МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Секція інформаційно-комунікаційних технологій

ВИПУСКНА РОБОТА

на тему:

«Графічний інтерфейс налаштування розширених списків контролю доступу на роутерах Cisco»

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

Керівник роботи — **Великодний Д.В.**

Студентка гр. ІН-71 — **Закірова О.І.**

СУМИ 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А.С.

"_____" _____2021 г.

ЗАВДАННЯ
до випускної роботи

Студентки четвертого курсу, групи ІН-71 спеціальності "Комп'ютерні науки" денної форми навчання, Закірової Олександри Ілгамівни.

Тема: "**Графічний інтерфейс налаштування розширених списків контролю доступу на роутерах Cisco**"

Затверджена наказом по СумДУ

№_____ от _____2021 г.

Зміст пояснювальної записки: 1) постановка задачі; 2) огляд функціональних можливостей списків контролю доступу; 3) конфігурація мережі на базі емулятора Cisco Packet Tracer; 5) розробка графічного інтерфейсу у вигляді веб-орієнтованої інформаційної системи; 6) тестування результатів роботи інтерфейсу.

Дата видачі завдання "_____" _____2020 г.

Керівник випускної роботи _____ Великодний Д.В.

Завдання прийняла до виконання _____ Закірова О.І.

РЕФЕРАТ

Записка: 55 стор., 35 рис., 1 додаток, 18 джерел.

Об’єкт дослідження — використання розширених списків контролю доступу у комп’ютерних мережах.

Мета роботи — розробка графічного інтерфейсу, який буде генерувати коди для налаштування розширених списків контролю доступу на роутерах мережі з використанням вхідних даних користувача.

Методи дослідження — конфігурація мережі в емуляторі Cisco Packet Tracer. Розробка графічного інтерфейсу та аналіз результатів його роботи в емуляторі.

Результати — розроблено графічний інтерфейс у вигляді веб-сторінки з використанням мови програмування JavaScript, а також технологій HTML, CSS. В інтерфейсі можливо згенерувати код налаштування розширених списків контролю доступу після введення вхідних даних. Інтерфейс було протестовано у емульованій комп’ютерній мережі в Cisco Packet Tracer.

СПИСКИ КОНТРОЛЮ ДОСТУПУ, РОУТЕР CISCO, ACL, CISCO PACKET TRACER, ГРАФІЧНИЙ ІНТЕРФЕЙС, КОМП’ЮТЕРНА МЕРЕЖА, JAVASCRIPT.

ЗМІСТ

ВСТУП

Починаючи з моменту створення телекомунікаційних мереж перед людьми постало питання їх безпечного взаємозв’язку. Існує велика кількість засобів та методів направлених на вирішення даної проблеми та забезпечення захисту інформації. Найпоширенішим варіантом можна вважати паролі, але також популярними є шифрування й криптографія даних, налаштування та контроль фізичної безпеки, апаратні та програмні застосунки.

Особливої уваги заслуговують списки контролю доступу, які виступали у якості мережевих фільтрів на інтерфейсах маршрутизаторів з того часу, як маршрутизатори з’явились. Хоча технологія списків доступу відноситься до базових, в даному випадку це не є недоліком, а навпаки дозволяє їм бути широко застосованими на практиці.

Списки контролю доступу є одними з найважливіших засобів організації базової безпеки локальних і розподілених IP-мереж. Вимога безпеки особливо актуальна в локальних корпоративних мережах, коли вони через прикордонні маршрутизатори пов'язані з відкритою глобальною мережею Internet.

У зв’язку з цим кожен мережевий адміністратор повинен вміти налаштовувати списки контролю доступу у власній мережі. Проте, даний процес може потребувати дуже багато часу, якщо людина не має достатньо досвіду використання подібних технологій. Вирішити цю проблему зможе графічний інтерфейс, який буде мати функціонал генерування кодів для маршрутизаторів Cisco.

Для того, щоб зробити це можливим, необхідно дослідити функціонал списків контролю доступу і створити веб-орієнтовану систему, якою зможуть користуватися як фахівці, так і початківці, з метою збереження часу та полегшення процесу забезпечення коректної поведінки мережі.

1. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Базові відомості про списки контролю доступу

Кажучи про списки контролю доступу, спеціалісти найчастіше за все використовують абревіатуру ACL – від англ. access control list. ACL – це список правил та об’єктів, який має на меті визначення певних портів служб, а також в деяких випадках імен доменів, доступних на вузлах або інших пристроях третього рівня OSI, яким надається або забороняється доступ до сервісу. Дана технологія є вбудованою функцією системи міжмережевого програмного забезпечення маршрутизаторів та комутаторів Cisco IOS (не плутати з iOS, яка є операційною системою мобільною продукції компанії Apple). ACL допомагають фільтрувати мережевий трафік, визначати та контролювати бізнес-політику мережі.

Хоча дана технологія являється одним з фундаментальних будівельних блоків конфігурації мережі, варто зазначити, що на відміну від багатьох інших функцій в системі IOS, ACL дійсно застаріли. Стандартні ACL, які перевіряють трафік, орієнтуючись на вихідну адреси Internet Protocol (IP), були ще частиною IOS 8.3. Для порівняння, IOS 9 була представлена в 1992 році, а в 2019 вийшла остання відома версія – IOS 15.8, яка використовується на обладнанні Cisco й досі [1].

Щоб зрозуміти в чому полягає важливість та зазвичай необхідність списків контролю доступу потрібно зрозуміти, як саме відбувається процес передачі даних у мережі. Отже, мережевий трафік проходить шлях у вигляді пакетів. Кожний пакет містить всередині невеликий фрагмент даних і усю необхідну інформацію, яка потребується для його доставки. За замовчуванням, коли маршрутизатор отримує пакет на інтерфейс, він має наступний алгоритм дій:

1. Отримати адресу призначення з пакету.
2. Знайти запис для адреси призначення в таблиці маршрутизації.
3. Якщо збіг знайдений, перенаправити пакет з відповідного інтерфейсу.
4. Якщо відповідність не знайдена, пакет негайно відкидається.

При цьому будь-який відправник, який знає правильну адресу призначення, може відправити свій пакет через маршрутизатор. Тому, якщо подібна поведінка системи не відповідає політиці компанії, мережевий адміністратор може встановити контроль над нею, налаштувавши контроль доступу на локальних маршрутизаторах. Після коректного налаштування будь-яка спроба проникнути в заборонені ресурси буде відхилена.

Контроль доступу в мережі може бути визначений на звичайному сервері, на маршрутизаторі, а такі технології, як брандмауери, маршрутизатори і будь-які пограничні технічні облаштування доступу навіть залежать від списків контролю доступу для того, щоб правильно функціонувати. При впровадженні ACL необхідно передбачити і реалізувати процедуру регулярного оновлення цих списків контролю доступу [2].

До речі, багато типів операційних систем реалізують списки ACL або мають історичну реалізацію подібного налаштування. Перший з яких був у файловій системі в Multics в 1965 р. [3].

1.2 Функціонал ACL

Операційна система маршрутизаторів від компанії Cisco Systems має потужні вбудовані засоби для створення різноманітних за складністю і функціональності списків доступу. Після перевірки та класифікації трафіку, ACL визначає, що з ним необхідно зробити. Існує велика кількість варіантів застосування списків контролю доступу. Серед них:

- На інтерфейсі: пакетна фільтрація
- На лінії Telnet: обмеження доступу до маршрутизатора
- VPN: який трафік потрібно шифрувати
- QoS: визначення пріоритету трафіку
- NAT: які адреси транслювати

Зупинимося на пакетній фільтрації, адже саме її я буду реалізовувати в моєму графічному інтерфейсі. Для даного типу фільтрації необхідно створити ACL незалежно, а потім розмістити на відповідному інтерфейсі маршрутизато-

ра. Одразу після виконання цієї операції почне відбуватися контроль трафіка, вхідного або вихідного, в залежності від того, який ви вказали у вашому списку [4].

Справа в тому, що списки доступу можуть фільтрувати трафік, що входить в маршрутизатор (in), так і трафік, що виходить із маршрутизатора (out). Напрямок трафіку вказується при розміщенні списку доступу на інтерфейсі.

Формат команди прив'язки списку до інтерфейсу наступний:

Router (config-if) # {протокол} access-group {номер} {in або out} [5]

Наприклад, ви хочете відправити певний запит з приватної мережі в ISP (Internet) (рис. 1.1). Спочатку маршрутизатор визначить, чи є ACL на інтерфейсі fa0/1. Якщо є, то обробка відправленого пакету буде вестись згідно з правилами цього списку доступу, в результаті йому або буде дозволено відправитись на вихідний інтерфейс fa0/0, або пакет буде видалено. Те ж саме на інтерфейсі fa0/0, якщо пакет успішно потрапив на нього, спочатку відбувається пошук ACL, у випадку його відсутності пакет відсилається в ISP, у випадку наявності, за результатом перевірки пакет або видаляється, або відсилається в ISP [4].

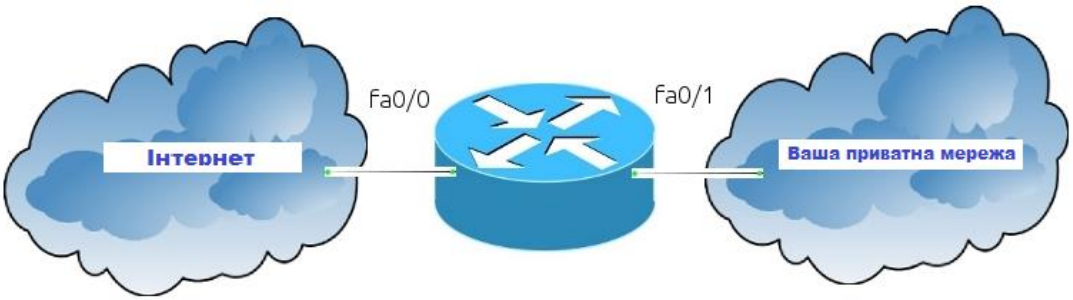

Рисунок 1.1 – Приклад мережевого зв'язку

Також на даному прикладі можна розглянути різницю між вхідним та вихідним ACL. Для того, щоб оптимально використовувати обчислювальні ресурси маршрутизатора необхідно розміщувати списки доступу в залежності від того, контроль якого трафіка ви маєте намір виконувати. Якщо ви хочете контролювати трафік, який надходить з ISP, перевіряючи його на безпечність, ACL потрібно розмістити на інтерфейсі fa0/0. В цьому випадку перевірка відбудеться один раз і пакет або буде визначений як безпечний, або видалиться.

На маршрутизаторі можуть функціонувати різні види мережевих протоколів, серед яких найпопулярніші IPv4, IPv6 або IPX, і списки контролю доступу ефективно працюють з будь-яким із них. Заборона або доступ мережевого трафіку забезпечуються на підставі аналізу співпадінь зі списком умов і правил, які адміністратор мав задати при налаштуванні ACL. Для цього списки доступу представляються у вигляді послідовних записів, в яких аналізуються один або декілька з наступних параметрів:

- адреса джерела,
- адреса призначення,
- використовуваний протокол,
- номер порту верхнього рівня (рис. 1.2)

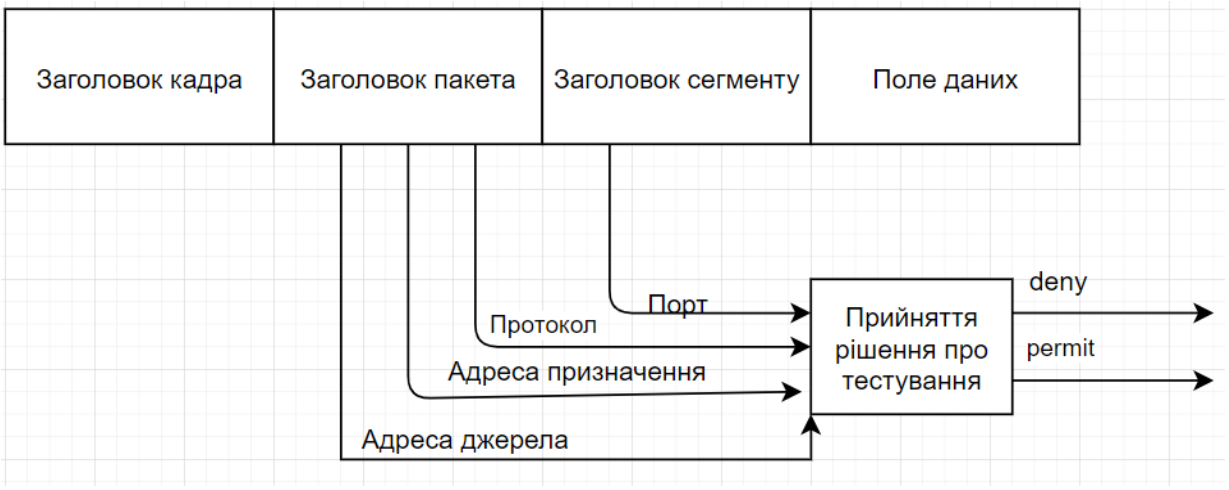

Рисунок 1.2 – Схема роботи списку контролю доступу на рівні пакету

Кожний інтерфейс маршрутизатора може містити окремий список доступу для вхідного та вихідного трафіків і для кожного встановленого мережевого протоколу. Наприклад, на трьох інтерфейсах маршрутизатора (рис. 1.3), з двома мережевими протоколами, IPv4 та IPv6, може бути створені 12 окремих списків доступу: шість для IPv4 і шість для IPv6. Тобто, на кожному інтерфейсі по 4 списки: 2 для того, що входить і 2 для вихідного трафіку.

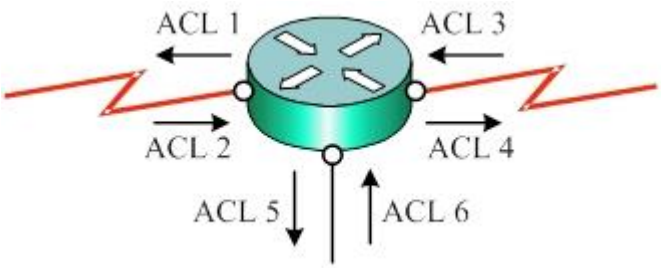

Рисунок 1.3 – Приклад розміщення ACL на інтерфейсах маршрутизатора

ACL також відповідають за гнучкість мережі. Існують списки правил, які зможуть обмежити відео-трафік у вашій мережі і цим самим зменшити навантаження, підвищуючи пропускну спроможність для передачі даних. Також завдяки подібному контролю можна заблокувати Telnet трафік, але дозволити маршрутизацію електронної пошти, яка може бути необхідна для роботи. Можна відслідковувати передачу файлів, наприклад, дозволяти або забороняти FTP та HTTP формати [5].

Дуже важливу роль в алгоритмі роботи списків доступу відіграє їх ієрархія. Синтаксично окреме правило списку - це один рядок. Для кожного пакету ACL проглядається заново, послідовно, починаючи з першого правила. Програмне забезпечення IOS Cisco перевіряє пакет по кожній умові. Якщо маршрутизатор зустріне умову, яка дозволяє пакету подальше просування, пакет буде відправлено далі по мережі, не зважаючи на те, чи є в подальших умовах його заборона, чи ні. Через це, існують спеціальні рекомендації щодо конфігурації списків контролю доступу. До речі, маршрутизатор не фільтрує пакети, які створив самостійно.

В кожному ACL в кінці є неявно задана за замовчуванням команда deny any. Вона означає заборону усього «іншого» трафіку. Отже в тому випадку, якщо в списку немає жодного правила, яке дозволяє один з видів трафіку, то весь трафік, який потраплятиме на даний інтерфейс, буде заблокований [6, 7].

Звичайним місцем розташування ACL є інтерфейси прикордонних маршрутизаторів. При такому місцезнаходженні ці пристрої стають міжмережевими екранами або брандмауерами (firewall), контролюючи непередбачений мереже-

вий трафік з Internet в корпоративну мережу, а також навпаки забезпечуючи захист периметра даної мережі [4, 6].

1.3 Ідентифікація списків контролю доступу

Для ідентифікації конкретного ACL всім його правилам присвоюється певний однаковий числовий номер (так звані "нумеровані" ACL) або символьне ім'я ("іменовані" ACL). Номер або ім'я використовуються для посилання на правила ACL як на єдиний об'єкт. Що стосується ідентифікаційних номерів створених списків доступу, вони повинні знаходитися в межах певного діапазону, заданого для цього типу списку (рис. 1.4) [5].

Діапазон номерів	Назва списку контролю доступу
1-99	IP standard access-list
100-199	IP extended access-list
1300-1999	IP standard access-list (extended range)
2000-2699	IP extended access-list (extended range)
600-699	Appletalk access-list
800-899	IPX standard access-list
900-999	IPX extended access-list

Рисунок 1.4 – Нумерація списків доступу в залежності від їх типу

Іменований варіант ідентифікації вважається кращим, так як він дозволяє потім редагувати створений список, в разі ж використання нумерованого способу, такий ACL при необхідності коректування можна тільки видалити повністю і потім заново створити, або також можна дописати черговий рядок в кінець списку умов. Не задовольняючий адміністратора список доступу повинен бути вилучений командою *no access-list* і потім створений заново.

Команди визначення списку доступу вводяться одна за одною, поки не буде сформований весь ACL. Він може складатися і з однієї команди.

При першому створенні ACL кожен запис списку доступу позначається порядковим номером, за замовчуванням в рамках десяти (10, 20, 30 і т.д). Завдяки чому, можна видалити певний запис і на її місце вставити інший, але ця

можливість з'явилася в Cisco IOS 12.3, до 12.3 доводилося ACL видаляти, а потім створювати заново повністю. Не можна розмістити більше одного списку на одному інтерфейсі, протоколі та напрямі одночасно [8].

Для фільтрації адрес в ACL використовується WildCard-маска. Це зворотна маска. Потрібно взяти шаблонний вираз: 255.255.255.255 і відняти від шаблону звичайну маску. Як, наприклад, 255.255.255.255-255.255.255.0, у нас виходить маска 0.0.0.255, що є звичайною маски 255.255.255.0, тільки 0.0.0.255 є WildCard маскою [4].

1.4 Види списків контролю доступу

Списки контролю доступу існують двох видів: стандартні і розширені. **Стандартний** вид дозволяє фільтрувати трафік тільки за одним критерієм: адреса відправника, в CCNA (Cisco Certified Network Associate) навіть розглядається конкретно тільки IP-адреса відправника. Подібна фільтрація не може дати велику кількість варіантів використання, але є важливою у тих випадках, коли потребується мінімальне використання ресурсів маршрутизатора. Можна, наприклад, поставити на виході з нашої мережі такий ACL:

access-list 1 permit host 192.168.10.50

access-list 1 permit host 192.168.10.53

access-list 1 permit host 192.168.10.60

Цей ACL буде дозволяти вихід в інтернет тільки з перерахованих в ньому трьох IP-адрес. Проте для такого завдання стандартного списку контролю доступу є більш ніж достатньо [9].

Загалом, стандартні списки контролю доступу використовуються для реалізації трьох типів політики:

- Доступ до ресурсів маршрутизатора.
- Розподіл маршруту
- Пакети, що проходять через маршрутизатор

Конфігурація списків доступу проводиться в два етапи:

1) створення списку доступу в режимі глобального конфігурування;

2) прив'язка списку доступу до інтерфейсу в режимі детальної конфігурації інтерфейсу [10].

Формат команди створення стандартного списку доступу наступний:

Router(config)# access-list {номер} {permit або deny} {адреса джерела}{маска адреси джерела}

В даному прикладі: номер ACL - будь-який номер з діапазону 1 ... 99 або 1300 ... 1999; адреса джерела - 32-бітова IP-адреса вузла або адреса підмережі; спец. маска адреси джерела - 32-бітова маска спеціального для ACL виду. У цій масці ланцюжок нулів зліва вказують на ті біти адреси відправника, які повинні обов'язково перевірятися на збіг з цими ж бітами в адресі відправника, а що залишилися одиниці маски вказують на ті біти, для яких збіг перевіряти не потрібно.

Пара адреса джерела та спеціальна маска цієї адреси утворюють адресний шаблон-правило для фільтрації пакетів на інтерфейсі. Якщо адреса відправника пакета збігається з адресним шаблоном, то виконується зазначена в команді операція – відповідно заборонити або дозволити. Якщо не збігається, то команда ігнорується і аналізується наступна по порядку команда з даного ACL. Введення спеціальної маски адреси дозволяє формувати адресний шаблон для фільтрації, як окремих IP-адрес, так і відразу групи адрес вузлів. Приклади адресних шаблонів:

- 192.168.15.22 0.0.0.0 - повинні перевірятися всі біти адреси на збіг з вказаною адресою.
- host 192.168.15.22 - інша дозволена і більш наочна форма запису шаблону, за дією подібна попередньої.
- 255.255.255.255 - не потрібно перевіряти ніякі біти на збіг, тобто правилом відповідає будь-якій адресі відправника.
- any - "всякий". Інша дозволена і більш наочна форма запису шаблону, за дією подібна до попередньої.

- 192.168.18.128 0.0.0.31 - з даними шаблоном порівнюються всі адреси в діапазоні 192.168.18.128 - 192.168.18.159. У справедливості твердження можна переконатися, представивши адресу і маску в двійковому коді.

Стандартний список доступу використовується для побудови наборів політик будь-якої IP-адреси або номер мережі. Як тільки набори політик визначені зі стандартним доступом списки, список доступу може обмежити доступ до мережевих ресурсів або Інтернету, визначити, які маршрути приймаються та як вони розподіляються, та змінити маршрутизацію метрики, що впливають на поведінку дорожнього руху.

Стандартні ACL - це найстаріший тип списків контролю доступу. Їх поява датується випуском ПО Cisco IOS Release 8.3. У всіх версіях програмного забезпечення номер access-list-number може варіюватися від 1 до 99. У ПО Cisco IOS Release 12.0.1 стандартні списки ACL почали використовувати додаткові номери (від 1300 до 1999). Ці додаткові номери відносяться до розширених списками ACL для IP. Програмне забезпечення Cisco IOS Release 11.2 надало можливість використовувати назву списку (поле name) в стандартних ACL.

Головною особливістю **розширених** списків контролю доступу та тим, що відрізняє їх від стандартних, є те, що вони дозволяють фільтрувати мережевий трафік по більшій кількості параметрів, ніж одна тільки адреса джерела. Окрім цього, фільтрація може відбуватися по таким критеріям, як:

- Адреса одержувача
- TCP / UDP порт відправника
- TCP / UDP порт одержувача
- Протокол, загорнутий в IP (відфільтрувати тільки TCP, тільки UDP, тільки ICMP, тільки GRE і т.п.)
- Тип трафіку для даного протоколу (наприклад, для ICMP відфільтрувати тільки ICMP-reply) [9].

Цей тип ACL також має змогу відокремлювати TCP-трафік, що йде в рамках встановленої TCP-сесії від TCP-сегментів, які тільки встановлюють з'єд-

нання. У зв’язку з розвинутими можливостями розширені списки доступу отримали додаткові технології, такі як:

- динамічні списки (dynamic ACL) – особливість їх полягає в тому, що деякі рядки списку до певного моменту не працюють, але вмикаються, коли адміністратор мережі підключається до маршрутизатора по telnet-зв’язку. Це часто використовується для безпеки мережі або виходу в інтернет.
- рефлексивні списки (reflexive ACL) - дзеркальні списки контролю доступу, вони дозволяють запам'ятовувати, хто звертався з нашої мережі назовні (з яких адрес, з яких портів, на які адреси, на які порти) і автоматично формувати дзеркальний ACL, який пропускатиме зворотний трафік ззовні всередину тільки в тому випадку, якщо зсередини було звернення до даного ресурсу.
- списки з часовим обмеженням (time-based ACL) – з назви зрозуміло, що ці списки працюють лише в певний проміжок часу, який вказується при їх створенні [9, 11].

Отже, розширені списки доступу можуть перевіряти набагато більше, ніж стандартні, а й працюють вони повільніше, так як доведеться заглядати всередину пакета, на відміну від стандартних де ми дивимося тільки поле адреси відправника.

Формат команди створення розширеного списку доступу наступний:

Router(config)# access-list {номер} {permit або deny} {протокол} {адреса джерела} {адреса призначення} {порт}

В поле протоколу задається ім'я або номер (0 - 255) протоколу мережі Інтернет. Найбільш часто використовуються протоколи IP, TCP, UDP, OSPF, RIP і ін. Поле порту використовується або для створення номера (0 - 65535), або - імені портів, наприклад, FTP або Telnet [5].

Формат команди прив'язки списку доступу до інтерфейсу аналогічний команді стандартного списку:

Router(config-if)# {протокол} access-group {номер} {in або out}

На відміну від стандартних списків, розширені списки фільтрують трафік більш "тонко". При створенні розширених списків в правилах доступу можна включати фільтрацію трафіку по протоколах і портам. Для вказівки портів в правилі доступу вказуються спеціальні позначення (рис 1.5). Наприклад, запис eq означає вимогу аналізу пакетів тільки з даним номером порту призначення. В свою чергу, neq означає вимогу аналізу пакетів з іншими номерами, за винятком даного. Запис range означає вимогу аналізу пакетів з номерами портів в зазначеному діапазоні [12].

Визначення	Дія, яка виконується
lt n	Все номера портов, меньшие n.
gt n	Все номера портов, большие n.
eq n	Порт n
neq n	Все порты, за исключением n.
range n m	Все порты от n до m включительно.

Рисунок 1.5 – Налаштування портів у ACL

Розширені ACL часто використовуються для брандмауерів та пограничних пристроїв, вказуючи на пакети, які можуть проходити через маршрутизатор між мережами різного ступеня довіри. Це дозволяє або забороняє передачу пакетів через маршрутизатор залежно від узгодження списків розширень.

Розширені списки ACL з'явилися в ПО Cisco IOS Release 8.3. У всіх версіях програмного забезпечення номер access-list-number може варіюватися від 101 до 199. У ПО Cisco IOS Release 12.0.1 в розширених списках ACL почали використовувати додаткові номери (від 2000 до 2699). Ці додаткові номери відносяться до розширених списками ACL для IP. Програмне забезпечення Cisco IOS Release 11.2 надало можливість використовувати назву списку (поле name) в розширених ACL [11, 13].

Існують спеціальні рекомендації щодо використання списків контролю доступу в вашій мережі, наприклад, стандартні списки доступу рекомендується

встановлювати по можливості ближче до адресата призначення, а розширені - ближче до джерела. Тобто стандартні списки доступу повинні блокувати пристрій або мережу призначення і розташовуватися ближче до мережі, що захищається, а розширені списки встановлюються ближче до можливого джерела небажаного трафіку.

Також загальною рекомендацією є те, що в рядках списків слід задавати умови фільтрації, починаючи від специфічних умов - до загальних. Тому що, список доступу виробляє фільтрацію пакетів по порядку. Умови списку доступу обробляються послідовно від вершини списку до основи, поки не буде знайдена відповідна умова. Якщо ніяка умова не знайдена, то тоді пакет відхиляється і знищується, оскільки неявне умова deny any (заборонити все інше) є неявно виражена в кінці будь-якого списку доступу. Якщо пакет протоколу IP не задовольняє списку доступу, він буде відхилений і знищений, при цьому відправнику буде надіслано повідомлення протоколу ICMP [14].

1.5 Постановка задачі

На основі розглянутої літератури та теоретичних відомостей по темі списків контролю доступу можна сказати, що основною метою даної кваліфікаційної наукової роботи є розробка веб-орієнтованої системи, функціонал якої надаватиме користувачу можливість оптимізувати процес налаштування розширених ACL на маршрутизаторах Cisco. Як результат успішної роботи програми, після введення відповідних даних, користувач зможе скопіювати автоматично згенерований код з графічного інтерфейсу в емулятор Packet Tracer, а також на реальне обладнання компанії Cisco.

Так як суть створення системи полягає в спрощенні процесу налаштування списків контролю доступу у мережі, графічний інтерфейс повинен бути інтуїтивно зрозумілим та простим у використанні навіть для людей, які тільки починають знайомство з телекомунікаційними мережами і не володіють усіма необхідними навичками для цієї спеціалізації.

Під час реалізації графічного інтерфейсу буде створена веб-сторінка, де буде зображена топологія мережі користувача, і будуть наявні поля для вводу вхідних даних та вибору необхідного типу розширених ACL. Якщо були введені валідні дані, система згенерує та виведе на екран код, який може бути використаний на обладнанні Cisco.

Постановка задачі:

1. За допомогою емулятора Packet Tracer створення базової топології мережі з налаштованою маршрутизацією.
2. Налаштування технологій розширених ACL в створеній мережі в емуляторі Packet Tracer.
3. Розробка графічного інтерфейсу для налаштування розширених ACL на обладнанні Cisco.
4. Тестування розробленого інтерфейсу в емуляторі Packet Tracer.

2. МОДЕЛЮВАННЯ МЕРЕЖІ З ВИКОРИСТАННЯМ ЕМУЛЯТОРА CISCO PACKET TRACER

2.1 Емулятор Cisco Packet Tracer

Packet Tracer - це відомий інструмент візуального моделювання. Він був розроблений компанією Cisco Systems, для того, щоб дозволити користувачам створювати мережеві топології і імітувати сучасні комп'ютерні мережі. Packet Tracer підтримується багатьма платформами, його можна запустити на Linux, Microsoft Windows і macOS. Також доступні аналогічні застосування для мобільних операційних систем Android і iOS [15].

Packet Tracer використовує призначений для користувача інтерфейс, що дозволяє користувачам додавати і видаляти змодельовані мережеві пристрої на власний розсуд. Фізичне з'єднання між пристроями представлене елементом "кабель". Packet Tracer підтримує безліч змодельованих протоколів прикладного рівня. На рис. 2.1 можна побачити стартове вікно програми.

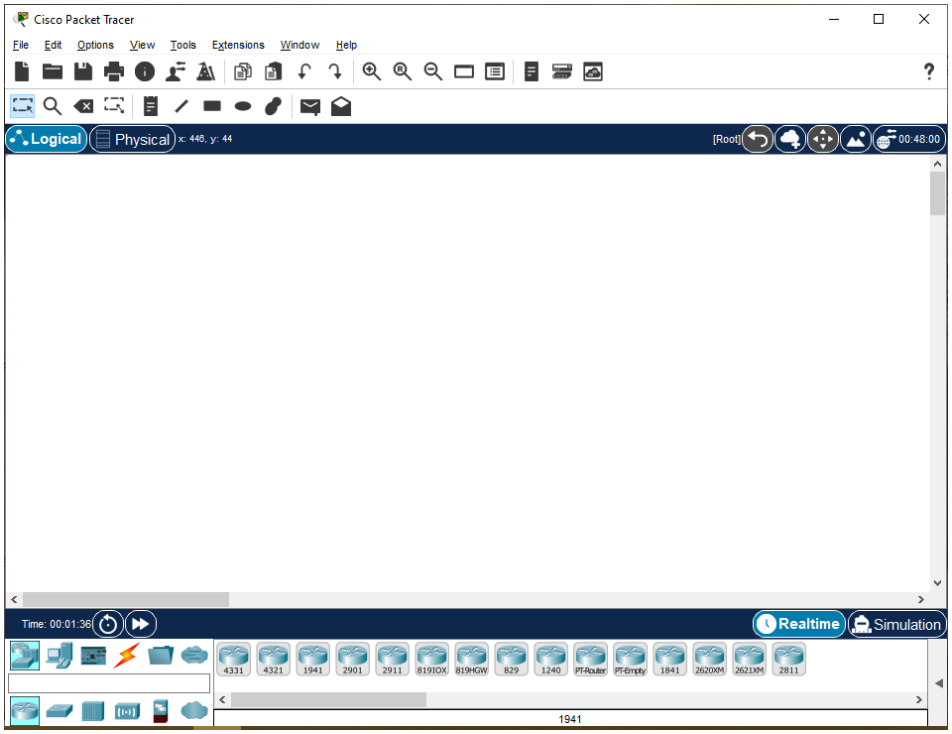

Рисунок 2.1 – Інтерфейс Cisco Packet Tracer

Внизу екрану знаходиться панель з вибором обладнання, яке ви можете додати до вашої топології. Серед них є велика різноманітність як звичайних

пристроїв, які є складовою будь-якої мережі, наприклад, маршрутизатори, комутатори, ноутбуки, комп’ютери, сервера, а також пристроїв, які є більш вузького напряму, наприклад ті, що можуть використовуватись в «розумному» домі, як детектори диму, вентилятори з дистанційним управлінням тощо.

Окрім моделювання певних аспектів комп'ютерних мереж, Packet Tracer також може використовуватися для спільної роботи. Починаючи з Packet Tracer 5.0, Packet Tracer підтримує розраховану на багато користувачів систему, яка дозволяє декільком користувачам сполучати декілька топологій в комп'ютерній мережі. Packet Tracer часто використовується в освітніх установах в якості навчального посібника, адже ця програма має функціонал створення завдань для студентів, з поясненням алгоритму їх виконання.

Варто усвідомлювати, що Packet Tracer є емулятором, а не симулятором мережі. Тобто він може продемонструвати роботу різних систем, взаємозв’язки між ними та особливості поведінки мережі в певних умовах, але не відтворює реальні ситуації. В цьому і полягає основна мета даної програми – надати користувачам змогу протестувати, як та чи інша конфігурація мережі може бути використана на практиці, імітуючи поведінку Cisco IOS. Це допомагає людям, відповідальним за формування мережі на реальному обладнанні, наприклад, мережевим адміністраторам компаній, уникнути ризиків, пов’язаних з фінансовими питанням, адже можна протестувати усе в режимі реального часу [16].

2.2 Створення топології мережі на базі емулятора Packet Tracer

Першим етапом підготовки до реалізації майбутнього графічного інтерфейсу є створення базової схеми, яка буде демонструвати роботу мережі та допоможе налаштувати поведінку системи, використовуючи імітацію програмного забезпечення Cisco IOS.

Було виконано конфігурацію наступної схеми (рис. 2.2). Дана система являє собою мережу уявної компанії з розробки програмного забезпечення. В компанії існує 5 пов’язаних між собою мереж: відділ розробки, тестування, менеджменту, DNS-сервер та серверна частина. Подібна конфігурація допоможе

пояснити взаємозв'язок між різними складовими системи і налаштувати поведінку мережі, надаючи реальні приклади.

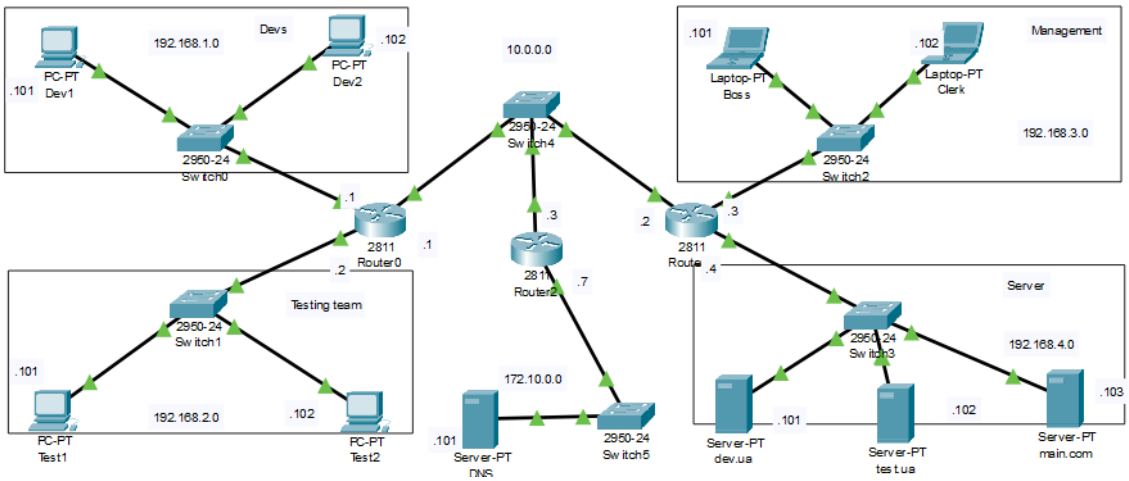

Рисунок 2.2 – Сконфігурована в емуляторі мережа

Отже, система складається з п'яти мереж. Відділ розробки (мережа 198.162.1.0) та тестування (мережа 192.168.2.0) використовують кожен по два комп'ютери, відділ менеджменту (мережа 192.168.3.0) складається з двух ноутбуків для голови компанії та для його секретаря відповідно. У серверній частині (мережа 192.168.4.0) знаходяться 3 сервера, які відповідають за кожний з відділів. Доступ на сервери відбувається з використанням технології DNS, сервер якої розташований у мережі 172.10.0.0. У кожного користувача та на серверах є налаштованими IP-адреси. Мережі пов'язані між собою шістьма комутаторами та трьома маршрутизаторами, на всіх з яких налаштована статична маршрутизація.

2.3. Конфігурація розширених списків доступу в емуляторі Cisco Packet Tracer

Для того, щоб можна було краще зрозуміти поведінку системи і в майбутньому порівняти результати самостійного налаштування розширених ACL з тим, як це буде відбуватися з використанням графічного інтерфейсу, було вирішено налаштувати найбільш поширені розширені списки контролю доступу, які в даній мережі могли би стати вирішенням конкретних проблем та задач.

Найпопулярнішими видами розширених списків контролю доступу в Cisco є:

- блокування трафіку host to host;
- блокування трафіку host to network;
- блокування трафіку network to host;
- блокування трафіку network to network;
- блокування пінгування але наявність дозволу доступу до сервера
- дозвіл та заборона ftp-трафіку

Почнемо з **заборони трафіку від одного вузла до іншого**. Цей вид блокування часто застосовується у випадках, коли, наприклад, наймають нового співробітника і йому ще рано мати доступ до критично важливих елементів та даних компанії. В нашому випадку, можна уявити ситуацію, в якій кожному робітнику з відділу розробки призначений один тестувальник (рис. 2.3).

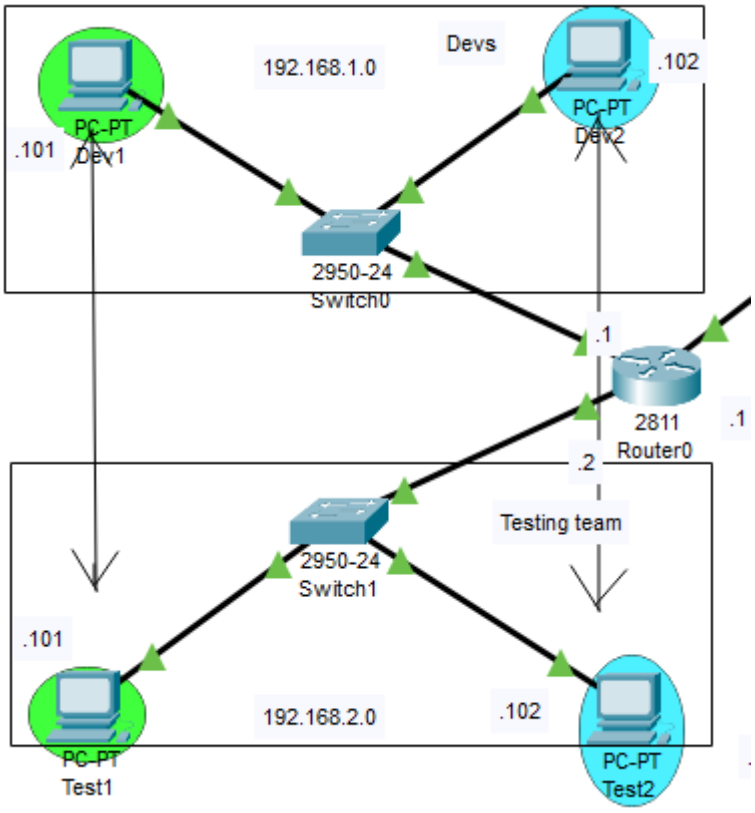

Рисунок 2.3 – Схема взаємозв'язку між розробниками та тестувальниками

Налаштування списку, який не дозволятиме розробникам мати доступ до вузла «чужого» тестувальника може допомогти керувати трафіком у системі. Додаємо наступний код на маршрутизатор 0.

```
Router>en
Router#conf t
Router(config)#access-list 101 deny icmp host 192.168.1.101 192.168.2.102 0.0.0.0
Router(config)#access-list 101 deny icmp host 192.168.1.102 192.168.2.101 0.0.0.0
Router(config)#access-list 101 permit icmp any any
Router(config)#int fa0/0
Router(config-if)#ip access-group 101 in
Router(config-if)#ex
```

Після розміщення списку контролю доступу 101 на інтерфейсі fa0/0 маршрутизатора 0 на вхідному трафіку, Dev1 має доступ лише до Test1, а Dev2 в свою чергу лише до Test2. Результат можна побачити використавши командну строку комп'ютера і команду ping в Packet Tracer (рис. 2.4, 2.5)

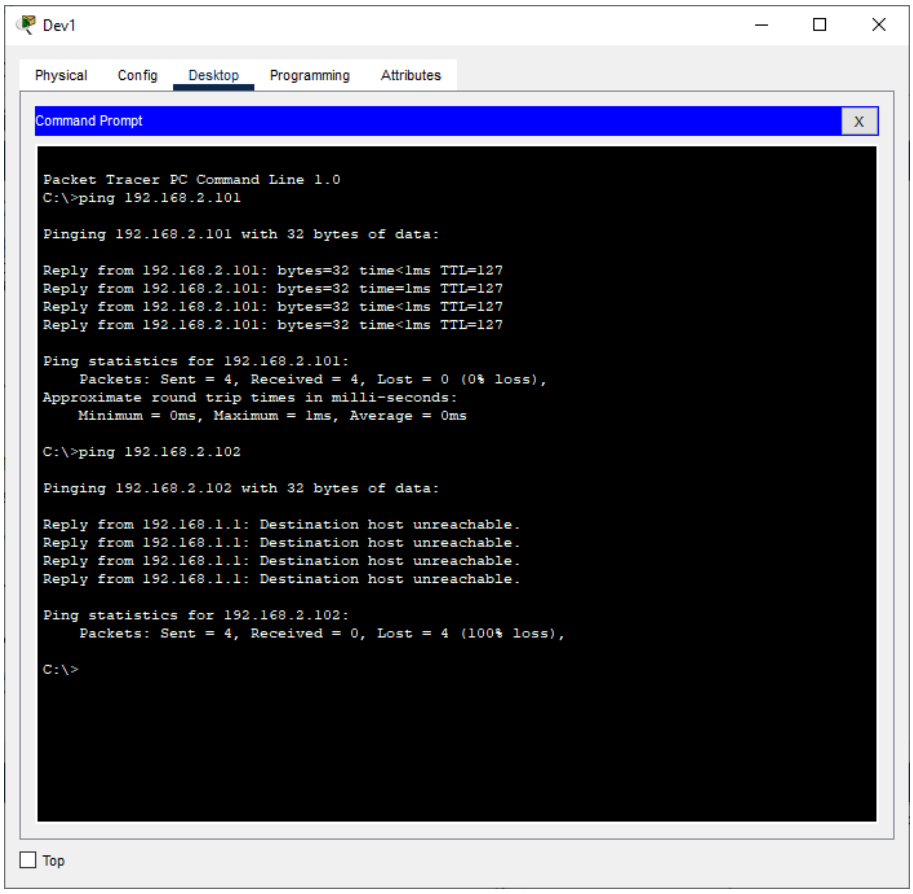

Рисунок 2.4 – Пінг з вузла Dev1 після налаштування

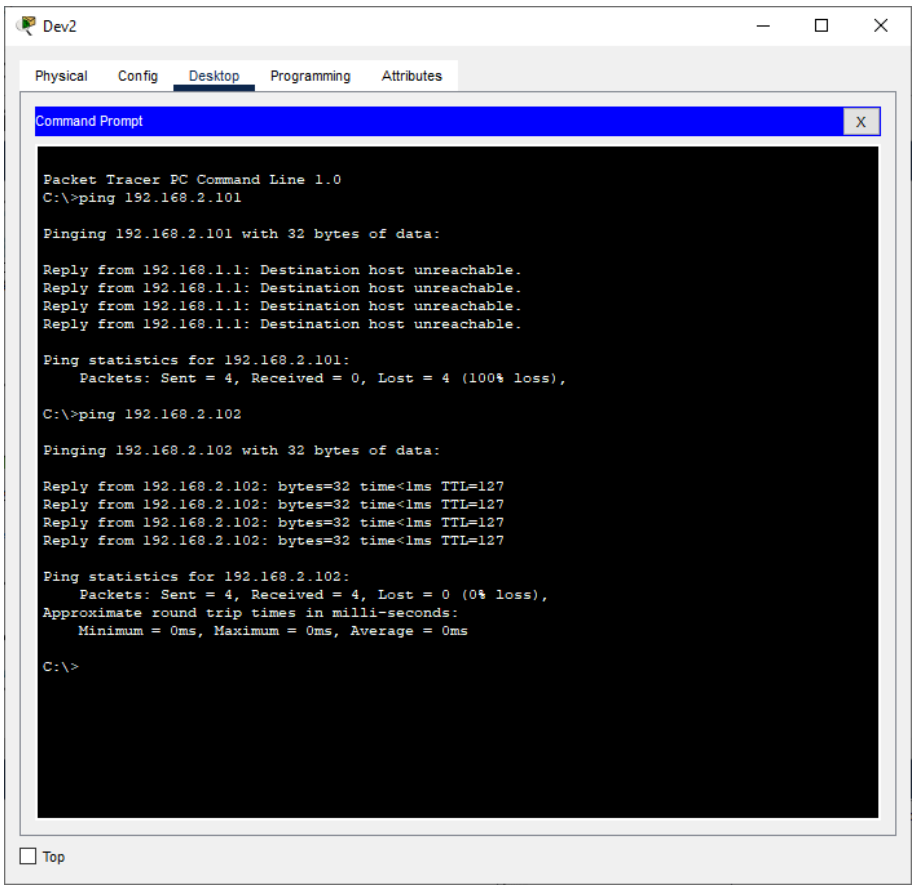

Рисунок 2.5 – Пінг з вузла Dev2 після налаштування

Наступний підвид розширених списків доступу, який ми розглянемо буде **блокування трафіку від одного вузла до іншої мережі** в цілому. Розглянемо схему, зображену на рисунку 2.6.

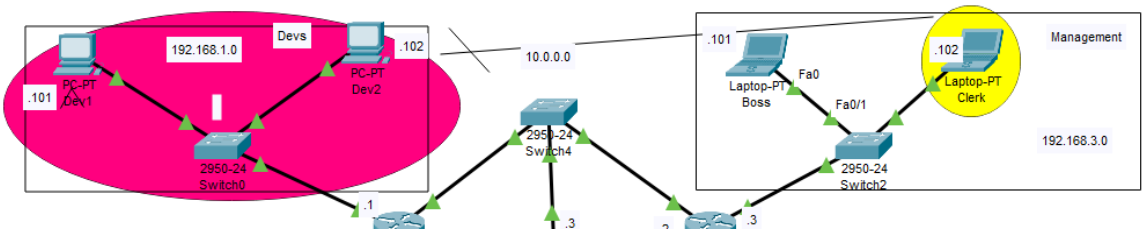

Рисунок 2.6 – Схема взаємозв'язку між секретарем та розробниками

Ми налаштуємо список доступу на маршрутизаторі 1 таким чином, що секретар голови компанії не буде мати можливість зв’язуватись напряму з відділом розробки, адже результати виконання роботи можна дізнатись безпосередньо у тестувальників.

```
Router>en
Router#conf t
```

```
Router(config)#access-list 102 deny icmp host 192.168.3.102 192.168.1.0 0.0.0.255
Router(config)#access-list 102 permit icmp any any
Router(config)#int fa0/0
Router(config-if)#ip access-group 102 in
Router(config-if)#ex
```

Список доступу номер 102 забороняє вузлу 192.168.3.102 направляти свій трафік до мережи 192.168.1.0. Використавши команду ping, бачимо успішний результат (рис. 2.7)

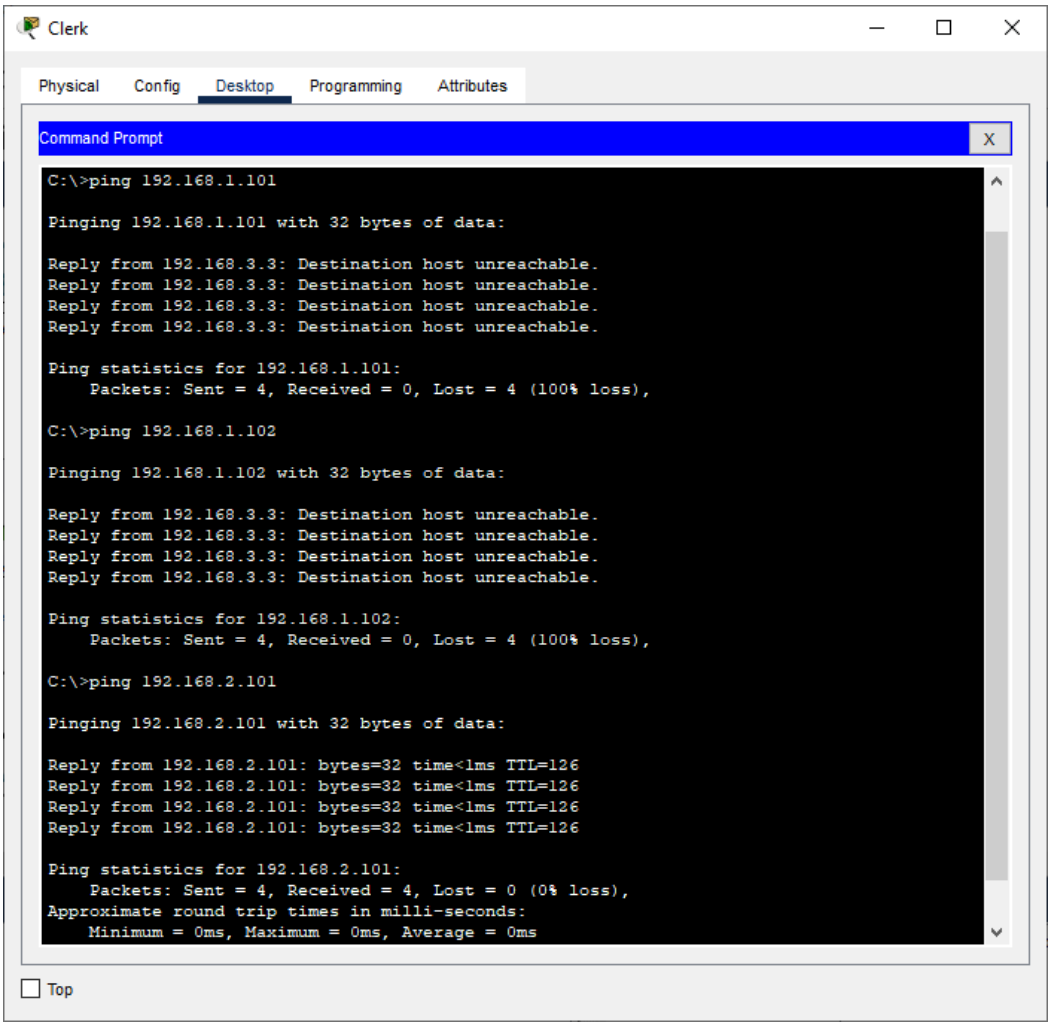

Рисунок 2.7 – Пінг з вузла Clerk у мережі розробників та тестувальників

Далі йде **блокування від мережі до вузла**. Мережі тестувальників не має сенсу мати доступ до серверів розробки та менеджменту. Проконтролювати це можна використовуючи третій вид розширених списків контролю доступу, суть якого полягає в блокуванні трафіку від мережі до вузла. Нехай відділ тестувальників має доступ лише до серверу test.ua.

```
Router>en
Router#conf t
Router(config)#access-list 103 deny icmp 192.168.2.0 0.0.0.255 192.168.4.101
0.0.0.0
Router(config)#access-list 103 deny icmp 192.168.2.0 0.0.0.255 192.168.4.103
0.0.0.0
Router(config)#access-list 103 permit icmp any any
Router(config)#int fa 0/1
Router(config-if)#ip access-group 103 in
Router(config-if)#ex
```

Завдяки розміщенню даного списку доступу на маршрутизаторі 0 мережа 192.168.2.0 матиме доступ лише до одного сервера, який знаходиться у мережі 192.168.4.0, до власного (рис. 2.8).

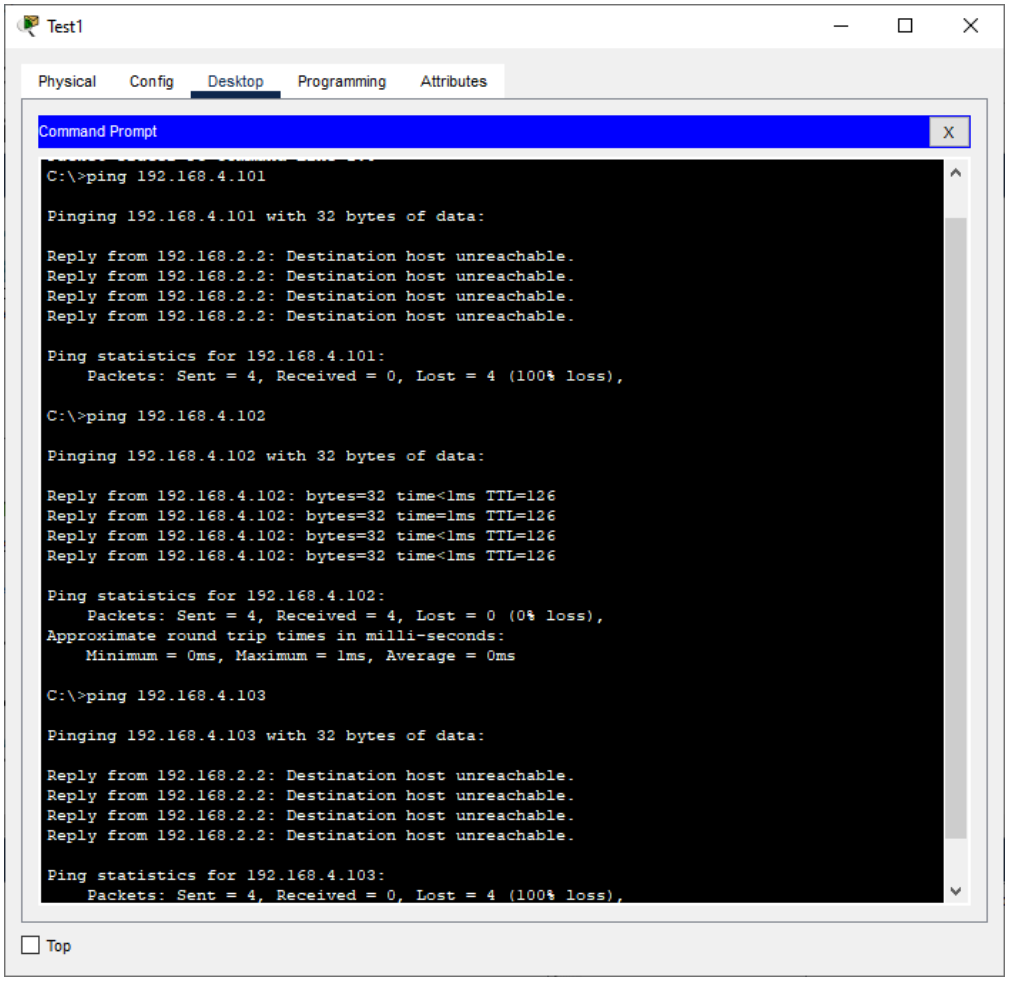

Рисунок 2.8 – З мережі тестувальників є доступ лише до власного сервера

Наступним, не менш важливим кроком у забезпеченні коректного використання трафіку є заборона відділам безпеки та тестування доступу до мережі менеджменту. Це буде представником **блокування мережі до мережі**. Так як склад працівників даних відділів може регулярно змінюватися, їм повинно бути відмовлено у доступі до мережі голови компанії задля безпеки зберігання важливої інформації.

```
Router>en
Router#conf t
Router(config)#access-list 109 deny icmp 192.168.1.0 0.0.0.255 192.168.3.0
0.0.0.255
Router(config)#access-list 109 permit icmp any any
Router(config)#int fa0/0
Router(config-if)#ip access-group 109 in
```

Перевіряємо, доступ до мережі менеджменту, використовуючи ping. Як результат – пакети були знищені.

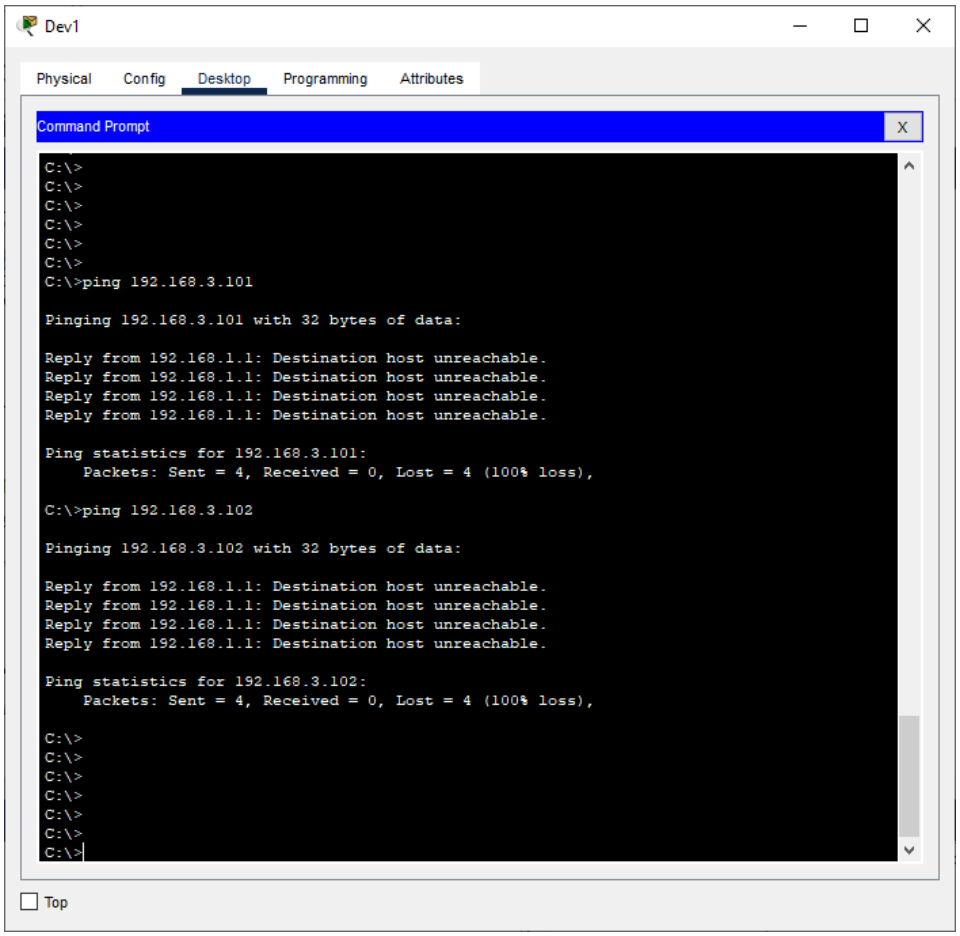

Рисунок 2.9 – Пінг з Dev до мережі менеджменту неможливий

Для того, щоб запобігти перевантаженню серверів необхідно **заборонити можливість надсилання на них ping-запитів, але залишити співробітникам доступ до серверів**. Це можна виконати, налаштувавши наступний список контролю доступу на роутері 1.

```
Router>en
Router#conf t
Router(config)#access-list 107 deny icmp any any echo
Router(config)#access-list 107 permit ip any any
Router(config)#int fa0/1
Router(config-if)#ip access-group 107 out
```

Як можна побачити, при спробі пінгування пакети були знищені, але доступ до серверу залишився (рис. 2.10, 2.11).

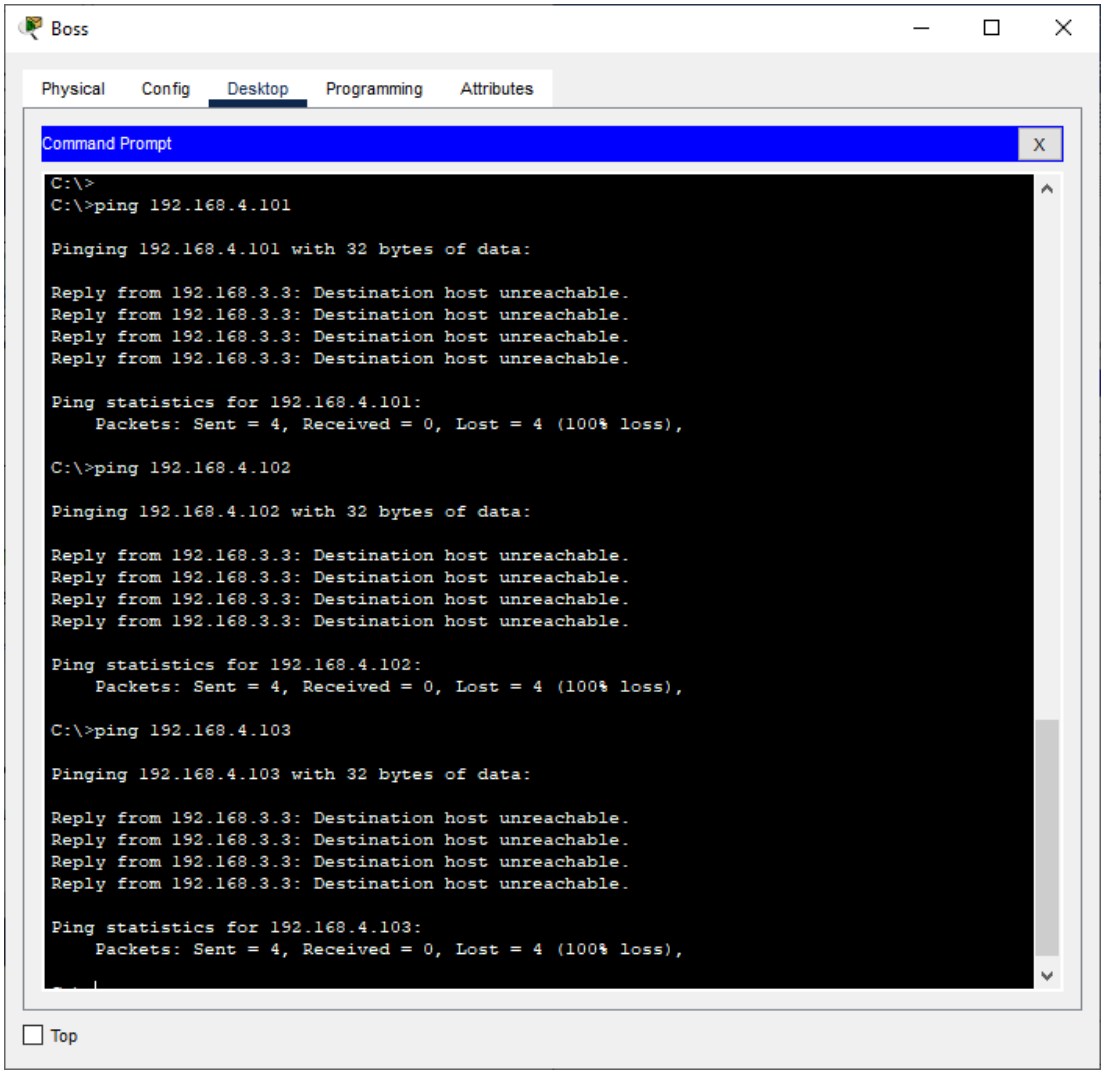

Рисунок 2.10 – Boss не може надсилати пінг-запити до серверів

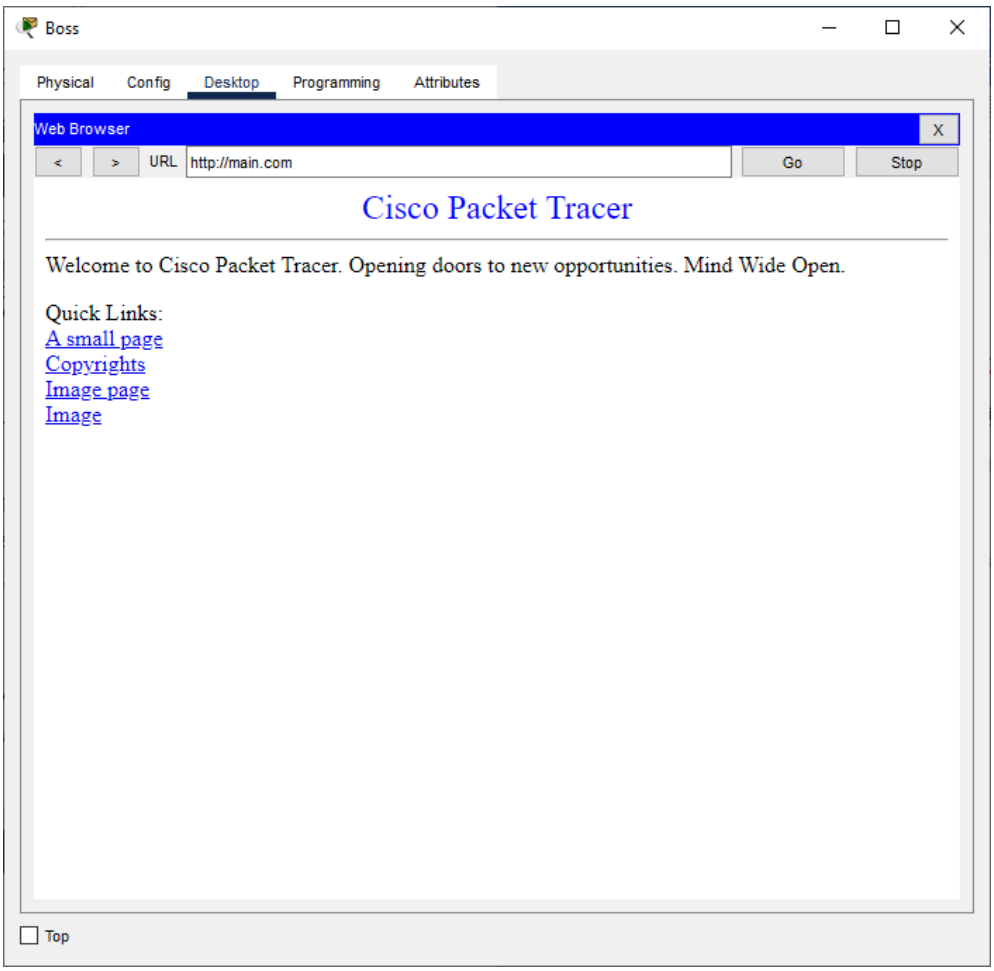

Рисунок 2.11 – Boss все ще має доступ до серверу main.com

Останнім, але не менш важливим і поширеним варіантом контролю трафіку у мережі є **регулювання ftp-трафіку**. Наступним налаштуванням можна заборонити передачу даного трафіку від відділу розробки до тестувальників, аби не перевантажувати мережу.

```
Router>en
Router#conf t
Router(config)#access-list 106 deny tcp 192.168.1.0 0.0.0.255 192.168.2.0 0.0.0.255
eq 21
Router(config)#int fa0/0
Router(config-if)#ip access-group 106 in
```

Необхідно створити пакет трафіку типу ftp. Для цього у Packet Tracer існує спеціальна функція (рис. 2.12).

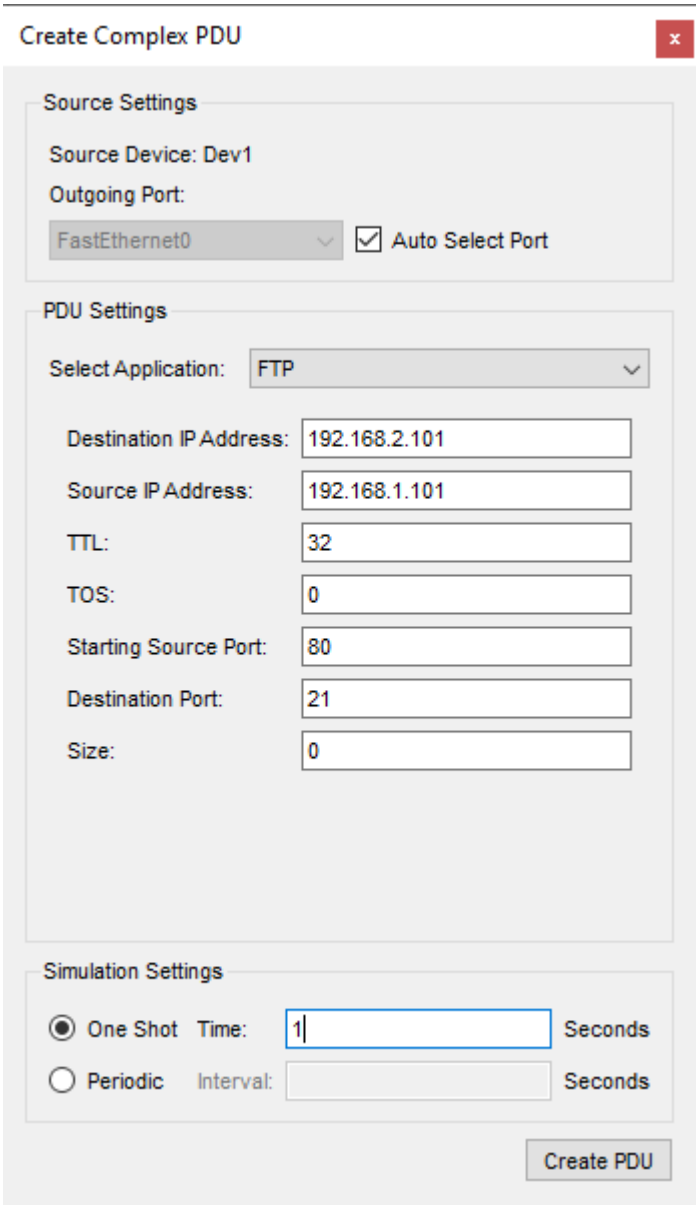

Рисунок 2.12 – Створення та відправка ftp-пакетів

Після надсилання створеного пакету, бачимо, що він не зміг досягнути пункту призначення, як і планувалося (рис. 2.13)

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num
	Failed	Dev1	192.168.2....	TCP		1.000	N	0

Рисунок 2.13 – Знищення пакетів, як заборонених

Хоча усі виконані налаштування не виглядають складними, але вони потребують часу та знань конфігурації мережі. Необхідно розробити інтерфейс, який допоможе спростити цю задачу.

3. ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ НАЛАШТУВАННЯ РОЗШИРЕНИХ СПИСКІВ КОНТРОЛЮ ДОСТУПУ

3.1. Розробка графічного інтерфейсу з використанням мови програмування JavaScript та технологій HTML і CSS

Створення графічного інтерфейсу доречніше за все виконувати у вигляді веб-сторінки з простим функціоналом. Так як наша система повинна бути зорієнтована на користувачів-початківців, саме веб-сторінка стане найкомфортнішою, адже всі ми користуємося браузерами, ми звикли до них і це не буде потребувати окремого налаштування різноманітних десктоп-застосунків, які можуть завдати шкоди вашій операційній системі.

Починаючи розробку веб-сторінки потрібно ознайомитися з технологіями, які необхідні для забезпечення ефективності, працездатності, функціональності та логіки системи. Я буду розробляти графічний інтерфейс з використанням мови гіпертекстової розмітки HTML, таблиць стилів CSS та мови програмування JavaScript.

Мова гіпертекстової розмітки HTML допомагає сформувати скелет сайту. Необхідно розуміти, в чому різниця між мовою програмування та розмітки, та чому HTML нею не являється. Дана технологія не може забезпечити динамічну функціональність, але вона може розмістити усі елементи, які ви потім зможете використати для функціонування.

Суть HTML полягає у впорядкуванні гіпертекстових посилань, які надають користувачам мережі Internet доступ до перегляду різноманітних веб-ресурсів. Останній раз ця мова оновлювалась в 2014 році, саме тоді вийшла HTML5, яка на даний момент являється стандартом.

Як і в CSS, у HTML дуже важливу роль відіграє ієрархія елементів. Кожна веб-сторінка зазвичай має дуже впорядковану структуру, яка впливає на те, як легко буде браузеру завантажити цей файл та відобразити для користувача. HTML є мовою з відкритим кодом та абсолютно безкоштовною у використанні, що є великою перевагою. Вона запросто інтегрується з іншими серверними мо-

вами, такими як PHP, Node.js, але також може працювати і лише в команді з CSS та JS.

Каскадні таблиці стилів використовуються для того, щоб надати скелету вашого сайту гарний зовнішній вигляд. CSS відповідає за все, що стосується краси та оформлення сайту: колір, стилі, макети, розміщення, шрифти, анімацію тощо. Він не діє окремо від HTML, використовуючи імена створених у окремому html-файлі класів, або унікальні ідентифікатори елементів для того, щоб змінювати їх.

До створення CSS, стилі елементам веб-сторінок призначались безпосередньо в документі html, за допомогою тега style, але це могло зробити код занадто важким для читання, особливо якщо він великий за розміром і містить велику кількість елементів, кожен з яких повинен бути оформленим [17].

Окрім цього для створення функціонального графічного інтерфейсу необхідно володіти навичками використання мови програмування JavaScript, яка є поширеною у веб-розробці. JavaScript (або скорочено JS, js) є об’єктно-орієнтованою мовою програмування. Зазвичай, вона використовується з метою надання веб-сторінкам інтерактивності, щоб дозволити користувачу взаємодію з різними елементами інтерфейсу.

JavaScript є клієнтською мовою програмування, він працює на стороні клієнта і забезпечує використання клієнт-серверної структури. Важливою перевагою у використанні JS є той факт, що для того, щоб почати з ним роботу не має необхідності встановлювати нове обладнання програмного забезпечення. Все необхідне для взаємодії з цією мовою вже налаштовано в вашому браузері.

Все, що стосується автоматизації та логіки роботи веб-сайту забезпечується за допомогою JS [18].

Тепер, коли ми ознайомилися більш детально з технологіями, необхідними для створення веб-сторінки, налаштували мережу компанії за допомогою емулятора Cisco Packet Tracer та продемонстрували роботу розширених списків

контролю доступу на Cisco IOS у створеній мережі, можна переходити до розробки графічного інтерфейсу.

3.2 Огляд функціоналу розробленої веб-орієнтованої системи

Веб-сторінка складається з трьох блоків (рис. 3.1). У верхній частині зліва розташована схема мережі, конфігурацію якої було виконано у емуляторі Cisco Packet Tracer. На схемі можна побачити її структуру, це необхідно для того, щоб користувач міг краще орієнтуватися у тому, які взаємозв'язки йому необхідно налагодити розширеними списками доступу. Справа від схеми можна побачити алгоритм дій, які потребується виконати, щоб вдало згенерувати код для налаштування маршрутизатора. Внизу знаходиться блок, з елементами, які необхідно заповнити, щоб задати бажані характеристики розширеного списку доступу.

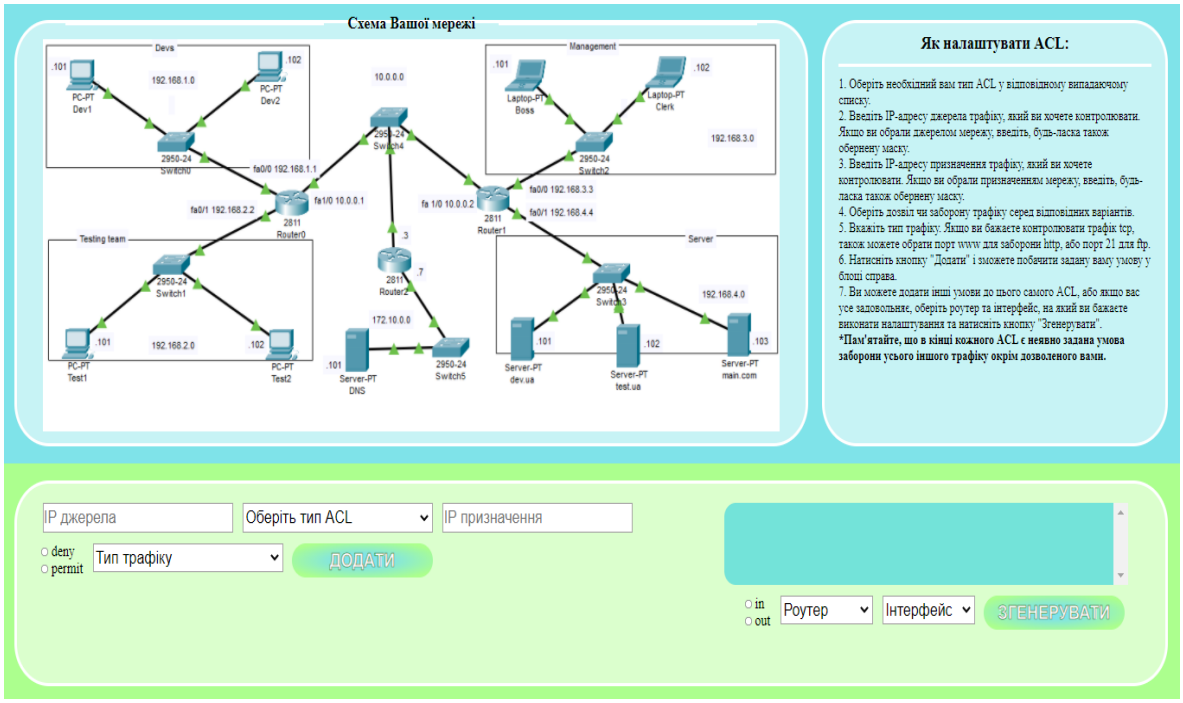

Рисунок 3.1 – Інтерфейс розробленої веб-сторінки

Розглянемо детальніше функціональний блок з необхідними для заповнення полями (рис. 3.2). Тут розташовані такі елементи як випадаючі списки, поля для введення тексту, радіобаттони та кнопка.

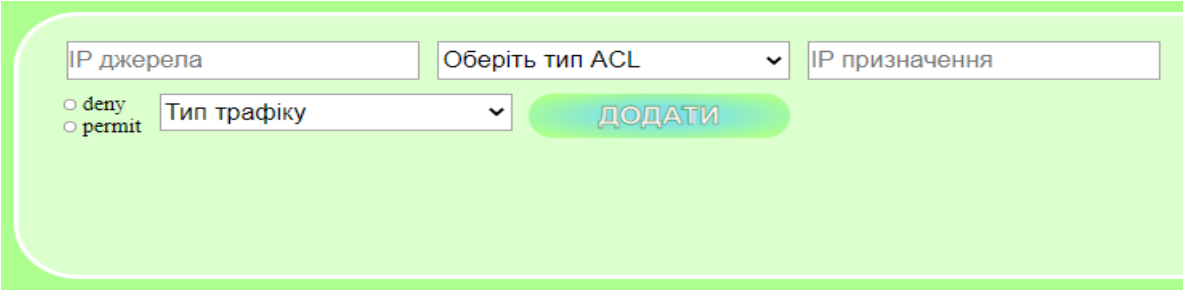

Рисунок 3.2 – Блок налаштування параметрів ACL

Випадаючий список у верхньому рядку містить типи розширених ACL, які використовувалися для налаштування поведінки мережі в емуляторі. Це типи “Host to Host”, “Host to Network”, “Network to Host” та “Network to Network”. Від того який тип користувач обере залежить чи буде йому необхідно вводити обернену маску для адрес джерела та/або призначення. (рис. 3.3)

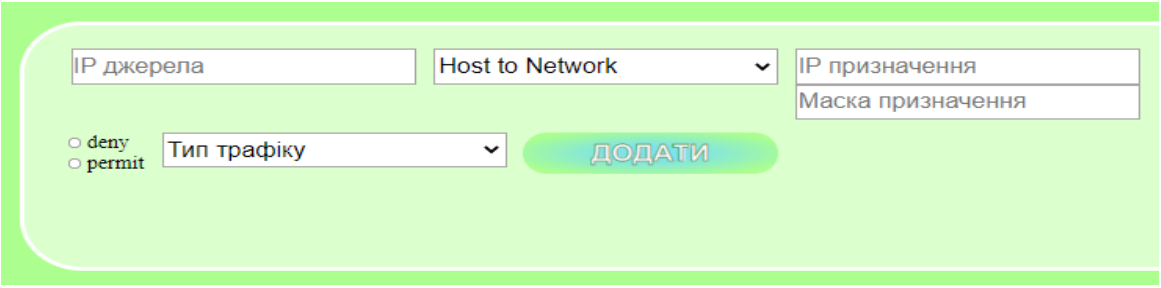

Рисунок 3.3 – Зміна відображення полів IP та маски в залежності від типу ACL

Поля для введення тексту містять валідацію на IP та маску адреси відповідно. Якщо користувач спробує ввести некоректні дані, поле підсвітиться червоним кольором та з’явиться повідомлення, щоб користувач міг зрозуміти в чому його помилка (рис. 3.4)

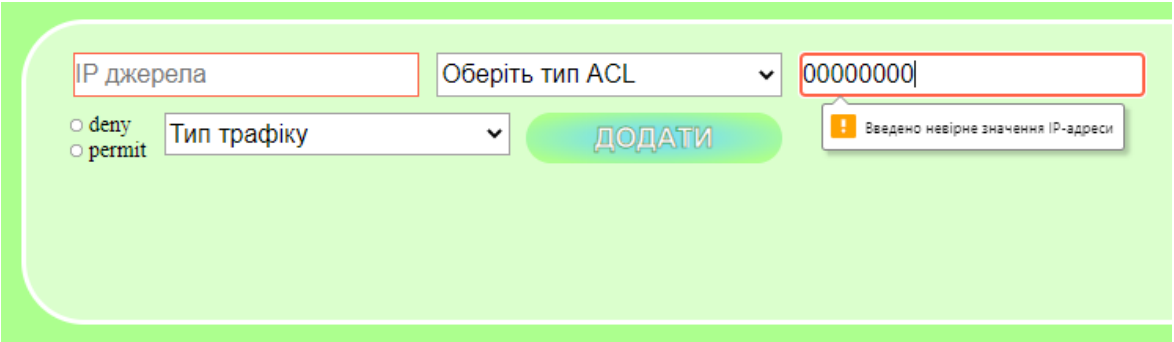

Рисунок 3.4 – Валідація полів для введення даних

Завдяки радіобаттонам “deny” та “permit” можна обрати, яку саме дію користувач бажає виконати з певним типом трафіку. Тип трафіку обирається у випадаючому списку. До варіантів вибору додано тільки три з можливих, у зв’язку з тим, що ми розглядали тільки їх під час налаштування розширених списків доступу в емульованій мережі. Серед розглянутих типів трафіку є “icmp”, “tcp” та “ip”. (рис. 3.5)

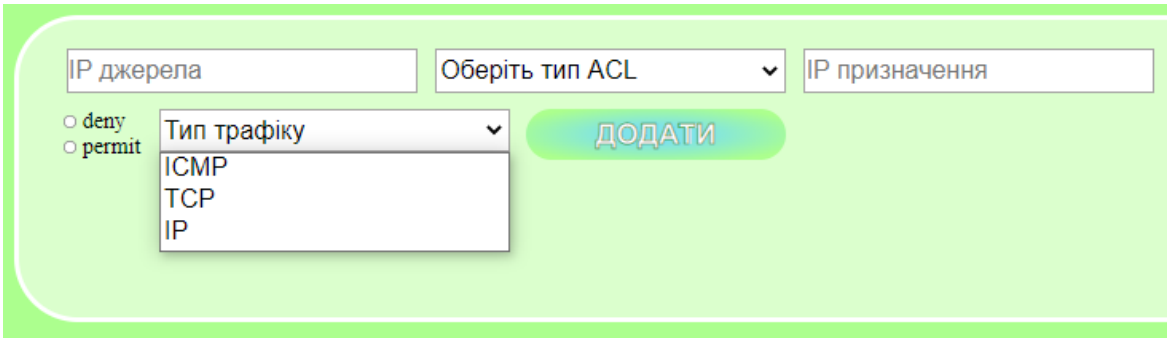

Рисунок 3.5 – Вибір типу трафіка в інтерфейсі

У випадку, якщо користувач обрав TCP, в нього з’являється можливість обрати також порт, який він хоче проконтролювати “eq www” або “eq 21”. Ця функція може допомогти, коли необхідний дозвіл або заборона передачі ftp- або http-пакетів. (рис. 3.6)

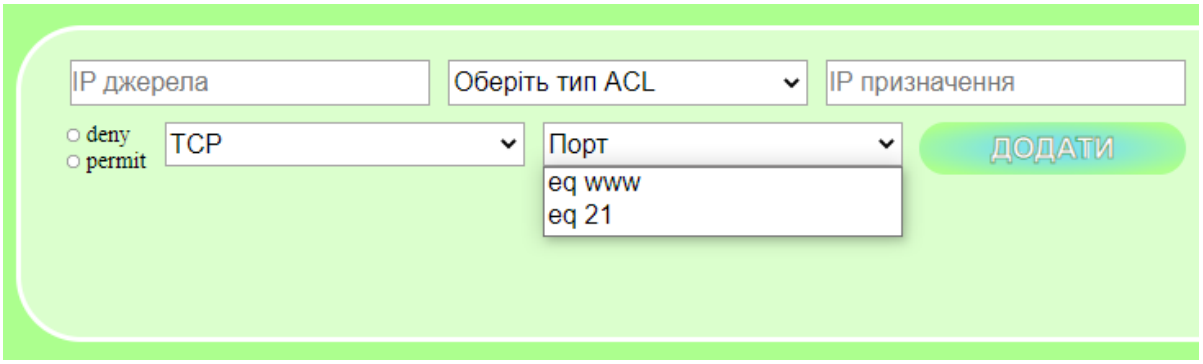

Рисунок 3.6 – При виборі tcp-трафіка з'являється можливість вибору порта

Після того, як користувач введе усі необхідні дані, він повинен натиснути кнопку «Додати». Тоді справа його щойно створене правило з’явиться у спеціально відведеному для цього блоці. Як це виглядає представлено на рис. 3.7. Ця функція існує для того, щоб користувач міг додати декілька правил для одного й того самого розширеного списку контролю доступу і тільки потім

згенерувати код, який він зможе використати для налаштування маршрутизатора.

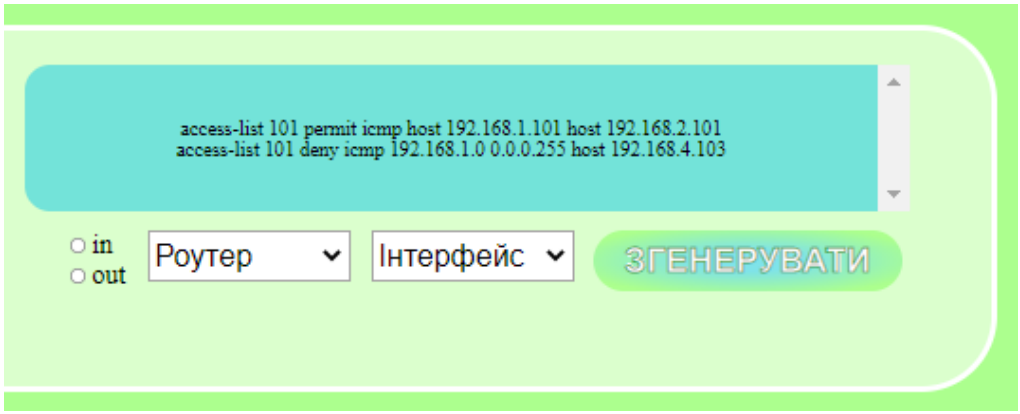

Рисунок 3.7 – Налаштування параметрів вибору маршрутизатора, інтерфейса та напрямку

Після того, як користувачем будуть додані усі бажані правила, йому необхідно обрати роутер, на якій він хоче примінити список доступу, обрати інтерфейс цього роутера та вхідний або вихідний трафік. При натисканні кнопки «Згенерувати» відкриється віконце з кодом налаштування (рис. 3.8, 3.9). Код можна скопіювати автоматично, використовуючи спеціальну кнопку внизу віконця.

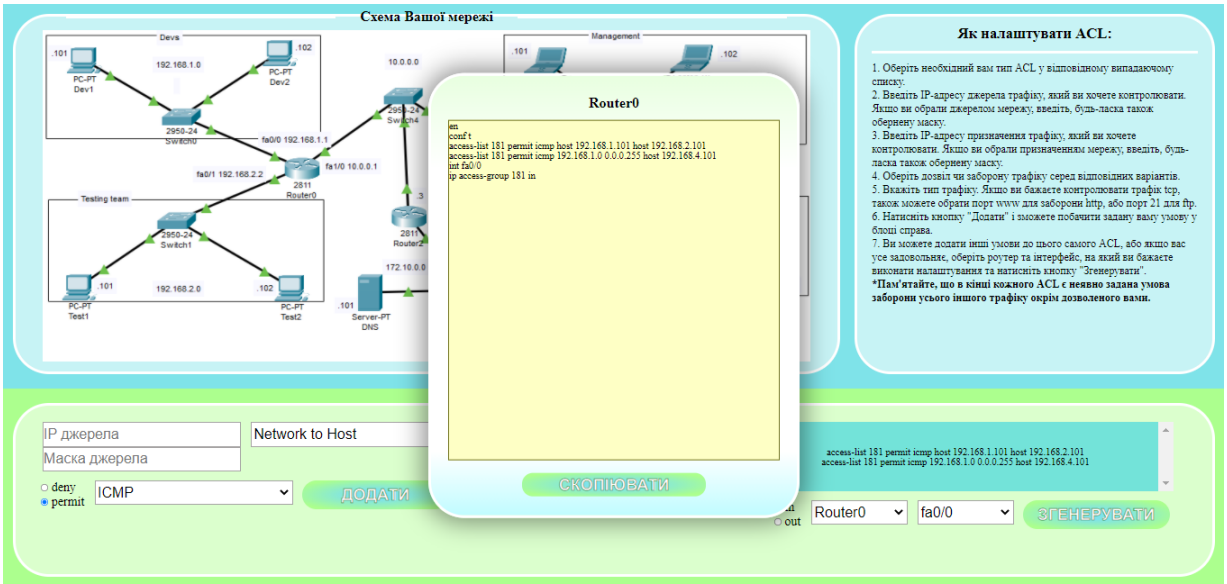

Рисунок 3.8 – Вигляд інтерфейсу під час появи спливаючого віконця з кодом

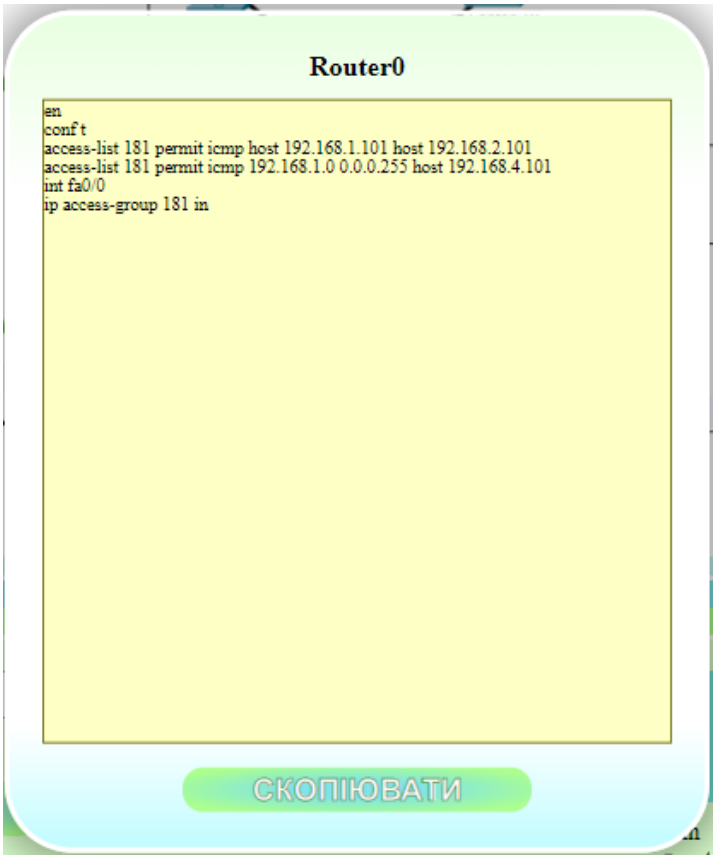

Рисунок 3.97 – Вигляд згенерованого коду, який можна скопіювати

Після натискання кнопки «Скопіювати» код буде збережений у буфер обміну, звідки його потім можна вставити в емулятор Cisco Packer Tracer, щоб перевірити чи підходить користувачу це налаштування, або безпосередньо на реальне обладнання Cisco.

3.3 Тестування графічного інтерфейсу в емуляторі Cisco Packet Tracer

Якщо ми хочемо переконатися в тому, чи ефективно працює розроблений веб-орієнтований графічний інтерфейс, нам варто перевірити це з використанням емулятора Cisco Packet Tracer, в якому вже є конфігурація такої самої мережі, і відсутні будь-які списки контролю доступу, розширені чи стандартні. У всіх пристроїв є доступ один до одного і налаштована статична маршрутизація.

Спробуємо згенерувати список контролю доступу для маршрутизатора під номером 0, який міг би контролювати відсутність у мережі розробників доступу до мережі менеджменту. (рис. 3.10)

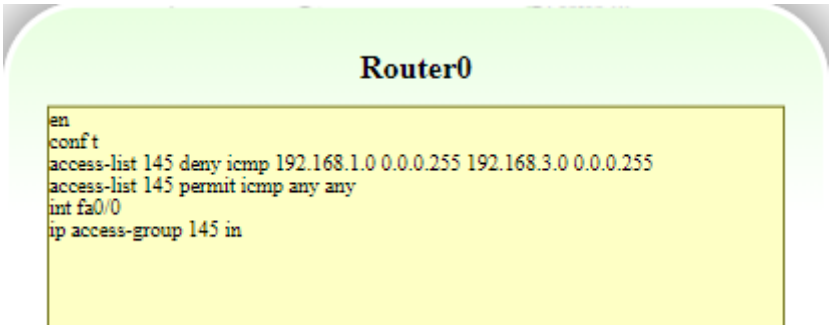

Рисунок 3.10 – Згенерований у інтерфейсі код для тестування тип Network to Network

Згенерований код необхідно скопіювати натиснувши відповідну кнопку в інтерфейсі та вставити у маршрутизатор 0 в емуляторі (рис. 3.11)

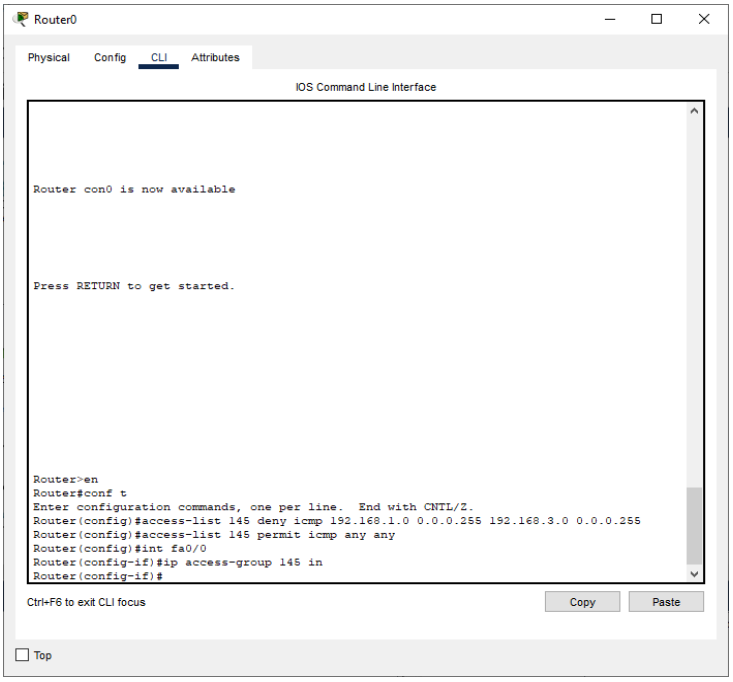

Рисунок 3.81 – Налаштування маршрутизатора 0 з використанням згенерованого коду

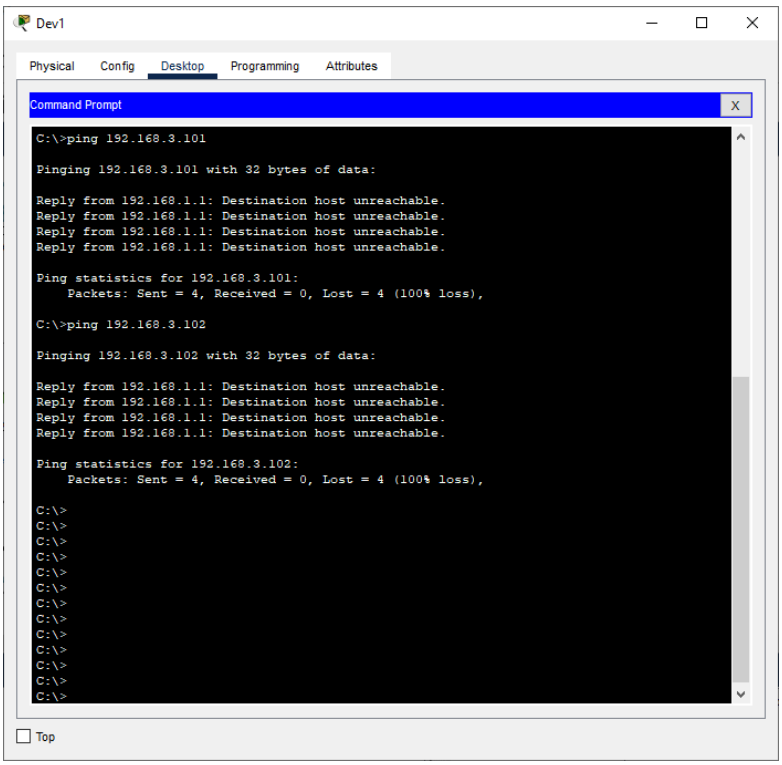

Рисунок 3.12 – Знищення заборонених пакетів

На рис. 3.12 бачимо, що налаштування працює правильно, пакети, які заборонені для передачі знищуються і не завантажують мережу.

Тепер потрібно перевірити, як працює генерування коду розширеного списку доступу, якщо в ньому наявні правила, які забороняють трафік tcp. Уявімо, що мережі менеджменту необхідно заборонити передачу ftp на свій сервер, але дозволити доступ до http-трафіку, щоб вони мали доступ на сервер (рис. 3.13)

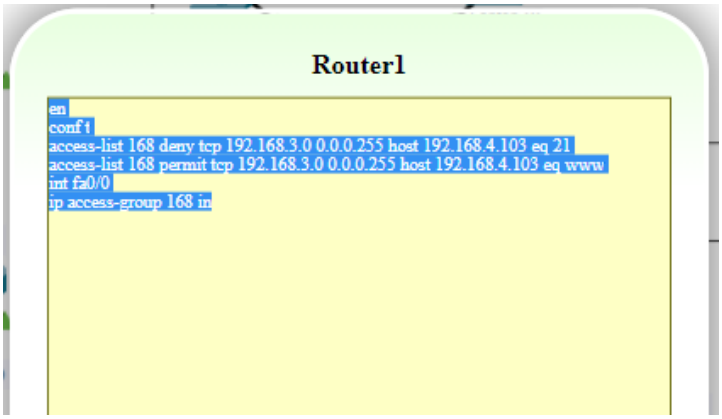

Рисунок 3.13 – Згенерований код для ACL з забороною ftp- та дозволом http-трафіків

Копіюємо отримані налаштування у маршрутизатор під номером 1 в емуляторі Packet Tracer (рис. 3.14)

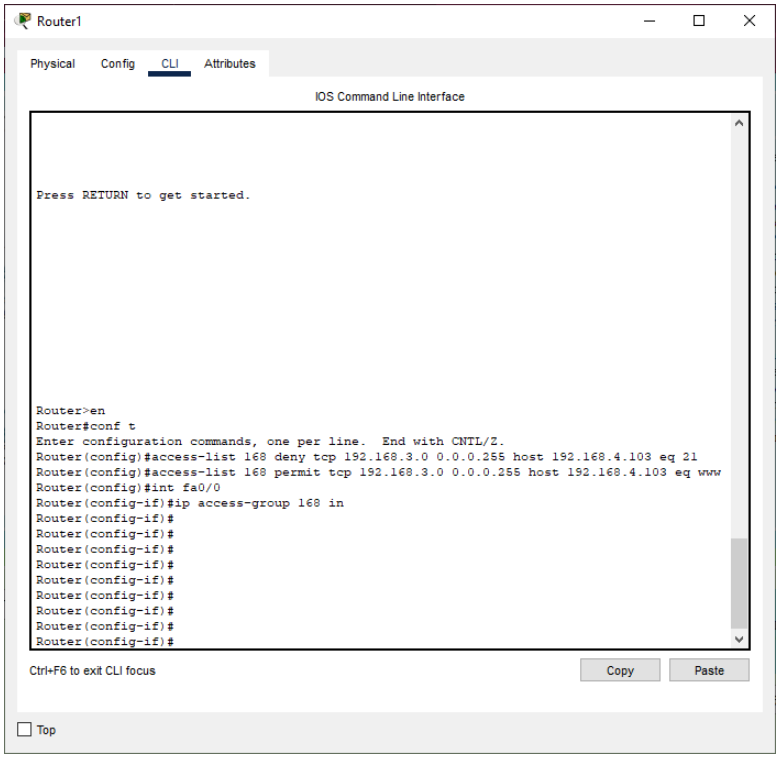

Рисунок 3.14 – Налаштування маршрутизатора 1, використовуючи згенерований інтерфейсом код

Створюємо та відправляємо пакети ftp та http-трафіків (рис. 3.15, 3.16) та дивимось на результат на рис. 3.17.

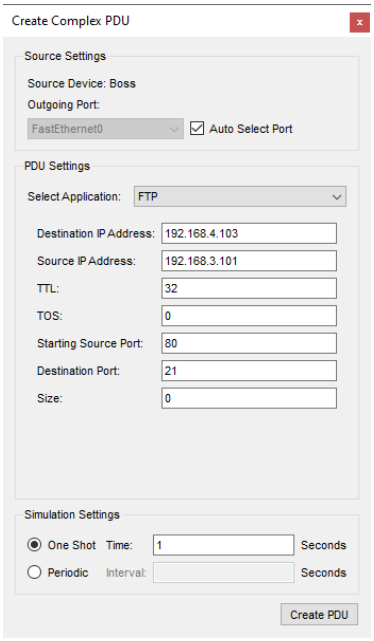

Рисунок 3.15 – Створення ftp-пакету

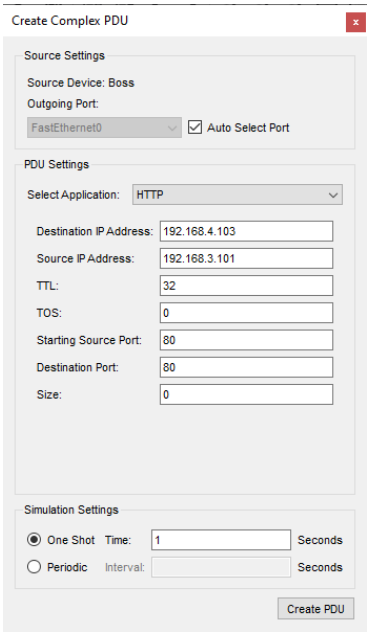

Рисунок 3.16 – Створення http-пакету

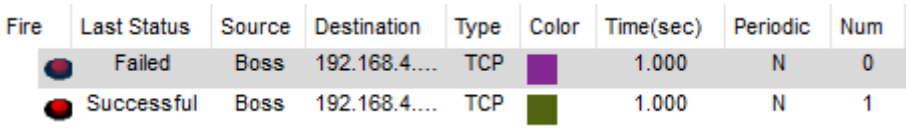

Рисунок 3.17 – Знищення ftp-пакету та дозвіл http-трафіку

Як результат - бачимо, що процес налаштування розширених списків доступу з використанням розробленого графічного інтерфейсу є значно зручнішим, більш швидким та приємним.

ВИСНОВКИ

Контроль поведінки користувачів та забезпечення раціонального використання трафіку у мережах важко уявити без правильно налаштованих на маршрутизаторах списків контролю доступу. У зв’язку з цим важливим питанням є, як оптимізувати та спростити цей процес для користувачів, які не мають достатньо досвіду. Відповіддю на це можна вважати створення графічного інтерфейсу, який буде легкодоступним, зручним у використанні та простим, навіть для новачків у сфері телекомунікаційних мереж.

Під час розробки веб-орієнтованої системи вдалося досягнути мети і створити такий інтерфейс, який задовольняв би потребам потенціального користувача. Даний інтерфейс надає можливість після введення параметрів, таких як IP-адреси джерела та призначення, їх обернені маски, заборона або дозвіл та якого саме типу трафіку, на якому порті, сформулювати перелік правил розширеного списку контролю доступу. Після вибору маршрутизатора, його інтерфейсу та бажаного напрямку, за яким користувач хоче контролювати трафік, для нього буде згенерований і відображений код налаштування обраного роутера. Текст коду можна скопіювати вручну або завдяки спеціально відведеній для цього кнопці.

Під час тестування за допомогою сконфігурованої у емуляторі Cisco Packet Tracer мережі було виявлено, що використання розробленого графічного інтерфейсу значно полегшує і пришвидшує процес налаштування маршрутизаторів у мережі. Потенційно розроблену веб-орієнтовану інформаційну систему можна також використовувати для виконання налаштувань на реальному обладнанні компанії Cisco.

СПИСОК ЛІТЕРАТУРИ

1. Олифер В., Олифер Н. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 5-е изд. — СПб.: Питер, 2016. — 992 с.
2. Організація комп'ютерних мереж: підручник: для студ. спеціальності 122 «Комп'ютерні науки» / КПІ ім. Ігоря Сікорського ; Ю. А. Тарнавський, І. М. Кузьменко. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 259 с.
3. Ричард Смит. Элементарная информационная безопасность, 2016 – 150с.
4. ACL: списки контроля доступа в Cisco IOS [Електронний ресурс] - https://habr.com/ru/post/121806/
5. Лекция 5: Списки контроля доступа [Електронний ресурс] - https://intuit.ru/studies/courses/3646/888/lecture/31159?page=5
6. С.А. Широков, В.Н. Круглов. Лабораторная работа. Обеспечение безопасности в сетях на базе оборудования Cisco. Списки управления доступом ACL и служба трансляции адресов NAT/ сост. Екатеринбург: УГТУ-УПИ, 2010 – 35 с. [Електронний ресурс] - https://works.doklad.ru/view/yMfDx_2DwNo/all.html
7. Access Control List (ACL) [Електронний ресурс] - https://www.imperva.com/learn/data-security/access-control-list-acl/
8. Уэнделл Одом Официальное руководство Cisco по подготовке к сертификационным экзаменам CCNA ICND2 200-101. Архив 3, 2015 [Електронний ресурс] - http://www.williamspublishing.com/PDF/978-5-8459-1811-6/part.pdf
9. Что такое ACL и как его настраивать [Електронний ресурс] - http://ciscotips.ru/acl
10. Cisco Switch Support for Cisco NAC Appliance. [Електронний ресурс] - http://www.Cisco.com/en/US/products/ps6128/products_device_support_table09186a 008075fff6.html
11. Настройка списков доступа IP [Електронний ресурс] - https://www.cisco.com/c/ru_ru/support/docs/security/ios-firewall/23602-confaccesslists.html

12. Лекция 9: Списки доступа ACL. Настройка статического и динамического NAT [Електронний ресурс] - https://intuit.ru/studies/courses/3549/791/lecture/29226?page=2
13. Access Control List (ACL) [Електронний ресурс] - https://cio-wiki.org/wiki/Access_Control_List_(ACL)
14. Access Control List Explained with Examples [Електронний ресурс] - https://www.computernetworkingnotes.com/ccna-study-guide/access-control-list-explained-with-examples.html
15. "Cisco Packet Tracer". Cisco Networking Academy. Retrieved 26 August 2018.
16. Michel Bakni, Yudith Cardinale, Luis Manuel Moreno. "An Approach to Evaluate Network Simulators: An Experience with Packet Tracer". 2018 - 29–36с.
17. Эд Титтел, Джефф Ноубл. HTML, XHTML и CSS для чайников, 7-е издание = HTML, XHTML & CSS For Dummies, 7th Edition. — М.: «Диалектика», 2011. — 400 с.
18. Мова JavaScript та її можливості [Електронний ресурс] - https://sites.google.com/site/webtehnologiietawebdizajn/mova-javascript-ta-ieie-mozlivosti

ДОДАТОК А

index.html

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

<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">
    <title>Access lists</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <main class="main">
        <div class="main_inner">
            <section class="schema">
                <h2 class="schema_header">Схема Вашої мережі</h2>
                <div class="schema_img-wrap">
                    <img class="schema_img" src="acl-top.bmp" alt="schema">
                </div>
            </section>
            <section class="info">
                <h2 class="info_header">Як налаштувати ACL:</h2>
                <ul class="info_list">
                    1. Оберіть необхідний вам тип ACL у відповідному випадаючому
списку.<br>
                    2. Введіть IP-адресу джерела трафіку, який ви хочете контро-
лювати. Якщо ви обрали джерелом мережу, введіть, будь-ласка також обернену мас-
ку.<br>
                    3. Введіть IP-адресу призначення трафіку, який ви хочете
контролювати. Якщо ви обрали призначенням мережу, введіть, будь-ласка також
обернену маску.<br>
                    4. Оберіть дозвіл чи заборону трафіку серед відповідних
варіантів.<br>
                    5. Вкажіть тип трафіку. Якщо ви бажаєте контролювати трафік
tcp, також можете обрати порт www для заборони http, або порт 21 для ftp.<br>
                    6. Натисніть кнопку "Додати" і зможете побачити задану ваму
умову у блоці справа. <br>
                    7. Ви можете додати інші умови до цього самого ACL, або якщо
вас усе задовольняє, оберіть роутер та інтерфейс, на який ви бажаєте виконати
налаштування та натисніть кнопку "Згенерувати".<br>
                    <b>*Пам'ятайте, що в кінці кожного ACL є неявно задана умова
заборони усього іншого трафіку окрім дозволеного вами.</b>
                </ul>
            </section>
        </div>
        <div class="tool">
            <div class="tool_inner">
                <div class="tool_fields-wrap">
                    <div class="tool_ip">
                        <div>
                            <input class="input-field" id="source_ip" placehold-
er="IP джерела" type="text" minlength="7"
                                maxlength="15" size="15"
                                pattern="^((\d{1,2}|1\d\d|2[0-4]\d|25[0-
5])\.){3}(\d{1,2}|1\d\d|2[0-4]\d|25[0-5])$"><br>
                            <input class="input-field" id="ip1_mask" placehold-
er="Маска джерела" />
```

```
                        </div>
                        <select class="select" id="acl_type">
                            <option disabled selected hidden value="">Оберіть
тип ACL</option>
                            <option value="0">Host to Host</option>
                            <option value="1">Host to Network</option>
                            <option value="2">Network to Host</option>
                            <option value="3">Network to Network</option>
                        </select>
                        <div>
                            <input class="input-field" id="dest_ip" placehold-
er="IP призначення" type="text"
                                minlength="7" maxlength="15" size="15"
                                pattern="^((\d{1,2}|1\d\d|2[0-4]\d|25[0-
5])\.){3}(\d{1,2}|1\d\d|2[0-4]\d|25[0-5])$"><br>
                            <input class="input-field" id="ip2_mask" placehold-
er="Маска призначення" />
                        </div>
                        <div class="action_wrap">
                            <div class="radio_group">
                                <div>
                                    <input   class="input-radio"   type="radio"
id="deny" name="den_per" value="deny">
                                    <label for="deny">deny</label>
                                </div>
                                <div>
                                    <input   class="input-radio"   type="radio"
id="permit" name="den_per" value="permit">
                                    <label for="permit">permit</label>
                                </div>
                            </div>
                            <select class="select" id="traffic_type">
                                <option disabled selected hidden value="">Тип
трафіку</option>
                                <option value="icmp">ICMP</option>
                                <option value="tcp">TCP</option>
                                <option value="ip">IP</option>
                            </select>
                            <select class="select" id="port_type">
                                <option   disabled   selected   hidden   val-
ue="">Порт</option>
                                <option value="eq www">eq www</option>
                                <option value="eq 21">eq 21</option>
                            </select>
                            <button     id="add_button"     class="button"><span
class="button_text">Додати</span></button>
                        </div>
                    </div>
                    <div class="tool_options">
                        <div class="added-rules"></div>
                        <div class="options_wrap">
                            <div class="radio_group">
                                <div>
                                    <input   class="input-radio"   type="radio"
id="in" name="direction" value="in">
                                    <label for="in">in</label>
                                </div>
                                <div>
                                    <input   class="input-radio"   type="radio"
id="out" name="direction" value="out">
                                    <label for="out">out</label>
                                </div>
                            </div>
```

```
                            <select class="select" id="routers">
                                <option    disabled    selected    hidden    val-
ue="">Роутер</option>
                                <option value="Router0">Router0</option>
                                <option value="Router1">Router1</option>
                            </select>
                            <select class="select" id="int">
                                <option    disabled    selected    hidden    val-
ue="">Інтерфейс</option>
                                <option value="fa0/0">fa0/0</option>
                                <option value="fa0/1">fa0/1</option>
                                <option value="fa1/0">fa1/0</option>
                            </select>
                            <button   id="generateButton"   class="button"><span
class="button_text">Згенерувати</span></button>
                        </div>
                    </div>
                </div>

            </div>
        </div>
    </main>
    <div id="modal" class="modal">
        <h2 class="modal_header">Router</h2>
        <div class="modal_text">
            en <br>
            conf t <br>

        </div>
        <button      id="modalButton"      class="button      modal_button"><span
class="button_text">Скопіювати</span></button>
    </div>
    <script src="script.js"></script>
</body>

</html>
```

style.css

```
*,
*::after,
*::before {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    height: 100vh;
    position: relative;
}

.main {
    z-index: 1;
    background: rgba(64, 213, 222, 0.67);
    height: 100%;
    width: 100%;
    display: flex;
    flex-direction: column;
}


.main_inner {
    flex: 1;
    display: flex;
    padding: 24px;
}

.schema {
    flex: 1;
    border: 5px solid white;
    border-radius: 60px;
    display: flex;
    flex-direction: column;
    justify-content: center;
    padding: 0 44px 38px 44px;
    background-color: rgba(255, 255, 255, 0.56);
    min-width: 945px;
    height: 600px;
    margin-right: 24px;
}

.schema_header {
    display: flex;
    justify-content: center;
    align-items: center;
    white-space: nowrap;
}

.schema_header::after,
.schema_header::before {
    content: '';
    margin: 20px 40px;
    width: 100%;
    height: 5px;
    background-color: white;
    text-transform: uppercase;
    font-size: 24px;
}
```

```
.schema_img-wrap {
    height: 100%;
    width: auto;
}

.schema_img {
    height: 100%;
    width: 100%;
}

.info {
    flex: 1;
    border: 5px solid white;
    border-radius: 60px;
    min-width: 300px;
    max-width: 600px;
    height: 100%;
    background: rgba(255, 255, 255, 0.56);
}

.info_header {
    display: flex;
    justify-content: center;
    align-items: center;
    font-size: 25px;
    min-height: 48px;
    padding: 13px;
}

.info_list {
    height: 100%;
    list-style-position: inside;
    position: relative;
    margin: 0 24px;
    padding: 13px 0;
    border-top: 5px solid white;
    font-size: 18px;
}

.tool {
    flex: 1;
    display: flex;
    justify-content: center;
    align-items: center;
    background: #ACFF8E;
    padding: 24px;
}

.tool_inner {
    width: 100%;
    height: 100%;
    border: 5px solid white;
    border-radius: 60px;
    display: flex;
    justify-content: center;
    align-items: top;
    background: rgba(255, 255, 255, 0.56);
}

.tool_fields-wrap {
    padding: 27px;
    display: flex;
```

```
    flex-direction: row;
}

/* .tool_fields-wrap > div{
    border: 1px solid red
} */

.tool_ip {
    flex-wrap: wrap;
    display: flex;
    height: fit-content;
    width: 60%;
}

.action_wrap{
    padding-top: 15px;
    display: flex;
}

.tool_options {
    flex-grow: 1;
    display: flex;
    flex-direction: column;
    align-items: center;
}

#add_button{
    width: 245px;
    margin-left: 15px;
    margin-right: auto;
}

.radio_group{
    padding-left: 15px;
    font-size: 21px;
}

.tool_options .radio_group {
    width: 60px;
    display: flex;
    flex-direction: column;
}

.added-rules{
    background-color: rgba(64, 213, 222, 0.67);
    border-radius: 20px;
    width: 700px;
    height: 115px;
    display: flex;
    justify-content: center;
    flex-direction: column;
    align-items: center;
    overflow-y: scroll;
}

.options_wrap{
    margin-top: 15px;
    display: flex;
}

.options_wrap > *{
    margin-left: 15px;
}
```

```
#int{
    width: 160px;
}

#routers{
    width: 160px;
}

.input-field, .select {
    margin-left: 17px;
    width: 330px;
    height: 41px;
    font-size: 24px;
    border: 2px solid #A6A4A4;
}

.input-field:invalid{
    border-color: tomato;
    outline-color: tomato;
}

#ip1_mask, #ip2_mask, #port_type{
    display: none;
}

.select {}

.button {
    border: none;
    background: radial-gradient(77.87% 77.87% at 50% 50%, #7FE3E9 21.88%,
#ACFF8E 75%);
    border-radius: 50px;
}

#generateButton{
    width: 245px;
}

.button_text {
    font-style: normal;
    font-weight: bold;
    font-size: 27px;
    line-height: 34px;
    text-transform: uppercase;
    color: #FFFFFF;
    -webkit-text-stroke: 1px #1BA529;
}

.modal {
    display: none;
    flex-direction: column;
    align-items: center;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    z-index: 3;
    background: linear-gradient(180deg, #E8FFE0 0%, #FFFFFF 20.87%, #FFFFFF
85.6%, #C1FCFF 100%);
    width: 618px;
    height: 742px;
    padding: 32px;
```

```
    border: 5px solid white;
    border-radius: 60px;
    box-shadow: 0px 0px 39px 10px rgba(0, 0, 0, 0.404);
}
.modal_header{
    padding-bottom: 15px;
}
.modal_text {
    background: #FEFFC5;
    border: 1px solid #72741C;
    width: 553px;
    height: 611px;
    box-sizing: border-box;

}

.modal_button {
    margin-top: 21px;
    width: 307px;
    height: 41px;
}
```

script.js

```
const generateButton = document.querySelector("#generateButton");
const modalButton = document.querySelector("#modalButton");
const modal = document.querySelector(".modal");
const toggleModal = () => {
  if (modal.style.display != "flex") {
    modal.style.display = "flex";
  } else {
    modal.style.display = "none";
  }
  console.log(modal.style.display)
};

const sourceIPField = document.querySelector("#source_ip");
const sourceIPMaskField = document.querySelector("#ip1_mask");
const destIPMaskField = document.querySelector("#ip2_mask");
const destIPField = document.querySelector("#dest_ip");
const aclTypeField = document.querySelector("#acl_type");

const actionRadios = document.getElementsByName('den_per');
const trafficTypeField = document.querySelector("#traffic_type");
const portTypeField = document.querySelector('#port_type');
const addButton = document.querySelector("#add_button");
const rulesListView = document.querySelector('.added-rules');

let sourceIP, destIP, aclType, action, trafficType, portType;
let accessNum = Math.round(Math.random() * (199 - 100) + 100);
let rule = 'access-list ' + accessNum + ' ';
const rules = [];



const claerFields = () => {
  let container, inputs, index;
  container = document.querySelector(".tool_ip");
  inputs = container.getElementsByTagName('input');
  for (index = 0; index < inputs.length; ++index) {
    if(inputs[index].type =="text")
      inputs[index].value = '';
  }
}

trafficTypeField.addEventListener("change", () => {
  portTypeField.style.display = trafficTypeField.value == "tcp" ? 'block' :
'none'
})

aclTypeField.addEventListener("change", () => {
  let state = aclTypeField.value;
  if (state == 0) {
    sourceIPMaskField.style.display = "none";
    destIPMaskField.style.display = "none";
  }
  if (state == 1) {
    sourceIPMaskField.style.display = "none";
    destIPMaskField.style.display = "block";
  }
  if (state == 2) {
    sourceIPMaskField.style.display = "block";
    destIPMaskField.style.display = "none";
  }
```

```
  if (state == 3) {
    sourceIPMaskField.style.display = "block";
    destIPMaskField.style.display = "block";
  }
});

addButton.addEventListener("click", () => {
  if (sourceIPField.checkValidity() && destIPField.checkValidity()) {
    sourceIP = sourceIPField.value;
    destIP = destIPField.value;
    aclType = aclTypeField.value;
    portType = portTypeField.value;
    for (let i = 0, length = actionRadios.length; i < length; i++)
      if (actionRadios[i].checked) {
        action = actionRadios[i].value;
        break;
      }
    trafficType = trafficTypeField.value;
    let source, dest;
    switch (aclType) {
      case "0":
        source = "host " + sourceIP;
        dest = "host " + destIP;
        break;
      case "1":
        source = "host " + sourceIP;
        dest = destIP + " " + destIPMaskField.value;
        break;
      case "2":
        source = sourceIP + " " + sourceIPMaskField.value;
        dest = "host " + destIP;
        break;
      case "3":
        source = sourceIP + " " + sourceIPMaskField.value;
        dest = destIP + " " + destIPMaskField.value;
        break;
    }

    rule += action + " " + trafficType + " " + source + " " + dest;
    if (trafficTypeField.value == "tcp") rule += " " + portType;
    rules.push(rule);
    ruleView = document.createElement("span");
    ruleView.textContent = rule;
    rulesListView.appendChild(ruleView);
    console.log(ruleView);
    console.log(rules);
    rule = "access-list " + accessNum + " ";
    claerFields();
  } else {
    errorMessage = "Введено невірне значення IP-адреси";
    sourceIPField.setCustomValidity(errorMessage);
    destIPField.setCustomValidity(errorMessage);
    sourceIPField.reportValidity()
    destIPField.reportValidity()
  }
});

const modalText = document.querySelector('.modal_text')
generateButton.addEventListener("click", () => {
  toggleModal();
  document.querySelector('.modal_header').innerHTML = docu-
ment.getElementById('routers').value
```

```
  rules.forEach(rule => {
    modalText.innerHTML += rule + '<br>'
  });
  modalText.innerHTML += 'int ' + document.getElementById('int').value + '<br>'
  let directionsRadio = document.getElementsByName('direction');
  let direction = "";
  for (let i = 0, length = directionsRadio.length; i < length; i++)
  if (directionsRadio[i].checked) {
    direction = actionRadios[i].value;
    break;
  }
  modalText.innerHTML += "ip access-group " + accessNum + " " + direction
});

const copyToClipboard = str => {
  const el = document.createElement('textarea');
  el.value = str;
  document.body.appendChild(el);
  el.select();
  document.execCommand('copy');
  document.body.removeChild(el);
};

modalButton.addEventListener("click", () => {
  copyToClipboard(modalText.innerText)
  modalText.innerHTML = "en <br> conf t <br>"
  toggleModal();
});
```