма повинна дозволити початківцям мати можливість налаштовувати віртуальну маршрутизацію, не вимагаючи знань команд конфігурації роутерів компанії Cisco.

Для реалізації графічного інтерфейсу буде створена веб-сторінка, де від користувача буде потрібно заповнити поля IP-адресу та маску і отримати команди налаштування, які можна скопіювати і перенести в симулятор GNS3 або на реальне обладнання. В результаті буде налаштований маршрутизатор, який буде виконувати завдання безперебійного підключення до Internet.

Постановка задачі:

1. Створення базової схеми для реалізації VRF- Lite в емуляторі GNS3.

2. Налаштування технологій VRF- Lite в емуляторі GNS3.

3. Перевірка функціонування налаштованої топології за допомоги GNS3.

4. Розробка графічного інтерфейсу налаштування VRF-Lite.

5. Тестування розробленої веб-орієнтованої програми в емуляторі GNS3.

2 МОДЕЛЮВАННЯ ТЕХНОЛОГІЇ VRF-Lite ТА АНАЛІЗ З ВИКОРИСТАННЯМ ЕМУЛЯТОРА GNS3

2.1 Емулятор комп'ютерних мереж GNS3

Graphical Network Simulator-3 (скорочений до GNS3) — це мережевий графічний симулятор мережі з графічним інтерфейсом (рис. 2.1), який дозволяє моделювати складні мережі, змоделювати віртуальну мережу з маршрутизаторів і віртуальних машин. Незамінний інструмент для навчання та роботи. Працює практично на всіх платформах. Дуже добре підходить для створення стендів на десктоп машинах. Залежно від апаратної платформи, на якій буде використовуватися GNS3, можлива побудова комплексних проектів, що складаються з маршрутизаторів Cisco, Cisco ASA, Juniper, а також серверів під управлінням мережевих операційних систем [15].

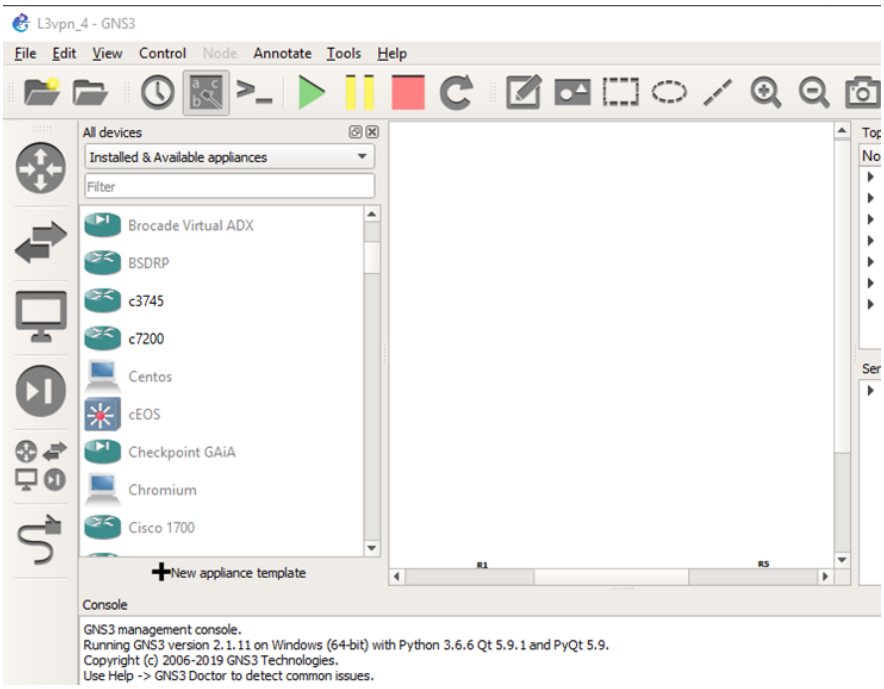

Рисунок 2.1 — Стартове вікно GNS3

При не можливості отримання доступу до реального обладнання, GNS3 надасть повний комплект можливостей для реалізації проекту . Він спроможний поєднувати віртуальні та фізичні пристрої, які можуть знадобитися для моделювання великих та складних мереж [16].

Поширене використання GNS3 в якості лабораторного стенду, де можна перевірити яке облачення спроможне підтримати ту чи іншу технологію чи схему. GNS3 за принципом роботи не симулятор, а емулятор. Варто розуміти що це різні між собою поняття. Емулятор дозволяє створити необхідну модель пристрою або комп'ютера і запускати на цих пристроях оригінальне програмне забезпечення. Доступні всі основні компоненти пристрою, в тому числі процесор, пам'ять і пристрої введення / виводу. У випадку з обладнанням Cisco, емулятор створює модель маршрутизатора і всередині запускає операційну систему Cisco IOS. Ми отримуємо віртуальній повнофункціональний маршрутизатор. Він імітує поведінку системи і інтерфейсу.

Перевагами GNS 3 є:

1. Перша і найважливіша причина — повний функціонал віртуальних пристроїв. Використовуючи IOS маршрутизаторів Cisco (рис. 2.2), нам доступні майже всі функції, які можуть бути на реальному маршрутизаторі.
2. Можливість побудови гетерогенних мереж. Це значить, що ми можемо використовувати в одному проекті схему де будуть не тільки пристрої Cisco, а й Juniper, Mikrotik, CheckPoint і т.д.
3. Використовування в мережі повноцінних робочих станцій і серверів. У GNS3 ми можемо додати віртуальну машину з використанням програмного забезпечення, наприклад комп'ютер з Windows 7 або Ubuntu використовуючи VirtualBox чи її аналог. Також є можливість використовувати Windows Server або RedHat.

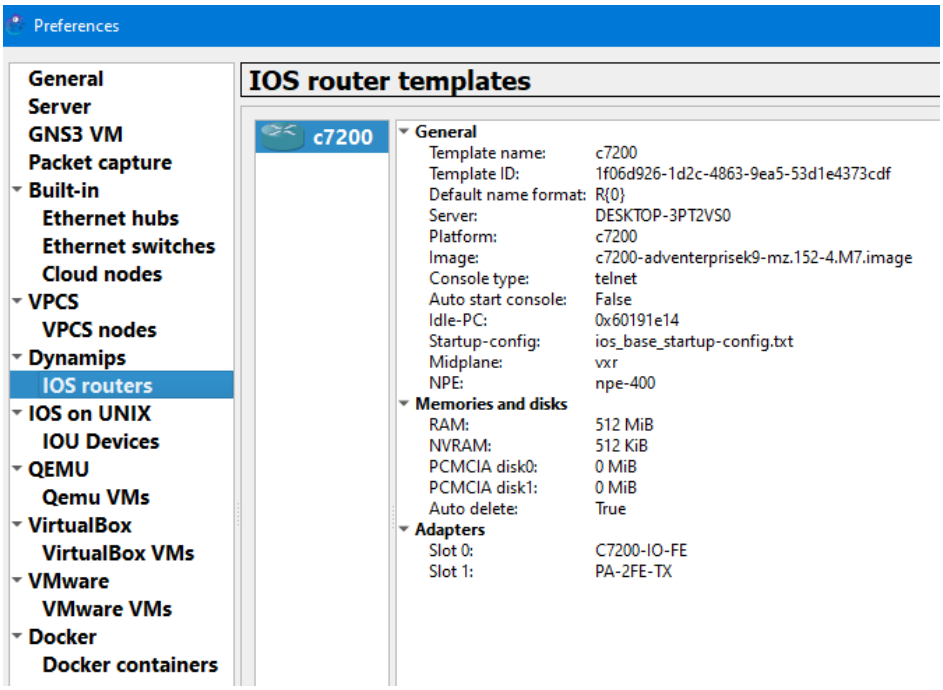

Рисунок 2.2 — Використання IOS роутера

Однак GNS 3 має і свої недоліки, а саме:

1. Істотний недолік — відсутність здатності емулювати комутатори. В реальних комутаторах кількість ASIC мікросхем дуже велика, їх поки що не є можливим емулювати на комп'ютері. Саме ASIC дають можливість обробки пакетів з величезною швидкістю. Маршрутизатори виконують команди на основі процесора, який схожий на комп'ютерний. Тому є можливість емуляції маршрутизаторів і з цим не виникає проблем [17].
2. Ще одним важливим недолік — дуже високі потреби системних ресурсів. Особливо велика потреба в багатому обсязі оперативної пам'яті та не менша потреба в хорошому центральному процесорі.
3. Третій недолік — недостатньо інтуїтивно зрозумілий графічний інтерфейс. В свою чергу він дозволяє подивитися задану топологію, тобто з’єднання та опис пристроїв, найчастіше цього буває замало.

2.2 Створення та тестування схеми

Для основи я обрав схему на якій ми будемо використовувати двох провайдерів та підключеної через маршрутизатор та Hab локальної сітки компанії. Топологія цієї схеми більш складна ніж інші, але і більш гнучка в конфігурації маршрутів. Спочатку треба зробити базове налаштування роутеру.

Розпочнемо створювати базову топологію підключення провайдерів щоб наш віртуальний офіс мав вихід до мережі (рис.2.2). Ця схема була обрана для реалізації технології в GNS3, та на її базі буде створений графічний інтерфейс [18].

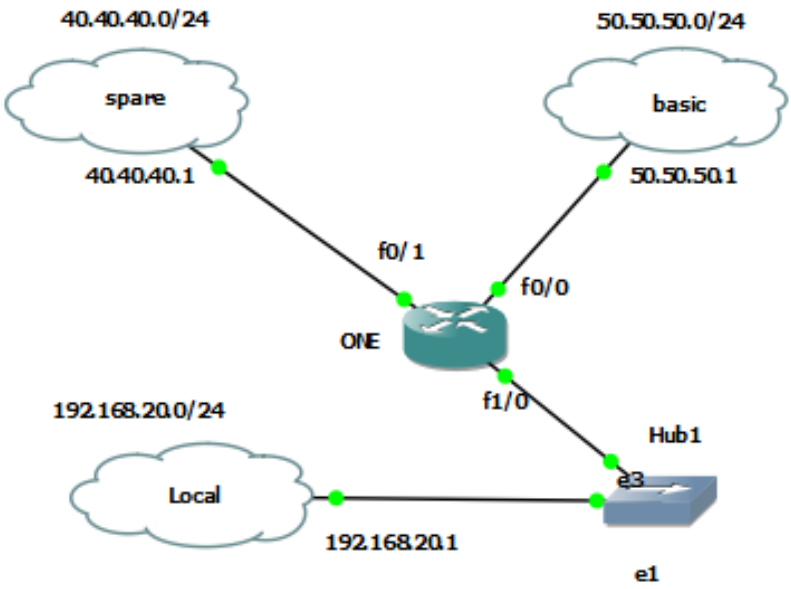

Рисунок 2.2 — Загальний вигляд схеми

Починаємо налаштовувати роутер, для початку задаємо дані для кожного порту (рис. 2.3). Команда description призначена для надання нагадування в конфігурації для опису того, для чого використовуються певні інтерфейси. Опис відображається у результатах наступних команд, таких як «show interface» та «show running-config». Ці команди можна використовувати на таких інтерфейсах: Interface Ethernet [19].

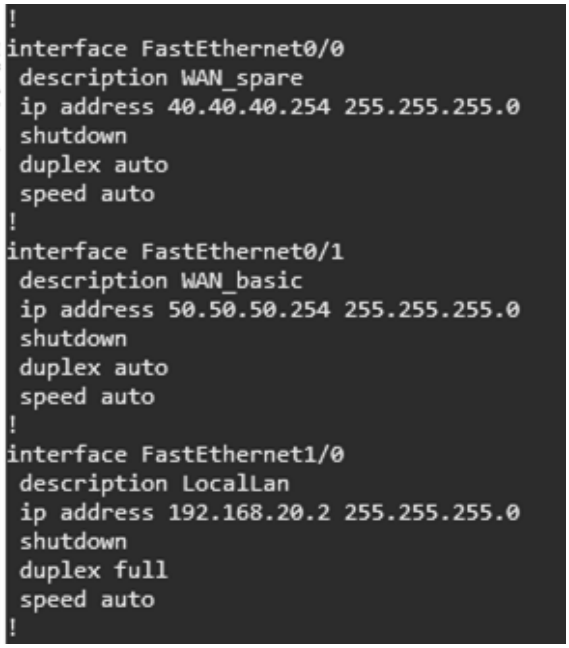

Рисунок 2.3 — Базові налаштований на роутері ONE

Після цього даємо ім’я для кожної VRF яку будемо використовувати, їх буде три. Дві VRF створюємо ми, при цьому створюється третя global VRF (рис.2.4). Командою Route Distinguisher є єдине, але дуже важливе завдання зробити свідомо не унікальний префікс унікальним. Виходить це за допомогою додавання 64-бітного Route Distinguisher до шуканого префіксу. Наприклад, маючи два VRF на одному інтерфейсі і Route Distinguisher 65000: 1 і 65000: 2, отримуємо унікальні префікси. Так як маршрутизатор знає довжину Route Distinguisher (вона фіксована і дорівнює 64 бітам) і на початку префікса, то йому не складає труднощів відкинути значення перших 64-х біт і помістити в таблицю маршрутизації клієнта тільки IPv4 префікс [20].

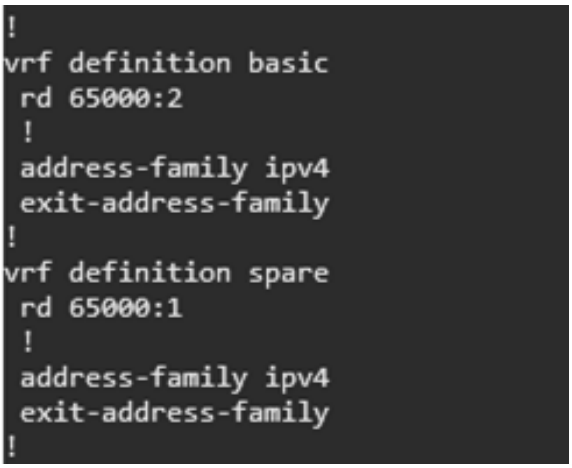

Рисунок 2.4 — Відображення VRF на роутері

Тепер можемо прикріпити кожен VRF до свого порту для того, щоб роутер розумів від якого провайдеру буде отримувати трафік и т.д. (рис.2.6).

```
conf t
interface FastEthernet0/0
vrf forwarding spare
ip address 40.40.40.254 255.255.255.0
no sh
exit
interface FastEthernet0/1
vrf forwarding basic
ip address 50.50.50.254 255.255.255.0
no sh
exit
```

Рисунок 2.6 — Присвоєння свого порту VRF

Рисунок 2.7 — Дані інтерфейсів

Отже переконавшись за допомогою команди «sh ip route vrf», що роутер підготовлений до подальшого налаштування (рис.2.7) ми можемо продовжити конфігурацію роутера ONE.

2.3 Налаштування GRE тунелів та NAT аналіз пакетів за допомогою емулятора GNS3

Спочатку требо об’єднати два створенні VRF з global VRF, для обміну маршрутами та виходу LocalLan в Internet.

Розпочнемо налаштовувати тунелі (рис.2.8), відразу будемо налаштовувати EIGRP для кожного провайдера. Аналогічна конфігурація буде для другого провайдера [21].

```
ONE#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
ONE(config)#interface Loopback11
ONE(config-if)#ip address 192.168.102.1 255.255.255.255
ONE(config-if)#interface Loopback12
ONE(config-if)#ip address 192.168.102.2 255.255.255.255
ONE(config-if)#ex
*Nov 22 21:37:00.615: %LINEPROTO-5-UPDOWN: Line protocol on In
*Nov 22 21:37:00.967: %LINEPROTO-5-UPDOWN: Line protocol on In
ONE(config-if)#ex
ONE(config)#interface Tunnel11
ONE(config-if)#ip address 192.168.102.13 255.255.255.252
ONE(config-if)#ip mtu 1500
ONE(config-if)#tunnel source 192.168.102.1
ONE(config-if)#tunnel destination 192.168.102.2
ONE(config-if)#ex
*Nov 22 21:37:19.283: %LINEPROTO-5-UPDOWN: Line protocol on In
ONE(config-if)#ex
ONE(config)#
ONE(config)#interface Tunnel12
ONE(config-if)#vrf forwarding spare
ONE(config-if)#ip address 192.168.102.14 255.255.255.252
ONE(config-if)#ip mtu 1500
ONE(config-if)#tunnel source 192.168.102.2
ONE(config-if)#tunnel destination 192.168.102.1
ONE(config-if)#ex
*Nov 22 21:37:44.747: %LINEPROTO-5-UPDOWN: Line protocol on In
ONE(config-if)#ex
ONE(config)#
```

Рисунок 2.8 — Налаштування тунелів на роутері

Налаштувавши тунелі, перевіримо їх використавши команду «sh ip route» в консолі на роутері (рис. 2.).

```
interface Tunnel11
 ip address 192.168.102.13 255.255.255.252
 ip mtu 1500
 tunnel source 192.168.102.1
 tunnel destination 192.168.102.2
!
interface Tunnel12
 vrf forwarding spare
 ip address 192.168.102.14 255.255.255.252
 ip mtu 1500
 tunnel source 192.168.102.2
 tunnel destination 192.168.102.1
!
```

Рисунок 2.9 — Тунелі на інтерфейсі роутерах

Однак поки що ми не можемо спілкуватися між локальними мережами. Це спричинене тим, що ми не налаштували EIGRP. Тепер приступило до налаштування протоколу динамічної маршрутизації (рис. 2.10).

```
ONE(config)#router eigrp 10
ONE(config-router)#address-family ipv4 vrf spare autonomous-system 10
ONE(config-router-af)#default-metric 16000 630 254 1 1500
ONE(config-router-af)#redistribute connected
ONE(config-router-af)#redistribute static
ONE(config-router-af)#network 0.0.0.0
ONE(config-router-af)#network 192.168.102.12 0.0.0.3
ONE(config-router-af)#exit-address-family
```

Рисунок 2.10 — Налаштування протоколу EIGRP

Після налаштування протоколу динамічної маршрутизації, вводимо команду «sh ip route eigrp» на нашому роутері. Дивимося що вийшло з маршрутами переданими по EIGRP. Як бачимо все добре, тому можемо продовжувати. (рис.2.11).

```
ONE#sh ip route eigrp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is 192.168.102.14 to network 0.0.0.0

D*EX  0.0.0.0/0 [170/27041280] via 192.168.102.14, 00:50:30, Tunnel11
      40.0.0.0/24 is subnetted, 1 subnets
D        40.40.40.0 [90/26882560] via 192.168.102.14, 01:06:57, Tunnel11
```

Рисунок 2.5 — Відображення EIGRP

Тепер приходимо до налаштування NAT та заповненню Access-list (рис.2.12).

```
router eigrp 10
 !
 address-family ipv4 vrf spare autonomous-system 10
  default-metric 16000 630 254 1 1500
  redistribute connected
  redistribute static
  network 0.0.0.0
  network 192.168.102.12 0.0.0.3
 exit-address-family
 network 192.168.20.0
 network 192.168.102.12 0.0.0.3
!
ip forward-protocol nd
no ip http server
no ip http secure-server
!
!
ip nat inside source list 101 interface FastEthernet0/0 vrf spare overload
ip route vrf spare 0.0.0.0 0.0.0.0 40.40.40.1 name spare_GR
!
access-list 101 remark NAT_spare
access-list 101 deny   ip any 10.0.0.0 0.255.255.255
access-list 101 deny   ip any 172.16.0.0 0.15.255.255
access-list 101 deny   ip any 192.168.0.0 0.0.255.255
access-list 101 permit ip 192.168.0.0 0.0.255.255 any
no cdp log mismatch duplex
!
```

Рисунок 2.12 — NAT та Access-list

Тепер налаштовуємо аналогічно іншого провайдера (рис. 2.13).

```
!
interface Loopback21
 ip address 192.168.102.3 255.255.255.255
!
interface Loopback22
 ip address 192.168.102.4 255.255.255.255
!
interface Tunnel11
 ip address 192.168.102.13 255.255.255.252
 ip mtu 1500
 tunnel source 192.168.102.1
 tunnel destination 192.168.102.2
!
interface Tunnel12
 vrf forwarding spare
 ip address 192.168.102.14 255.255.255.252
 ip mtu 1500
 ip nat inside
 ip virtual-reassembly in
 tunnel source 192.168.102.2
 tunnel destination 192.168.102.1
!
interface Tunnel21
 ip address 192.168.102.17 255.255.255.252
 ip mtu 1500
 tunnel source Loopback21
 tunnel destination 192.168.102.4
!
```

Рисунок 2.6 — Налаштування GRP першого та другого провайдера

Технологія NAT та Access-List провайдерів (рис. 2.14).

```
ip nat inside source list 102 interface FastEthernet0/0 vrf basic overload
ip nat inside source list 101 interface FastEthernet0/0 vrf spare overload
ip route vrf spare 0.0.0.0 0.0.0.0 40.40.40.1 name spare_GR
ip route vrf basic 0.0.0.0 0.0.0.0 50.50.50.1 name basic_GR
!
access-list 101 remark NAT_spare
access-list 101 deny   ip any 10.0.0.0 0.255.255.255
access-list 101 deny   ip any 172.16.0.0 0.15.255.255
access-list 101 deny   ip any 192.168.0.0 0.0.255.255
access-list 101 permit ip 192.168.0.0 0.0.255.255 any
access-list 102 remark NAT_basic
access-list 102 deny   ip any 10.0.0.0 0.255.255.255
access-list 102 deny   ip any 172.16.0.0 0.15.255.255
access-list 102 deny   ip any 192.168.0.0 0.0.255.255
```

Рисунок 2.7 — NAT та Access-List

Налаштувавши протокол динамічної маршрутизації, NAT, GRP, Access-list для кожного провайдеру використавши команду «sh ip route» на нашому роутері. Дивимося що тепер у нас два дефолтних машрути, це відбувається тому що метрики однакові. Тобто пінгуя щось за нашими провайдерами роутер буде відправляти пакети то через одного провайдера, то через второго (рис. 2.8).

```
D*EX  0.0.0.0/0 [170/27041280] via 192.168.102.18, 00:08:18, Tunnel21
                [170/27041280] via 192.168.102.14, 00:08:18, Tunnel11
      40.0.0.0/24 is subnetted, 1 subnets
D        40.40.40.0 [90/26882560] via 192.168.102.14, 02:08:22, Tunnel11
      50.0.0.0/24 is subnetted, 1 subnets
D        50.50.50.0 [90/26882560] via 192.168.102.18, 00:08:44, Tunnel21
      192.168.20.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.20.0/24 is directly connected, FastEthernet1/0
L        192.168.20.2/32 is directly connected, FastEthernet1/0
      192.168.102.0/24 is variably subnetted, 8 subnets, 2 masks
C        192.168.102.1/32 is directly connected, Loopback11
C        192.168.102.2/32 is directly connected, Loopback12
C        192.168.102.3/32 is directly connected, Loopback21
C        192.168.102.4/32 is directly connected, Loopback22
C        192.168.102.12/30 is directly connected, Tunnel11
L        192.168.102.13/32 is directly connected, Tunnel11
C        192.168.102.16/30 is directly connected, Tunnel21
L        192.168.102.17/32 is directly connected, Tunnel21
ONE#
```

Рисунок 2.8 — Топологія двох провайдерів

Виправимо це, нехай у нас буде basic пріоритетним. Для EIGRP є декілька рішень - встановити delay або bandwich. Використаємо перше рішення - де менше delay - там і краще маршрут(рис. 2.9).

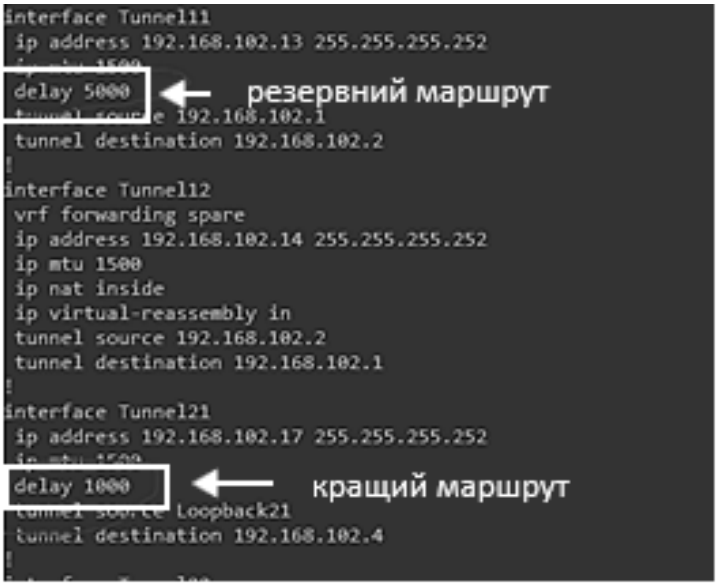

Рисунок 2.9 — Показники delay для провайдерів

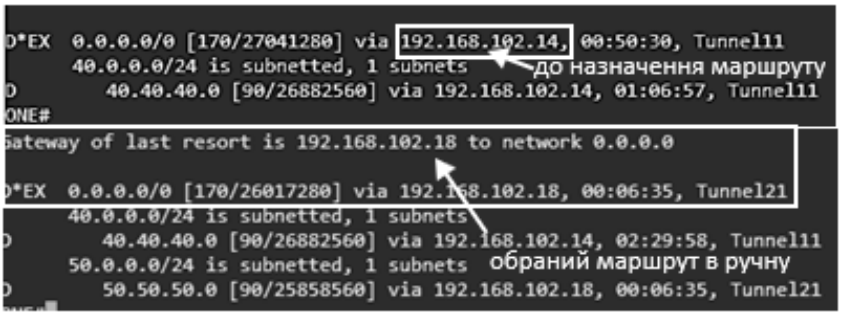

Рисунок 2.17 — Порівняння до і після позначення delay

Ми можемо бачити (рис. 2.10) що тепер роутер буде виводити дані за межі локальної сітки за допомоги провайдера з найменшою затримкою мережі. І ось ми підійшли до головного резервації провайдерской лінії.

2.4 Налаштування перенаправлення пакетів та аналіз роботи за допомогою емулятора GNS3

І ось ми можемо бачимо, що наша схема працює, але треба захистити внутрішню мережу від збоїв або відключення основного провайдера. Залишилося лише дуже уважно налаштувати перенаправлення пакетів на резервний провайдер у випадку відмови основного.

Налаштування для цього (рис. 2.18) та перевірка результату (рис. 2.19).

```
ip sla auto discovery
ip sla 1
 icmp-echo 40.40.40.1
 vrf spare
 threshold 1000
 timeout 1500
 frequency 3
ip sla schedule 1 life forever start-time now
ip sla 2
 icmp-echo 50.50.50.1 source-ip 50.50.50.254
 vrf basic
 threshold 1000
 timeout 1500
 frequency 3
ip sla schedule 2 life forever start-time now
```

Рисунок 2.18 — Конфігурація роутера

```
ONE#
*Nov 23 00:58:23.502: %SYS-5-CONFIG_I: Configured from console by console
ONE#sh track br
Track   Object                          Parameter        Value Last Change
1       ip sla      1                   reachability     Up    00:04:16
2       ip sla      2                   reachability     Up    00:01:35
ONE#
```

Рисунок 2.11 — Результат набраних команд

Тепер перевіримо роботу нашої схеми, для цього спочатку подивимося за допомогою команди «sh ip route eigrp» (рис. 2.12) та «sh track br» (рис. 2.12) конфігурацію нашого роутера.

```
ONE#sh ip route eigrp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is 192.168.102.18 to network 0.0.0.0

D*EX  0.0.0.0/0 [170/26017280] via 192.168.102.18, 00:17:25, Tunnel21
      8.0.0.0/32 is subnetted, 1 subnets
D EX     8.8.8.8 [170/27041280] via 192.168.102.14, 00:23:10, Tunnel11
      40.0.0.0/24 is subnetted, 1 subnets
D        40.40.40.0 [90/26882560] via 192.168.102.14, 03:31:18, Tunnel11
      50.0.0.0/24 is subnetted, 1 subnets
D        50.50.50.0 [90/25858560] via 192.168.102.18, 01:07:55, Tunnel21
ONE#
```

Рисунок 2.20 — Перевірка налаштованого EIGRP

```
ONE#sh track br
Track   Object                          Parameter       Value Last Change
1       ip sla      1                   reachability    Up    00:22:56
2       ip sla      2                   reachability    Up    00:20:14
ONE#
```

Рисунок 2.21 — Перевірка налаштованих тунелів

Ми можемо бачити, що схема функціонує в звичайному режимі. Тепер ми повинні перевірити нашу конфігурацію. Прибравши зв'язок з провайдером «basic» роутер повинен визначити, що основний провайдер не відповідає (рис. 2.22) і треба перенаправити пакети на резервованого провайдера (рис. 2.23).

```
ONE#sh track br
Track   Object                          Parameter       Value Last Change
1       ip sla      1                   reachability    Up    00:00:28
2       ip sla      2                   reachability    Down  00:00:03
ONE#
```

Рисунок 2.12 — Повідомлення про відмову основного провайдера

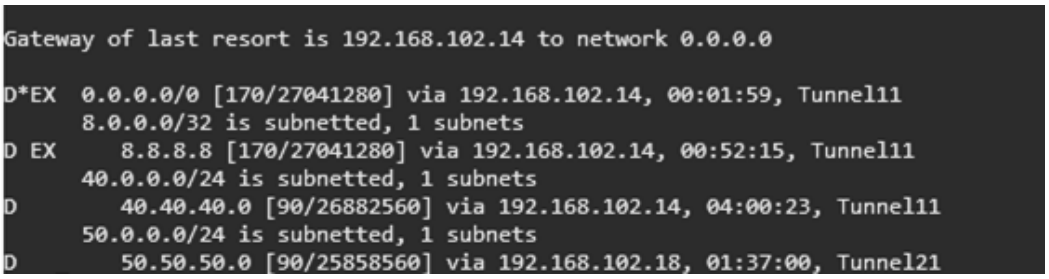

Рисунок 2.13 — Роутер перекинув трафік на зарезервованого провайдера

Приведені вище дані підтверджують працездатність схеми. Вона повністю може забезпечити постійний доступ до мережі невеликій компанії. Є декілька альтернатив побудування схеми. Можна використовувати протокол BGP, але на мій погляд, в наш час розглянута схема найкращій варіант для середнього та малого бізнесу, так як вона хоч більш заморочлива, але і більш гнучка в налаштуваннях маршрутів.

Тепер приступимо до створення графічного інтерфейсу на базі нашої схеми.

2.5 Мова програмування JavaScript

Мова JavaScript, для створення інтерфейсу, була обрана, тому що вона дозволяє реалізувати його у вигляді веб-додатку, бо в сучасному світі усі звикли до веб-сторінок. Кожен інтуїтивно може зрозуміти як з ними взаємодіяти, та вони підтримуються безліччю приладів. Також були використані мови HTML та CSS. Але основу веб-додатку буде виконувати мова JavaScript, тому про неї я розповім більше.

JavaScript ("JS" для стислості) - це повноцінний динамічна мова програмування, яка застосовується до HTML документів, і може забезпечити динамічну інтерактивність на веб-сайтах. Його розробив Brendan Eich, співзасновник проекту Mozilla, Mozilla Foundation і Mozilla Corporation.

JavaScript неймовірно універсальний і доброзичливий до новачків. Маючи великий досвід, ви зможете створювати ігри, анімовану 2D і 3D графіку, повномасштабні програми з базами даних і багато іншого [22].

JavaScript сам по собі досить компактний, але дуже гнучкий. Розробниками написано велику кількість інструментів поверх основної мови JavaScript, які розблокують величезну кількість додаткових функцій з дуже невеликими зусиллями. До них відносяться:

- Програмні інтерфейси додатка (API), вбудовані в браузери, що забезпечують різні функціональні можливості, такі як динамічне створення HTML і установку CSS стилів, захоплення і маніпуляція відеопотоком, робота з веб-камерою користувача або генерація 3D графіки і аудіо семплів.
- Сторонні API дозволяють розробникам впроваджувати функціональність в свої сайти від інших розробників, таких як Twitter або Facebook.
- Також ви можете застосувати до вашого HTML сторонні фреймворки і бібліотеки, що дозволить вам прискорити створення сайтів і додатків [22, 23].

Вона являється мовою програмування, що дозволяє реалізувати ряд складних рішень в web-документах. Вона допомагає зробити сторінки сайту більш інтерактивними, обробляє дії користувачів. Це об'єктно-орієнтована клієнтська мова, яка підтримується додатками, що працюють з дизайном сайту [23].

Ознайомившись з JS я можу сказати, що її використання допоможе зробити інтерфейс більш зрозумілим для новачків, та покращить розуміння виконаної роботи. Забезпечить захист від невірних даних, та дасть гнучкість у використанні так як її можна запустити на будь якому пристрої з браузером та найголовніше не потребує великого об’єму ресурсів приладу та підключення до інтернету.

3 СТВОРЕННЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ ДЛЯ ПОБУДОВИ VRF-LITE

3.1 Створення графічного інтерфейсу налаштування VRF-Lite з використанням мови програмування JavaScript.

Технологія VRF-Lite була сконфігуровано в GNS3, де за допомогою команд було зроблено налаштування роутера. За результатами побудованої схеми було виявлено мінус емулятора GNS3 - є відсутність графічного інтерфейсу, який би спростив конфігурацію мультисервісних задач. Тому було прийняте рішення вирішити актуальну задачу по розробці веб-орієнтованого графічного інтерфейсу для автоматичної конфігурації VRF-Lite.

Відкритий веб-інтерфейс дає змогу ознайомитись з структурою інтерфейсу. Вона показує 5 блоків, які в свою чергу дають користувачеві розуміння структури схеми, 4 кнопки, 1 поле для отримання конфігурації. Вводячи IP адресу та маску для налаштування роутера, потрібно натиснути клавішу «Конфігурація» у блоці «Конфігурації» після цього з`являться команди для роутера.

Інтерфейс веб-орієнтованого графічного інтерфейсу є зручним та інтуїтивно зрозумілим навіть для нового користувача (рис 3.1).

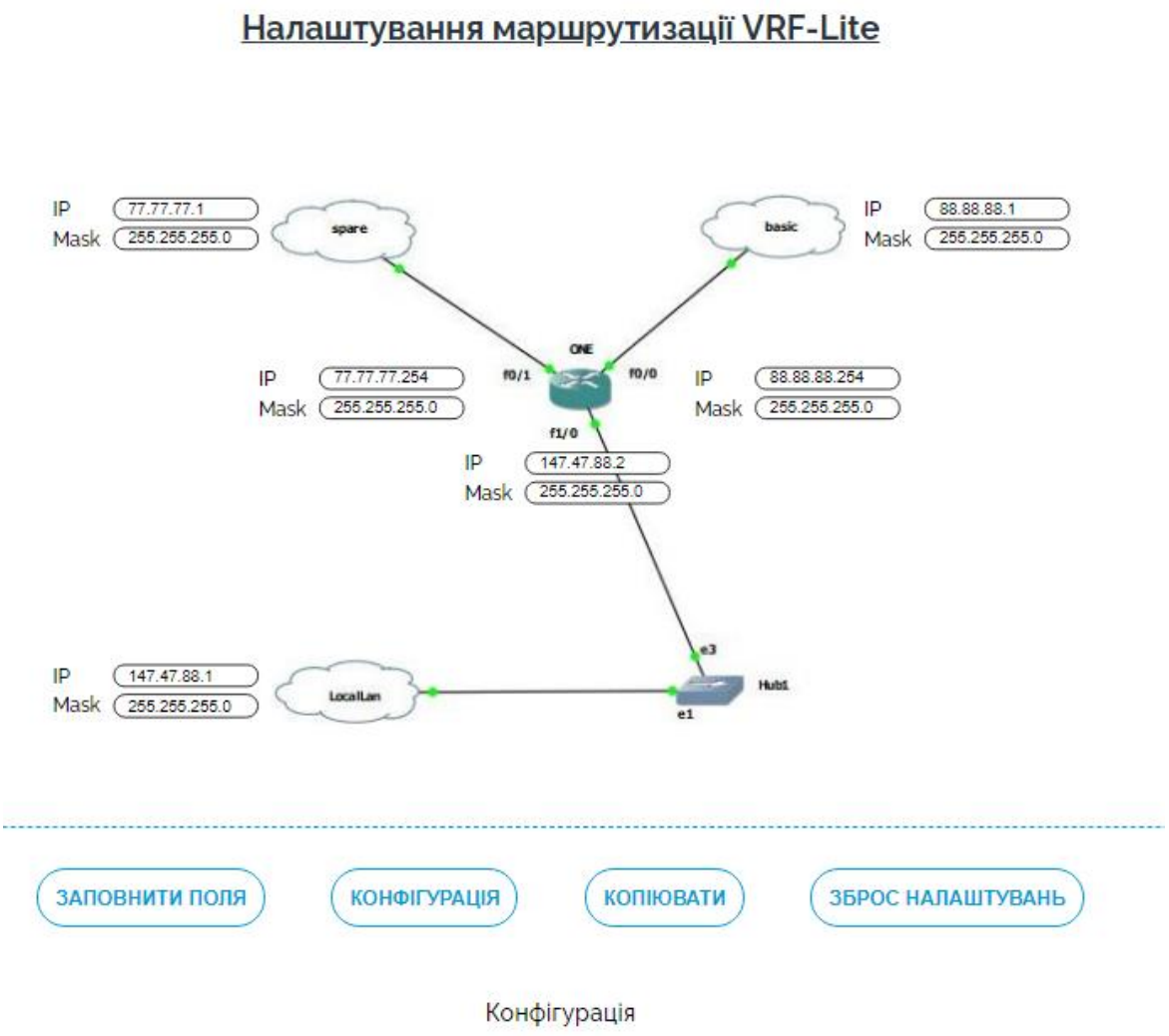

Рисунок 3.1 – Вигляд графічного інтерфейсу

Графічний інтерфейс перевіряє поля IP та MASK на правильність. Якщо буде розпізнаній невірний формат, чи порожня форма то користувач отримає у блоці «Конфігурації» попередження про невірний IP чи MASK, в конкретному полі для зручності знаходження помилки, та не будуть генеровуватися команди (рис. 3.2).

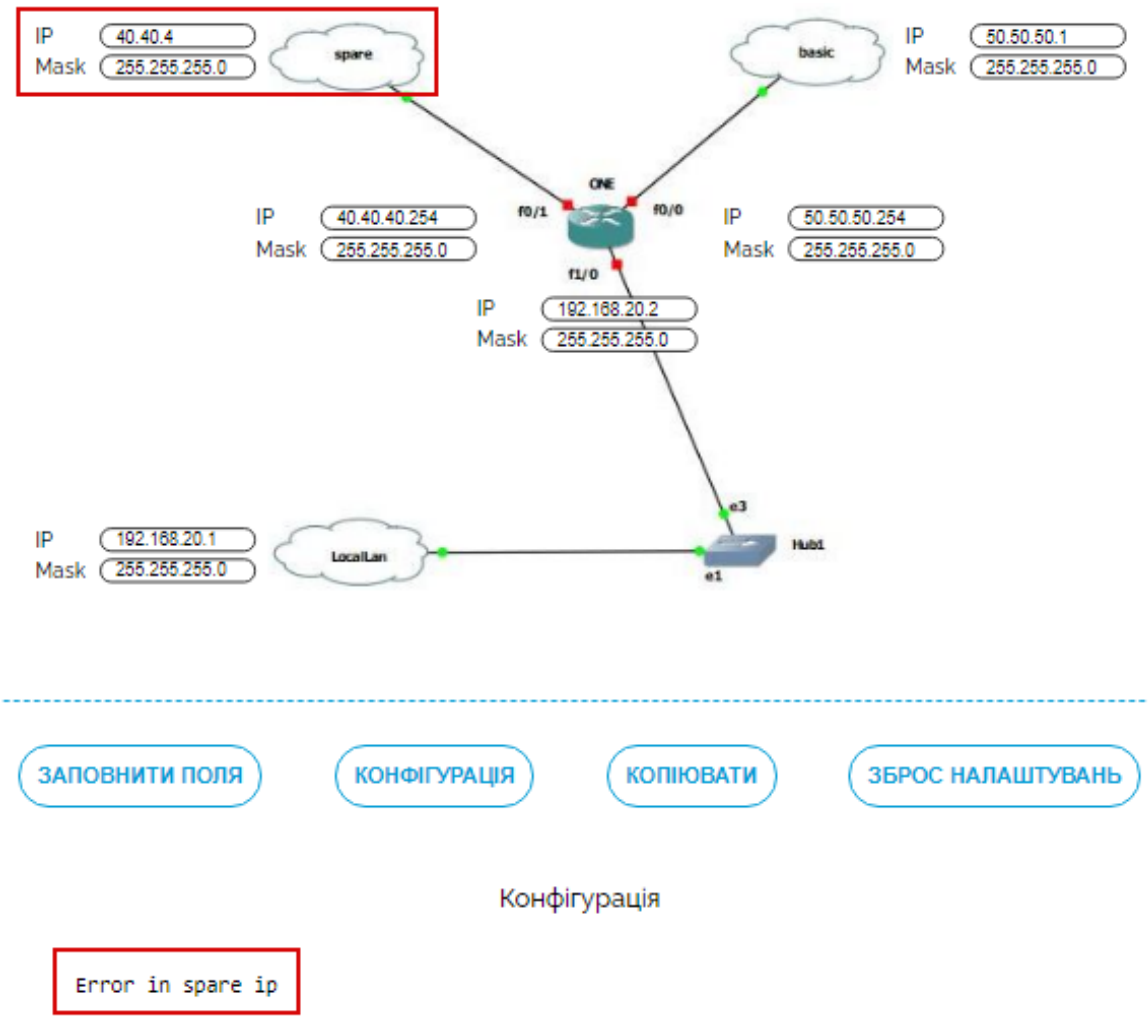

Рисунок 3.2 – Реагування інтерфейсу на помилку у формі IP адреси на провайдері «spare»

Інтерфейс здатен генерувати код для 1 роутера. Команди знаходяться у текстовому вікні, якщо користувач заповнив правильно всі поля, після цього скористувавшись кнопкою «Конфігурація» як показано на рис. 3.3.

Рисунок 3.3 – Скомпільовані команди для налаштувань роутера

Команд для конфігурації дуже багато, для зручності та швидкості можна скористатися кнопкою «Копіювати» (рис. 3.4). Кнопка виконує дію копіювання команд до буферу обміну, можна використати ці налаштування, щоб вставити їх на емульований або фізичний роутер.

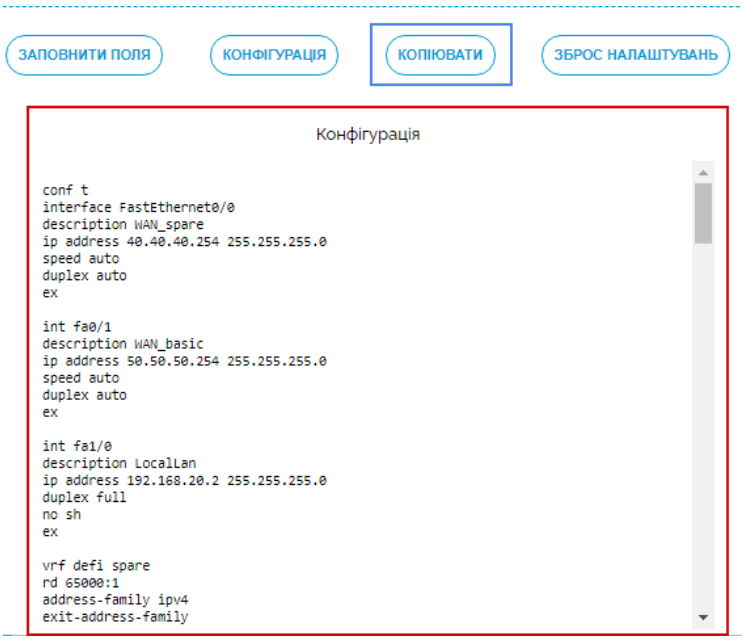

Рисунок 3.4 – Генерований код в блоці «Конфігурації» з кнопкою «Копіювати»

3.2 Тестування графічного інтерфейсу налаштування VRF-Lite

Спочатку в інтерфейсі використаємо «Заповнити поля», з її допомогою ми швидко заповнимо форму базовими даними(рис. 3.5).

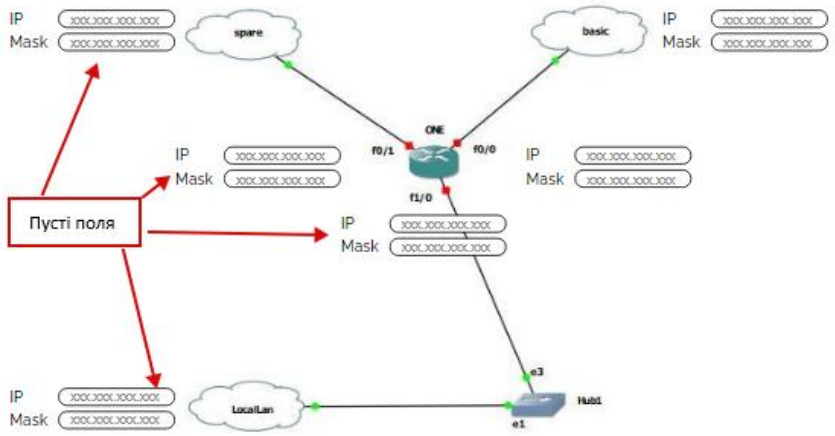

Рисунок 3.5 – Вигляд форми до натискання «Заповнити поля»

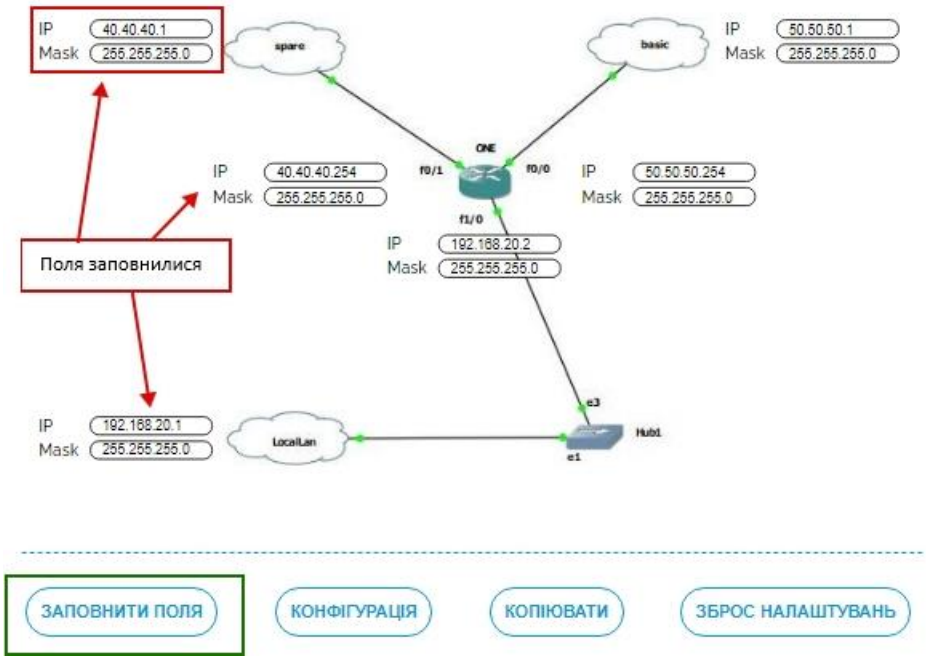

Рисунок 3.6 – Вигляд форми після натискання «Заповнити поля»

Натискаємо кнопку «Конфігурація». Отримаємо команди в блоці «Конфігурації» для налаштування технології VRF-Lite.

Налаштувавши маршрутизатор за допомогою команд, які згенерувалися, переконаємось у вірному налаштуванні, використавши команду «sh run», де ми бачимо конфігурацію роутера. Далі нам потрібно перевірити створенні VRF. Переконавшись в правильному налаштуванні ми робимо висновок, що графічний інтерфейс працює вірно. (рис.3.7), (рис.3.8), (рис.3.9).

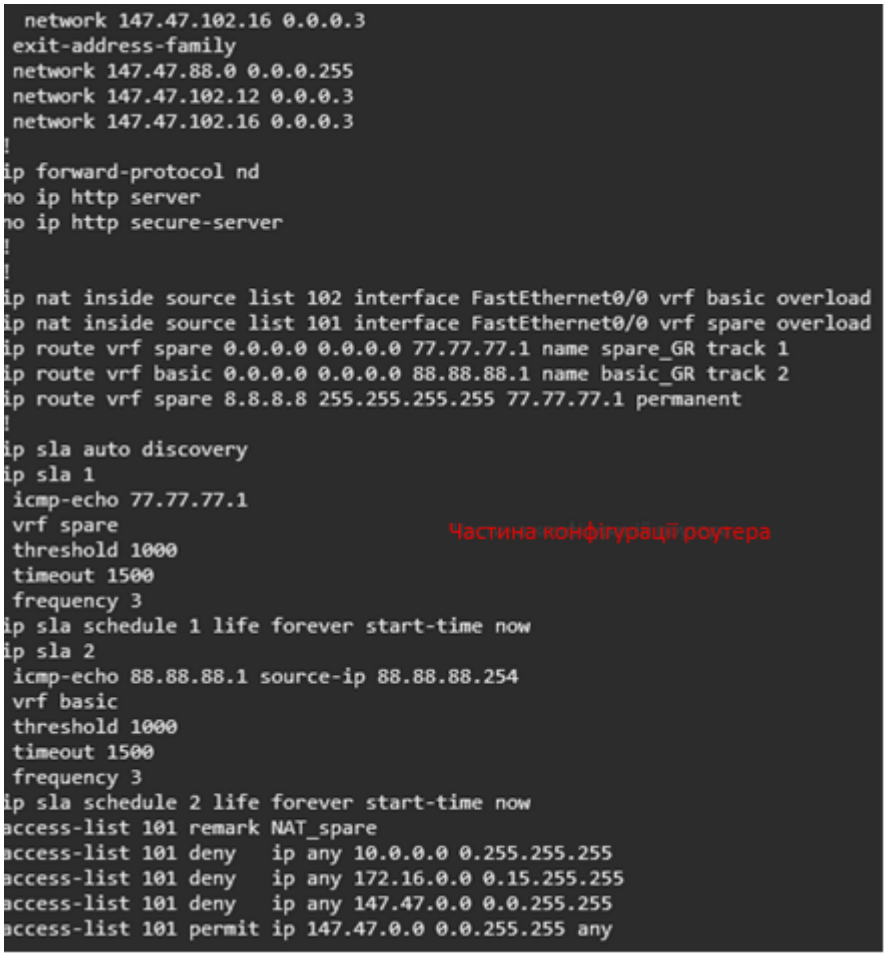

Рисунок 3.7 – Результат використання команди «sh run»

Використаємо команду «sh ip route» для відображення поточного стану таблиці маршрутизації, як можемо бачити таблиця заповнена. Також бачимо, що базовим маршрутом являється «Tunnel21» (рис.3.8).

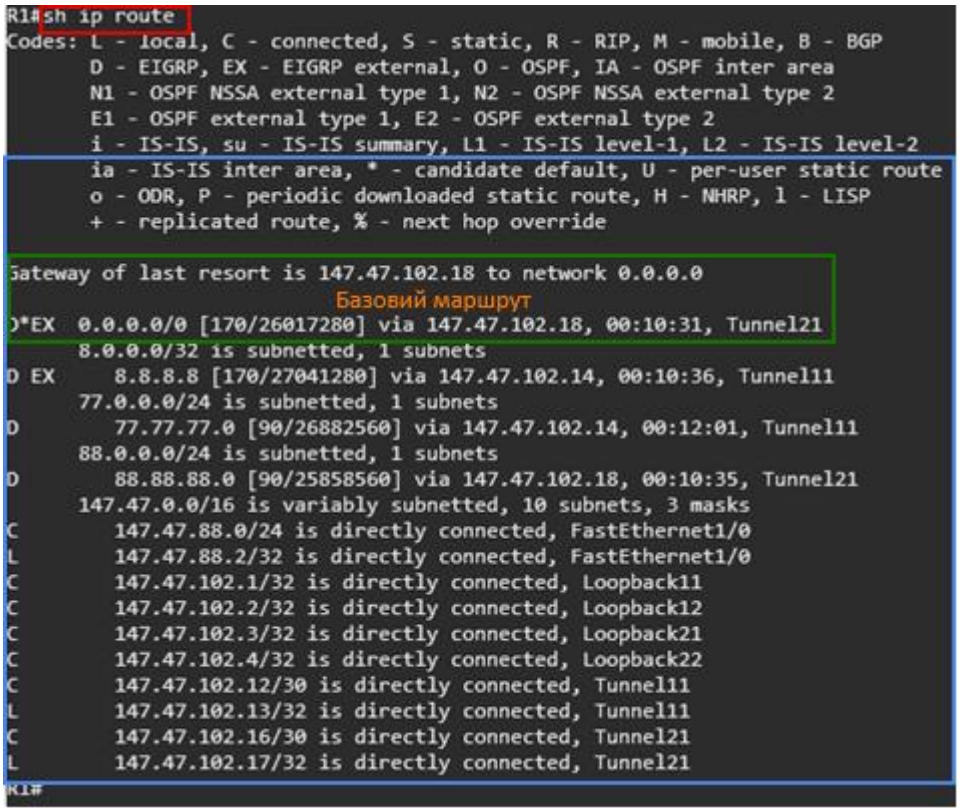

Рисунок 3.8 – Результат використання команди «sh ip route»

Настав час перевірити чи працюють команди перекидання пакетів між провайдерами при відключенні основного провайдера (рис.3.9).

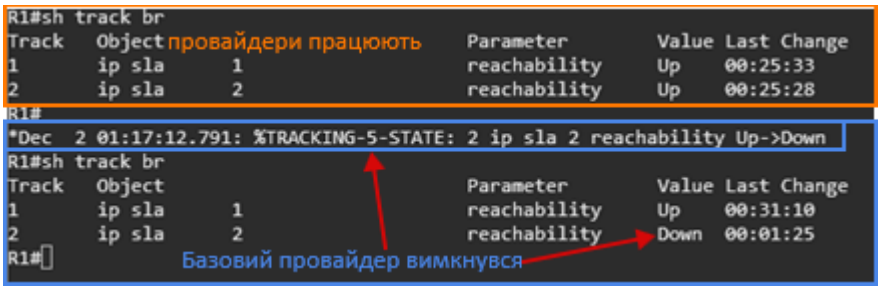

Рисунок 3.8 – Результат використання команди «sh track br», до та після вимкнення базового провайдера

Після цього знову звернемося до команди «sh ip route». Тепер бачимо, що маршрут змінився з «Tunnel21» на «Tunnel11» це свідчить про те, що роутер після відключення базового провайдера буде відсилати трафік на резервного провайдера (рис.3.9).

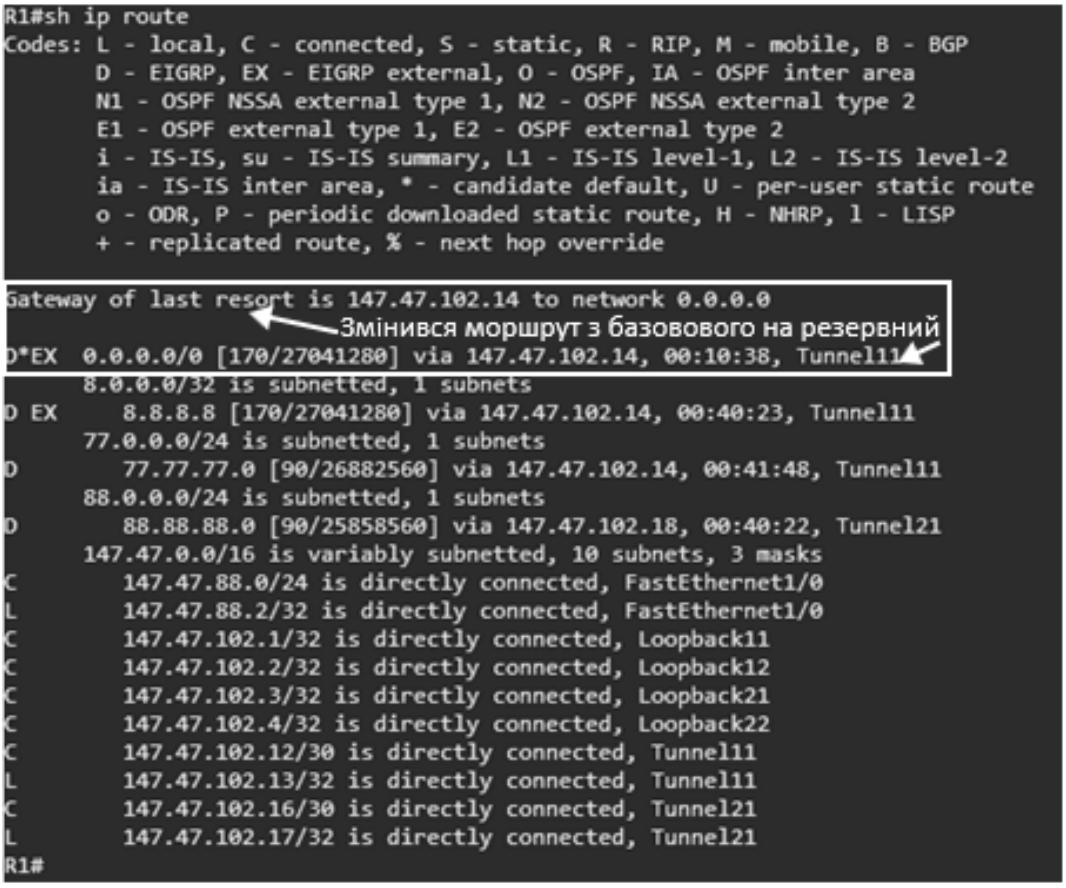

Рисунок 3.9 – Результат використання команди «sh ip route», до та після вимкнення базового провайдера

На усі налаштування за допомогою графічного інтерфейсу може піти 5-10 хвилин в залежності від опиту користувача, це швидше ніж вводити конфігурацію власноруч.

ВИСНОВКИ

Технологія VRF-Lite, дозволяє без істотних фінансових вкладень реалізувати захист від перебоїв підключення до інтернету.

Налаштування обраної технології в емуляторах або симуляторах, таких як GNS3, Cisco Packet Tracer, або на фізичному обладнанні є важкою, дорогою та довгою задачею яка потребує знать команд налаштування технології. Зокрема недоліком симуляторів є відсутність інтуїтивно зрозумілого графічного інтерфейсу для конфігурування віртуальної маршрутизації , що робить налаштування складним та довгим для початківця. Тому було прийняте рішення в потребі веб-орієнтовної програми, її графічний інтерфейс дозволяє згенерувати команди налаштування технологію VRF-Lite на маршрутизаторах. Від користувача буде потрібно заповнити поля IP-адресу та маску і отримати команди налаштування, які можна скопіювати і перенести в емулятор GNS3 або на реальне обладнання. В результаті буде налаштований маршрутизатор, який буде виконувати завдання безперебійного підключення до Internet.

Інтерфейс дає можливість користувачам початківцям налаштовувати технологію, не вимагаючи від них поглиблених знань команд конфігурації роутерів. Розроблений інтерфейс може допомогти, як початківцям так і фахівцям, значно пришвидшити налаштування як в симуляторах, емуляторах так і на реальному обладнанні.

СПИСОК ЛІТЕРАТУРИ

1. Cisco. Готовимся к сертификации cisco.: URL: http://ciscotips.ru/vrf.
2. Cisco. Виртуальные маршрутизаторы: URL: https://www.cisco.com/c/en/us/td/docs/security/firepower/660/fdm/fptd-fdm-config-guide-660/fptd-fdm-virtual-routers.html#id_99432.
3. Анатолий, Б. VRF-Lite теория и практика: URL: https://mum.mikrotik.com/presentations/RU17M/presentation_4939_1510118918.pdf.
4. Virtual Routing and Forwarding (VRF): URL: https://campus.barracuda.com/product/cloudgenfirewall/doc/74549106/virtual-routing-and-forwarding-vrf/.
5. Using nat on cisco router: URL: https://deltaconfig.com/using-nat/.
6. Cisco. Network Address Translation (NAT) FAQ: URL: https://www.cisco.com/c/en/us/support/docs/ip/network-address-translation-nat/26704-nat-faq-00.html.
7. Frankel, S., Kent, K., Lewkowski, R., та ін. Guide to IPsec VPNs: 2019. 55с.
8. Н. Олифер, С. О. Компьютерные сети принципы, технологии,протоколы: / за ред. Питер. Питер, 2016. 616с.
9. Б.Ю. Жураковський, І. О. З. Комп’ютерні мережі навчальний посібник для виконання лабораторних робіт: 2020. 166–168с.
10. Наташа Самойленко. EIGRP: URL: http://xgu.ru/wiki/EIGRP.
11. Г.Г. Киричек. Технології проектування телекомунікаційних мереж. Реалізація GRE тунелю: 2020. 4с.
12. Готовимся к сертификации cisco.: URL: http://ciscotips.ru/gre.
13. Cisco. Enhanced Interior Gateway Routing Protocol: URL: https://www.cisco.com/c/en/us/support/docs/ip/enhanced-interior-gateway-routing-protocol-eigrp/16406-eigrp-toc.html.
14. Інфокомунікації – сучасність та майбутнє: URL: https://onat.edu.ua/wp-

content/uploads/2018/04/сб2_с3_2017.pdf.

15. Мамойленко, С. Н. Среда моделирования GNS3: .
16. Графічний симулятор мережі gns3.: URL: https://pcuawiki.ru/rizne/9170-grafichnij-simuljator-merezhi-gns3.html.
17. Cooper051. Основы GNS3. Обзор: URL: https://habr.com/ru/post/266503/.
18. Ales999. VRF-Lite: URL: https://ales999.livejournal.com/28606.html?utm_source=3userpost.
19. Cisco. Description (interface): URL: https://www.cisco.com/c/m/en_us/techdoc/dc/reference/cli/nxos/commands/l2/description-interface.html#:~:text=The description command is meant,Ethernet interface.
20. Route Distinguisher: URL: https://habr.com/ru/sandbox/99255/.
21. Cisco. Configuring VRF-lite: URL: https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst4500/12-2/31sg/configuration/guide/conf/vrf.pdf.
22. Основы JavaScript: URL: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/JavaScript_basics.
23. JavaScript: URL: https://astwellsoft.com/blog/tehnology/javascript.html.

ДОДАТОК А

Конфігурація роутера ONE

```
conf t
interface FastEthernet0/0
description WAN_spare
ip address 40.40.40.254 255.255.255.0
speed auto
duplex auto
ex

int fa0/1
description WAN_basic
ip address 50.50.50.254 255.255.255.0
speed auto
duplex auto
ex

int fa1/0
description LocalLan
ip address 192.168.20.2 255.255.255.0
duplex full
no sh
ex

conf t
vrf defi spare
rd 65000:1
address-family ipv4
exit-address-family

vrf definition basic
rd 65000:2
address-family ipv4
exit-address-family
ex


conf t
interface FastEthernet0/0
vrf forwarding spare
ip address 40.40.40.254 255.255.255.0
no sh
exit
interface FastEthernet0/1
vrf forwarding basic
ip address 50.50.50.254 255.255.255.0
no sh
exit
end
 conf t
 interface Loopback11
 ip address 192.168.102.1 255.255.255.255
 interface Loopback12
 ip address 192.168.102.2 255.255.255.255
 ex
```

```
interface Tunnel11
ip address 192.168.102.13 255.255.255.252
ip mtu 1500
tunnel source 192.168.102.1
tunnel destination 192.168.102.2
ex

interface Tunnel12
vrf forwarding spare
ip address 192.168.102.14 255.255.255.252
ip mtu 1500
tunnel source 192.168.102.2
tunnel destination 192.168.102.1
ex

conf t
router eigrp 10
address-family ipv4 vrf spare autonomous-system 10
default-metric 16000 630 254 1 1500
redistribute connected
redistribute static
network 0.0.0.0
network 192.168.102.12 0.0.0.3
exit-address-family

network 192.168.102.12 0.0.0.3
network 192.168.20.0 0.0.0.255
end

conf t
ip route vrf spare 0.0.0.0 0.0.0.0 40.40.40.1 name spare_GR
end

conf t
int Fa0/0
ip nat outside
exit
int tu12
ip nat inside
exit

access-list 101 remark NAT_spare
access-list 101 deny ip any 10.0.0.0 0.255.255.255
access-list 101 deny ip any 172.16.0.0 0.15.255.255
access-list 101 deny ip any 192.168.0.0 0.0.255.255
access-list 101 permit ip 192.168.0.0 0.0.255.255 any
ip nat inside source list 101 interface Fa0/0  vrf spare overload
end

sh ip nat stat

conf t
interface Loopback21
ip address 192.168.102.3 255.255.255.255
interface Loopback22
ip address 192.168.102.4 255.255.255.255
interface Tunnel21
ip address 192.168.102.17 255.255.255.252
ip mtu 1500
tunnel source Loopback21
tunnel destination 192.168.102.4
interface Tunnel22
vrf forwarding basic
ip address 192.168.102.18 255.255.255.252
ip mtu 1500
tunnel source 192.168.102.4
tunnel destination 192.168.102.3
exit
router eigrp 10
address-family ipv4 vrf basic autonomous-system 10
default-metric 16000 630 254 1 1500
redistribute connected
redistribute static
network 0.0.0.0
network 192.168.102.16 0.0.0.3
exit-address-family
network 192.168.102.16 0.0.0.3
exit
```

```
ip route vrf basic 0.0.0.0 0.0.0.0 50.50.50.1 name basic_GR
access-list 102 remark NAT_basic
access-list 102 deny   ip any 10.0.0.0 0.255.255.255
access-list 102 deny   ip any 172.16.0.0 0.15.255.255
access-list 102 deny   ip any 192.168.0.0 0.0.255.255
access-list 102 permit ip 192.168.0.0 0.0.255.255 any

int FastEthernet0/1
ip nat outside
int tunnel 22
ip nat inside
ip nat inside source list 102 interface FastEthernet0/0 vrf basic overload
exit

conf t
int tu11
delay 5000
int tu21
delay 1000
end

conf t
ip sla 1
icmp-echo 40.40.40.1
vrf spare
threshold 1000
timeout 1500
frequency 3
exit

conf t
ip sla schedule 1 life forever start-time now
ex

ip route vrf spare 8.8.8.8 255.255.255.255 40.40.40.1 permanent
track 1 ip sla 1 reachability
exit

ip sla 2
icmp-echo 50.50.50.1 source-ip 50.50.50.254
vrf basic
threshold 1000
timeout 1500
frequency 3
exit

ip sla schedule 2 life forever start-time now
track 2 ip sla 2 reachability
ex
end

conf t
no ip route vrf spare 0.0.0.0 0.0.0.0 40.40.40.1 name spare_GR
no ip route vrf basic 0.0.0.0 0.0.0.0 50.50.50.1 name basic_GR
ip route vrf spare 0.0.0.0 0.0.0.0 40.40.40.1 name spare_GR track 1
ip route vrf basic 0.0.0.0 0.0.0.0 50.50.50.1 name basic_GR track 2
end
```

ДОДАТОК Б

Index.html

```
html>
      <head>
      <script src="https://cdn.jsdelivr.net/npm/vue@2.6.12"></script>
      <link rel="preconnect" href="https://fonts.gstatic.com">
      <link
href="https://fonts.googleapis.com/css2?family=Raleway:wght@500;700&display=s
wap" rel="stylesheet">
      <link rel="stylesheet" href="css/style.css">
      </head>
      <body>
            <div id="app">
                  <h1   class="title_prog">Налаштування   маршрутизації   VRF-
Lite</h1>
                  <section class="map" v-bind:style="{ backgroundImage: 'url('
+ bgimage + ')' }">
                        <div class="ip_mask spare">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.spare.ip">
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.spare.mask">
                              </div>
                        </div>
                        <div class="ip_mask basic">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.basic.ip">
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.basic.mask">
                              </div>
                        </div>
                        <div class="ip_mask int_fa_01">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_01.ip">
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_01.mask">
                              </div>
                        </div>
                        <div class="ip_mask int_fa_00">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"         name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_00.ip">
```

```
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"          name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_00.mask">
                              </div>
                        </div>
                        <div class="ip_mask int_fa_10">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"          name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_10.ip">
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"          name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.int_fa_10.mask">
                              </div>
                        </div>
                        <div class="ip_mask local-lan">
                              <div>
                                    <p class="label">IP </p>
                                    <input         type="text"          name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.local_lan.ip">
                              </div>
                              <div>
                                    <p class="label">Mask </p>
                                    <input         type="text"          name=""
placeholder="xxx.xxx.xxx.xxx" v-model="custom.local_lan.mask">
                              </div>
                        </div>
                  </section>

            <section class="config">
                  <div>
                        <input        class="button_conf"        type="button"
value="Заповнити поля" v-on:click="setDefault">
                        <input        class="button_conf"        type="button"
value="Конфігурація" v-on:click="checkData">
                        <input        class="button_conf"        type="button"
value="Копіювати" v-on:click="copyConfig">
                        <input class="button_conf" type="button" value="Зброс
Налаштувань" v-on:click="setNull">
                  </div>
                  <div class="result">
                        <p class="result_title">Конфігурація</p>
                        <textarea  id='result'  v-model="result"  type="text"
rows="25" cols="80" readonly></textarea>

                  </div>
            </section>

      </body>
</html>
<script src="js/script.js"></script>
```

Script.js

```
var vm = new Vue({
      el: '#app',
      data: {
            custom :{
                  spare : {
                        ip:"",
                        mask:""
                  },
                  basic : {
                        ip:"",
                        mask:""
                  },
                  int_fa_00 : {
                        ip:"",
                        mask:""
                  },
                  int_fa_01 : {
                        ip:"",
                        mask:""
                  },
                  int_fa_10 : {
                        ip:"",
                        mask:""
                  },
                  local_lan : {
                        ip:"",
                        mask:""
                  },
            },
            empty :{
                  spare : {
                        ip:"",
                        mask:""
                  },
                  basic : {
                        ip:"",
                        mask:""
                  },
                  int_fa_00 : {
                        ip:"",
                        mask:""
                  },
                  int_fa_01 : {
                        ip:"",
                        mask:""
                  },
                  int_fa_10 : {
                        ip:"",
                        mask:""
                  },
                  local_lan : {
                        ip:"",
                        mask:""
                  },
            },
            default :{
                  spare : {
                        ip:"40.40.40.1",
```

```
                        mask:"255.255.255.0"
                  },
                  basic : {
                        ip:"50.50.50.1",
                        mask:"255.255.255.0"
                  },
                  int_fa_00 : {
                        ip:"50.50.50.254",
                        mask:"255.255.255.0"
                  },
                  int_fa_01 : {
                        ip:"40.40.40.254",
                        mask:"255.255.255.0"
                  },
                  int_fa_10 : {
                        ip:"192.168.20.2",
                        mask:"255.255.255.0"
                  },
                  local_lan : {
                        ip:"192.168.20.1",
                        mask:"255.255.255.0"
                  },
            },
            valid:false,
            result:"",
            bgimage:'image/map_denied.JPG'
      },
      methods:{
            setDefault: function(event){
                  this.$data.custom                                                 =
JSON.parse(JSON.stringify(this.$data.default));
            },
            setNull: function(event){
                  this.$data.custom                                                 =
JSON.parse(JSON.stringify(this.$data.empty));
                  this.$data.result = ""
                  this.$data.valid = false
            },
            fillConfig: function(){
                  const config = `conf t        \n\
interface FastEthernet0/0\n\
description WAN_spare\n\
ip                  address                  ${this.$data.custom.int_fa_01.ip}
${this.$data.custom.int_fa_01.mask}\n\
speed auto\n\
duplex auto\n\
ex \n\
            \n\
int fa0/1\n\
description WAN_basic\n\
ip                  address                  ${this.$data.custom.int_fa_00.ip}
${this.$data.custom.int_fa_00.mask}\n\
speed auto\n\
duplex auto\n\
ex\n\
            \n\
int fa1/0\n\
description LocalLan\n\
ip                  address                  ${this.$data.custom.int_fa_10.ip}
${this.$data.custom.int_fa_10.mask}\n\
duplex full\n\
no sh\n\
```

```
ex\n\
\n\
vrf defi spare\n\
rd 65000:1\n\
address-family ipv4\n\
exit-address-family\n\
ex\n\
\n\
vrf definition basic\n\
rd 65000:2      \n\
address-family ipv4\n\
exit-address-family\n\
ex\n\
\n\
interface FastEthernet0/0\n\
vrf forwarding spare\n\
ip                  address                  ${this.$data.custom.int_fa_01.ip}
${this.$data.custom.int_fa_01.mask}\n\
no sh\n\
ex\n\
\n\
interface FastEthernet0/1\n\
vrf forwarding basic                \n\
ip                  address                  ${this.$data.custom.int_fa_00.ip}
${this.$data.custom.int_fa_00.mask}\n\
no sh\n\
exit\n\
\n\
interface Loopback11\n\
ip                                                                          address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.1
255.255.255.255\n\
interface Loopback12\n\
ip                                                                          address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.2
255.255.255.255\n\
ex\n\
\n\
interface Tunnel11\n\
ip                                                                          address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.13
255.255.255.252\n\
ip mtu 1500\n\
tunnel                                                                       source
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.1\n\
tunnel                                                                    destination
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.2\n\
ex\n\
\n\
interface Tunnel12\n\
vrf forwarding spare\n\
ip                                                                          address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.14
255.255.255.252\n\
ip mtu 1500\n\
tunnel                                                                       source
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.2\n\
tunnel                                                                    destination
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.1\n\
ex\n\
\n\
router eigrp 10\n\
```

```
address-family ipv4 vrf spare autonomous-system 10\n\
default-metric 16000 630 254 1 1500\n\
redistribute connected\n\
redistribute static\n\
network 0.0.0.0\n\
network
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.12
0.0.0.3\n\
exit-address-family\n\
\n\
network
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.12
0.0.0.3\n\
\n\
network    ${this.$data.custom.local_lan.ip.split('.').slice(0,3).join('.')}.0
0.0.0.255\n\
ex\n\
\n\
ip  route  vrf  spare  0.0.0.0  0.0.0.0  ${this.$data.custom.spare.ip}  name
spare_GR\n\
\n\
int Fa0/0\n\
ip nat outside\n\
ex\n\
\n\
int tu12\n\
ip nat inside\n\
ex\n\
\n\
access-list 101 remark NAT_spare\n\
access-list 101 deny ip any 10.0.0.0 0.255.255.255      \n\
access-list 101 deny ip any 172.16.0.0 0.15.255.255      \n\
access-list             101             deny              ip              any
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.0.0
0.0.255.255\n\
access-list                  101                  permit                   ip
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.0.0
0.0.255.255 any\n\
ip nat inside source list 101 interface Fa0/0  vrf spare overload\n\
end\n\
\n\
conf t\n\
interface Loopback21\n\
ip                                                                  address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.3
255.255.255.255\n\
ex\n\
\n\
interface Loopback22\n\
ip                                                                  address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.4
255.255.255.255\n\
ex\n\
\n\
interface Tunnel21\n\
ip                                                                  address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.17
255.255.255.252\n\
ip mtu 1500\n\
tunnel source Loopback21\n\
tunnel                                                          destination
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.4\n\
```

```
ex\n\
\n\
interface Tunnel22\n\
vrf forwarding basic\n\
ip                                                                    address
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.18
255.255.255.252\n\
ip mtu 1500\n\
tunnel                                                                 source
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.4\n\
tunnel                                                              destination
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.3\n\
exit\n\
\n\
router eigrp 10\n\
address-family ipv4 vrf basic autonomous-system 10\n\
default-metric 16000 630 254 1 1500\n\
redistribute connected\n\
redistribute static\n\
network 0.0.0.0\n\
network
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.16
0.0.0.3\n\
exit-address-family\n\
network
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.102.16
0.0.0.3\n\
exit\n\
\n\
ip  route  vrf  basic  0.0.0.0  0.0.0.0  ${this.$data.custom.basic.ip}  name
basic_GR\n\
access-list 102 remark NAT_basic\n\
access-list 102 deny   ip any 10.0.0.0 0.255.255.255\n\
access-list 102 deny   ip any 172.16.0.0 0.15.255.255\n\
access-list         102         deny                              ip         any
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.0.0
0.0.255.255\n\
access-list                  102                  permit                   ip
${this.$data.custom.local_lan.ip.split('.').slice(0,2).join('.')}.0.0
0.0.255.255 any\n\
end\n\
\n\
conf t\n\
int FastEthernet0/1\n\
ip nat outside\n\
int tunnel 22\n\
ip nat inside\n\
ip nat inside source list 102 interface FastEthernet0/0 vrf basic overload\n\
exit\n\
\n\
conf t\n\
int tu11\n\
delay 5000    \n\
int tu21\n\
delay 1000\n\
end\n\
\n\
conf t\n\
ip sla 1\n\
icmp-echo ${this.$data.custom.spare.ip}          \n\
vrf spare\n\
threshold 1000\n\
```

```
timeout 1500\n\
frequency 3\n\
exit\n\
\n\
ip sla schedule 1 life forever start-time now\n\
\n\
ip route vrf spare 8.8.8.8 255.255.255.255 ${this.$data.custom.spare.ip}
permanent\n\
track 1 ip sla 1 reachability\n\
exit\n\
\n\
ip sla 2\n\
icmp-echo               ${this.$data.custom.basic.ip}               source-ip
${this.$data.custom.int_fa_00.ip} \n\
vrf basic\n\
threshold 1000\n\
timeout 1500\n\
frequency 3\n\
exit\n\
\n\
ip sla schedule 2 life forever start-time now\n\
track 2 ip sla 2 reachability\n\
end\n\
\n\
conf t\n\
no ip route vrf spare 0.0.0.0 0.0.0.0 ${this.$data.custom.spare.ip} name
spare_GR\n\
no ip route vrf basic 0.0.0.0 0.0.0.0 ${this.$data.custom.basic.ip} name
basic_GR\n\
ip route vrf spare 0.0.0.0 0.0.0.0 ${this.$data.custom.spare.ip} name
spare_GR track 1\n\
ip route vrf basic 0.0.0.0 0.0.0.0 ${this.$data.custom.basic.ip} name
basic_GR track 2\n\
end`
                  return config
            },
            checkData: function(event){
                  status = true
                  code = ""
                  for (var key in this.$data.custom){
                        if (status == "false"){
                              break
                        }
                        for (var field_key in this.$data.custom[key]){


      if(this.$data.custom[key][field_key].match(/^(25[0-5]|2[0-4][0-
9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-
4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$/)){
                              }else{
                                    status = false
                                    if  (this.$data.custom[key][field_key]  ==
""){
                                          code = "No data in "+ key + ":" +
field_key +". Fill all text inputs."
                                    }else{
                                          code = 'Error in ' + key + ' ' +
field_key
                                          break
                                    }

                              }
```

```
                        }
                    }
                if (status == "true"){
                        code = this.fillConfig()
                }
                this.$data.valid = status
                this.$data.result = code
                // this.$data.custom = this.$data.empty
            },
            copyConfig: function(){
                if (this.$data.valid){
                        var copyText = document.querySelector("#result");
                        copyText.select();
                        document.execCommand("copy");
                        document.getSelection().removeAllRanges();
                }

            }
        },
        watch:{
            'valid' : function(val, oldVal){
                if (val == 'true'){
                        this.$data.bgimage = 'image/map_access.JPG'
                }else{
                        this.$data.bgimage = 'image/map_denied.JPG'
                }
            }
        }
    })
```

Stayl.css

```
body {
    font-family: 'Raleway', sans-serif;
}

.title_prog {
    text-align: center;
    font-size: 22px;
    color: #2A3541;
    /* border-bottom: 2px dashed #009CD6; */
    text-decoration: underline
}
.map{
    background-repeat:  no-repeat;
    background-position: center;
    background-size: 480px 480px;
    width: 900px;
    height: 480px;
      position: relative;
    margin: 0 auto;
    transform: scale(0.8);

}
.ip_mask{
    position: absolute;
    width: 170px;

}

.ip_mask div{
```

```
    display: flex;
    flex-direction: row;
    justify-content:space-between;
    margin: 5px 0;
}


.ip_mask input{
    width: 120px;
    display: inline;
    float: right;
    margin: 0;
    border-radius: 15px;
    padding: 0 10px;
    border: 2px solid #000;
}

.ip_mask input:focus{
    outline: none;
    border: 2px solid #009CD6;
}

.ip_mask .label{
    margin: 0;
    display: inline-block;
}

.ip_mask.spare{
    left: 30px;
    top: 40px;
}

.ip_mask.basic{
    right: 30px;
    top: 40px;
}

.ip_mask.int_fa_00{
    right: 170px;
    top: 180px;
}

.ip_mask.int_fa_01{
    left: 200px;
    top: 180px;
}

.ip_mask.int_fa_10{
    left: 370px;
    top: 250px;
}

.ip_mask.local-lan{
    left: 30;
    bottom: 0;
}

.config{
    border-top: 2px dashed #009CD6;
    margin-top: 20px;
    padding: 20px;;
    display: flex;
```

```
    flex-direction: column;
    align-items: center;
    align-content:space-evenly;
}

.config div{
    margin: 5px;
}

.button_conf{
    background: #fff;
    text-transform: uppercase;
    font-weight: 700;
    color: #009CD6;
    border-radius: 50px;
    border: 2px solid #009CD6;
    padding: 10px;
    margin: 0 20px;
    outline: none;
}

.button_conf:hover{
    background: #009CD6;
    color: #fff;
    transition: all 0.5s;
}

.result{
    display: flex;
    flex-direction: column;
    align-items: center;
    padding: 20px;
    margin: 20px;
}

.result textarea{
    background: #fff;
    padding: 20px;
    white-space: pre-line;
    border: none;
    resize: none;
    outline: none;
}

.result-title{
    font-size: 1.2em;
    text-align: center;
    display: block;
}
```