

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

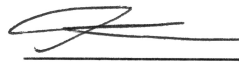
ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект програмного застосунку для
розрахунку оптимальних списків доступу для
міжмережного екрану»

Виконав студент 4-го курсу групи КБ-19-1
спеціальності 125 «Кібербезпека»

 Євгеній Мисюк

Керівник доцент кафедри КІ та КБ, к.т.н., доцент

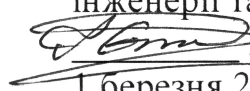
 Юрій Россінський

Рецензент доцент кафедри КІ та КБ, к.т.н., доцент

 Юрій Бродський

Житомир – 2023

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
 Андрій Єфіменко
1 березня 2023 р.

ЗАВДАННЯ

на випускн у кваліфікаційну роботу бакалавра

Студент: Мисюк Євгеній Юрійович

Тема роботи: «Проект програмного застосунку для розрахунку оптимальних списків доступу для міжмережного екрану»

затверджена Наказом університету від 1 березня 2023 р. № 99

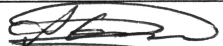
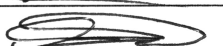
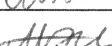
Термін здачі студентом закінченої роботи 20 червня 2023 р.

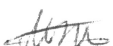
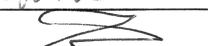
Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

Об'єктом дослідження є процес створення застосунку для розрахунку оптимальних списків доступу міжмережного екрану. Предметом дослідження є програмний застосунок розроблений в Microsoft Visual Studio для розрахунку оптимальних списків доступу для міжмережного екрану.

Обладнання та програмне забезпечення: Microsoft Visual Studio 2019

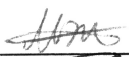
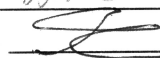
Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Єфіменко А.А.	28.03.2023 	28.03.2023 
2	Єфіменко А.А.	28.03.2023 	28.03.2023 
3	Єфіменко А.А.	28.03.2023 	28.03.2023 

Студент  Євгеній Мисюк
Керівник  Юрій Россінський

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	28 березня 2023	Виконав
2	Проведення літературного огляду та аналіз існуючих досліджень	30 березня 2023 – 02 квітня 2023	Виконав
3	Визначення методології дослідження та розробка плану дослідження	04 квітня 2023 – 07 квітня 2023	Виконав
4	Збір та аналіз даних	10 квітня 2023 – 12 квітня 2023	Виконав
5	Розробка програмного застосунку	15 квітня 2023 – 29 квітня 2023	Виконав
6	Встановлення, налагодження та тестування роботи продукту	01 травня 2023 – 06 травня 2023	Виконав
7	Аналіз результатів та висновки	08 травня 2023 – 12 травня 2023	Виконав
8	Написання тексту роботи, включаючи вступ, розділи та висновки	15 травня 2023 – 28 травня 2023	Виконав
9	Форматування та оформлення роботи згідно з вимогами	29 травня 2023 – 05 червня 2023	Виконав
10	Перегляд та редагування роботи	06 червня 2023 – 14 червня 2023	Виконав
11	Отримання відгуку та рецензії на кваліфікаційну роботу	18 червня 2023 – 20 червня 2023	Виконав
12	Захист кваліфікаційної роботи	28 червня 2023	Виконав

Студент  Євгеній Мисюк
 Керівник  Юрій Россінський

АНОТАЦІЯ

Кваліфікаційна робота складається з проєкту застосунку для розрахунку ACL її реалізацію та пояснювальної записки. Пояснювальна записка складається з 99 сторінок та 38 ілюстрацій.

Метою роботи є розробка застосунку для розрахунку списків доступу.

Розроблений застосунок надає функціонал для ефективного розрахунку списків контролю доступу на основі визначених критеріїв. Він використовує алгоритми та структури даних для обробки та аналізу вимог до контролю доступу. Застосунок надає зручний інтерфейс користувача для введення необхідних параметрів та генерації списків контролю доступу. Використовуються різні техніки оптимізації для забезпечення ефективності та точності згенерованих списків.

Проєкт включає реалізацію застосунку. Проводяться процедури тестування та валідації для перевірки функціональності та продуктивності застосунку.

Дослідження, проведені в рамках роботи, включають аналіз існуючих підходів та методів для розрахунку списків контролю доступу, ідентифікацію викликів та вимог, а також пропозицію оптимізованого рішення. Запропоноване рішення враховує такі фактори, як масштабованість, гнучкість та дотримання стандартів галузі.

Отримані результати демонструють ефективність розробленого застосунку у генерації оптимальних списків контролю доступу. Застосунок надає цінний інструмент для мережевих адміністраторів та фахівців з безпеки для оптимізації процесу конфігурації контролю доступу та підвищення безпеки мережі.

КЛЮЧОВІ СЛОВА: ACL, СПИСКИ КОНТРОЛЮ ДОСТУПУ, МЕРЕЖЕВА БЕЗПЕКА, РОЗРОБКА ЗАСТОСУНКІВ, ОПТИМІЗАЦІЯ, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, МАСШТАБОВАНІСТЬ, ГНУЧКІСТЬ, МЕРЕЖЕВЕ УПРАВЛІННЯ

KIK.KP.B – 125 – 11 – ПЗ

Змн.	Арк.	№ докум.	Підпис	Дата	Лім.	Арк.	Аркушів
Розроб.		Мисюк С.Ю.		20.06.23	Проект програмного застосунку для розрахунку оптимальних списків доступу для міжмережного екрану ФІКТ, Гр. КБ-19-1		
Керівник		Росієвський Ю.М.		20.06.23			
Реценз		Бродський Ю.Б.		20.06.23			
Н. контр.		Росієвський Ю.М.		20.06.23			
Зав. каф.		Єфіменко А.А.		20.06.23			
						4	99

ANNOTATION

The qualification work consists of a project of an application for calculating ACL, its implementation, and an explanatory report. The explanatory report comprises 99 pages and includes 38 illustrations.

The aim of the work is to develop an application for calculating optimal access control lists.

The developed application provides functionalities for efficient calculation of access control lists based on specified criteria. It incorporates algorithms and data structures to process and analyze access control requirements. The application offers a user-friendly interface for inputting the necessary parameters and generating the access control lists. Various optimization techniques are employed to ensure the efficiency and accuracy of the generated lists.

The project encompasses the design and implementation of the application, including the development of relevant modules, user interface design, and integration of necessary libraries and frameworks. Testing and validation procedures are conducted to verify the functionality and performance of the application.

The research conducted as part of the work includes an analysis of existing approaches and methods for access control list calculation, identification of challenges and requirements, and the proposal of an optimized solution. The proposed solution considers factors such as scalability, flexibility, and adherence to industry standards.

The achieved results demonstrate the effectiveness of the developed application in generating optimal access control lists. The application provides a valuable tool for network administrators and security professionals to streamline the process of access control configuration and enhance network security.

KEYWORDS: ACL, ACCESS CONTROL LISTS, NETWORK SECURITY, APPLICATION DEVELOPMENT, OPTIMIZATION, USER INTERFACE, SCALABILITY, FLEXIBILITY, NETWORK ADMINISTRATION.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	7
ВСТУП.....	8
РОЗДІЛ 1. АНАЛІЗ ПОБУДОВИ ТА ЗАСТОСУВАННЯ СПИСКІВ ДОСТУПУ .	10
1.1. Визначення та класифікація списків доступу	10
1.2. Алгоритми роботи та правила формування списків доступу	18
1.3. Засоби автоматизованого розрахунку та формуванню списків доступу	22
1.4. Висновки до розділу 1	28
РОЗДІЛ 2. ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ РОЗРАХУНКУ ОПТИМАЛЬНИХ СПИСКІВ ДОСТУПУ	30
2.1. Проєктування загальної структури застосунку	30
2.2. Проєктування модулів розрахунку та формування списків доступу	34
2.3. Особливості формування списків доступу для різних міжмережних екранів	43
2.4. Висновки до розділу 2	47
РОЗДІЛ 3. ВСТАНОВЛЕННЯ, НАЛАГОДЖЕННЯ, ТА ТЕСТУВАННЯ РОБОТИ ЗАСТОСУНКУ	49
3.1. Встановлення та налагодження та видалення застосунку.....	49
3.2. Тестування роботи застосунку для формування простих списків доступу..	54
3.3. Тестування роботи застосунку для оптимальних списків доступу	57
3.4. Висновки до розділу 3	61
ВИСНОВКИ.....	62
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	64
ДОДАТКИ.....	66

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ACL – Access Control List – список контролю доступу

ПЗ – програмне забезпечення

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

IP – Internet Protocol

VLAN – Virtual Local Area Network

MAC – Media Access Control

VPN – Virtual Private Network

ICMP – Internet Control Message Protocol

DoS – Denial of Service

QoS – Quality of service

IPv4 – Internet Protocol version 4

IPv6 – Internet Protocol version 6

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росініський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Актуальність теми. В сучасному світі, де мережева безпека стає все більш важливою, захист мережі та контроль доступу до різних ресурсів має вирішальне значення. Міжмережні екрани (firewalls) є ключовим елементом інфраструктури мережі, що забезпечує контроль та фільтрацію мережевого трафіку.

Однак, одним з найважливіших аспектів налаштування міжмережних екранів є правильне створення списків доступу, які визначають, які комунікації дозволені або заборонені між різними мережевими ресурсами. Цей процес формування списків доступу може бути складним і вимагати великої кількості ручної роботи, особливо в разі складних мереж або великої кількості правил.

У зв'язку з цим, розробка програмного застосунку, який автоматизує розрахунок та формування оптимальних списків доступу для міжмережних екранів, має велику актуальність.

Метою кваліфікаційної роботи є розробка програмного застосунку для розрахунку оптимальних списків доступу для міжмережних екранів. Головною метою є автоматизація та спрощення процесу формування списків доступу, що дозволить покращити ефективність та безпеку мережевої інфраструктури.

Конкретні цілі дослідження включають:

1. Дослідження побудови та застосування списків доступу. Вивчення основних принципів та правил формування списків доступу для міжмережних екранів. Аналіз існуючих засобів та методів автоматизованого розрахунку списків доступу.

2. Проектування програмного застосунку. Розробка загальної структури програмного застосунку, включаючи модулі розрахунку та формування списків доступу. Розгляд особливостей формування списків доступу для різних типів міжмережних екранів.

3. Встановлення, налагодження та тестування застосунку. Реалізація програмного застосунку у середовищі Microsoft Visual Studio 2019. Проведення тестування для перевірки правильності розрахунку та ефективності формування списків доступу.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росініський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		8

4. Висновки та рекомендації. Аналіз результатів роботи програмного застосунку та оцінка його ефективності. Формулювання висновків щодо досягнених результатів, переваг та обмежень програмного застосунку. Надання рекомендацій щодо подальшого розвитку та використання програмного застосунку.

Об’єктом дослідження є процес створення застосунку для розрахунку оптимальних списків доступу міжмережного екрану.

Предметом дослідження є програмний застосунок розроблений в Microsoft Visual Studio для розрахунку оптимальних списків доступу для міжмережного екрану.

У процесі виконання кваліфікаційної роботи був розроблений програмний застосунок для розрахунку простих та оптимальних списків доступу для міжмережних екранів.

Завданням кваліфікаційної роботи є:

1. Визначення та класифікація списків доступу;
2. Аналіз алгоритмів роботи та правила формування списків доступу;
3. Аналіз засобів автоматизованого розрахунку та формуванню списків доступу;
4. Проектування загальної структури застосунку;
5. Проектування модулів розрахунку та формування списків доступу;
6. Аналіз особливостей формування списків доступу для різних міжмережних екранів;
7. Встановлення та налагодження застосунку;
8. Тестування роботи застосунку для формування простих списків доступу;
9. Тестування роботи застосунку для оптимальних списків доступу;

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ ПОБУДОВИ ТА ЗАСТОСУВАННЯ СПИСКІВ ДОСТУПУ

1.1. Визначення та класифікація списків доступу

ACL (Access Control List) в контексті мережних пристроїв, зокрема маршрутизаторів, є механізмом для фільтрації трафіку на основі інформації, що міститься в заголовку пакета. Вони використовуються для надання дозволу або заборони проходження пакетів через пристрій на основі визначених критеріїв. Коли мережний трафік перетинає інтерфейс, на якому налаштовано ACL, маршрутизатор послідовно порівнює кожен пакет з правилами ACL, щоб визначити, чи відповідає пакет якому-небудь правилу. Цей процес називається фільтрацією пакетів.

Списки доступу (ACL) є одним із стародавніх та основних механізмів контролю доступу в мережах. Вони з'явилися ще в ранніх версіях операційних систем, коли мережевий трафік був менш складним і розподілений оброблявся на мережевих пристроях, таких як маршрутизатори.

Перші ACL були реалізовані у вигляді простих списків правил, які вказували, які IP-адреси чи підмережі були дозволені або заборонені для комунікації. Такі списки мали обмежену функціональність та могли контролювати тільки джерело або призначення IP-адреси.

З розвитком мереж та протоколів з'явилися більш потужні та гнучкі механізми ACL. Вони стали здатні контролювати не тільки IP-адреси, а й порти TCP/UDP, типи протоколів, стан пакетів та інші характеристики мережевих пакетів.

Разом з розширенням можливостей ACL з'явилися й нові виклики. Більш складні мережі, розподілені сервіси та зростання обсягу мережевого трафіку призвели до потреби у більш розгорнутих та ефективних механізмах контролю доступу.

Сучасні мережеві пристрої, такі як маршрутизатори та комутатори, підтримують розширені ACL, які можуть виконувати складніші функції, такі як

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		10

загальні списки доступу, ACL на основі стану, глибокий інспекційний аналіз пакетів і багато іншого.

Списки доступу (ACL) залишаються важливим елементом безпеки та управління трафіком в сучасних мережах. Вони використовуються для контролю доступу до ресурсів мережі, фільтрації пакетів із заданих джерел та призначень, обмеження сервісів, що надаються, і захисту мережевих пристроїв від шкідливого трафіку.

ACL використовуються для реалізації політик безпеки та управління трафіком на різних рівнях мережевої ієрархії, включаючи зовнішній периметр, внутрішню мережу та сегменти користувачів. Вони можуть бути застосовані на маршрутизаторах, комутаторах, файрволах та інших мережевих пристроях.

ACL відіграють важливу роль у мережевих пристроях, таких як маршрутизатори, комутатори та файрволи. Вони забезпечують механізми для контролю доступу до мережевих ресурсів, забезпечення безпеки мережі та управління трафіком.

У маршрутизаторах ACL використовуються для фільтрації IP-пакетів, що проходять через маршрутизатор, на основі різних критеріїв, таких як джерело та призначення IP-адреси, порти TCP/UDP, типи протоколів тощо. Вони можуть використовуватись для встановлення правил маршрутизації, обмеження доступу до певних мереж або сервісів, а також для захисту мережевих пристроїв від небажаного трафіку.

У комутаторах ACL можуть використовуватись для контролю доступу до портів комутатора, фільтрації пакетів на основі MAC-адресів, VLAN, типів протоколів та інших критеріїв. Вони дозволяють обмежувати комутацію трафіку в межах мережевих сегментів, забезпечуючи безпеку та управління трафіком на рівні комутатора.

У файрволах ACL використовуються для контролю доступу до ресурсів мережі, фільтрації пакетів на основі різних критеріїв та реалізації політик безпеки. Вони можуть бути застосовані на рівні мережевого периметру, контролюючи

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				11
Змн.	Арк.	№ докум.	Підпис	Дата		

вхідний і вихідний трафік між внутрішньою та зовнішньою мережею, а також в межах внутрішньої мережі, для контролю доступу до різних сегментів та сервісів.

ACL використовується для різних цілей контролю доступу до мережі та фільтрації трафіку. Деякі з основних призначень ACL включають:

1. Керування безпекою. ACL використовується для забезпечення безпеки мережі шляхом обмеження доступу до ресурсів і сервісів. Можна налаштувати правила, які дозволяють або забороняють певні типи трафіку, наприклад, блокувати небажані з'єднання, відфільтровувати шкідливі пакети або запобігати атакам типу DoS (Denial of Service).

2. Керування трафіком. ACL дозволяє керувати потоком трафіку в мережі, встановлюючи пріоритети для певних типів пакетів. Наприклад, можна налаштувати ACL, які надають перевагу голосовому або відеотрафіку, забезпечуючи кращу якість обслуговування (QoS) для цих додатків.

3. Фільтрація трафіку. ACL використовується для фільтрації трафіку на основі певних критеріїв, таких як IP-адреси джерела та призначення, порти TCP/UDP, протоколи та інші заголовки пакетів. Це дозволяє обмежувати або дозволяти трафік залежно від встановлених правил, що сприяє поліпшенню продуктивності мережі і забезпеченню відповідності до внутрішніх правил та регуляторних вимог.

4. Віддалений доступ. ACL може бути використано для обмеження доступу до віддалених підключень, таких як Telnet або SSH. Це дозволяє контролювати, хто має право з'єднуватися з мережевими пристроями для адміністрування і забезпечення безпеки мережі.

5. VPN і шифрування. ACL використовується для налаштування параметрів VPN (Virtual Private Network) і встановлення правил шифрування для захисту конфіденційності даних під час передачі через незахищені мережі, такі як Інтернет.

У сучасних мережах існує ряд вимог до безпеки та управління трафіком, що впливають на використання списків доступу. Деякі з цих вимог включають:

– Забезпечення захисту мережі від зовнішніх загроз і несанкціонованого доступу.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				12
Змн.	Арк.	№ докум.	Підпис	Дата		

- Контроль доступу користувачів до ресурсів мережі на основі їх ролей, обов'язків та привілеїв.
- Фільтрація трафіку для обмеження атак на мережеві пристрої та сервіси.
- Забезпечення конфіденційності та цілісності даних, що пересилаються по мережі.
- Моніторинг та аналіз мережевого трафіку для виявлення потенційних загроз та атак.
- Забезпечення відповідності до нормативних вимог та стандартів безпеки.

Ці вимоги впливають на проектування та конфігурацію списків доступу, вимагаючи врахування потреб мережі, політик безпеки та вимог до управління трафіком.

За характеристикою класифікують такі списки доступу ACL:

1. Стандартні ACL;
2. Розширені ACL;
3. Інші типи.

Стандартні списки доступу (ACL) є одним з типів ACL і використовуються для фільтрації IP-трафіку на основі джерела IP-адреси. Основні розглянуті пункти цього типу ACL включають:

Опис та характеристики:

- Стандартні ACL працюють на основі джерела IP-адреси пакета.
- Вони здатні встановлювати правила для відхилення або дозволу на основі адресного діапазону або окремих IP-адрес.
- Стандартні ACL оперують на рівні мережевого шару (шар 3) моделі OSI.

Використання та обмеження:

- Стандартні ACL часто використовуються для простих сценаріїв фільтрації, таких як блокування або дозвіл доступу з конкретних діапазонів IP-адрес.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				13
Змн.	Арк.	№ докум.	Підпис	Дата		

– Однак, вони не здатні враховувати інші параметри, такі як порт або протокол, і тому їх застосування обмежене у випадках, коли потрібно більше точного контролю трафіку.

Розширені списки доступу (ACL) є іншим типом ACL і надають більш гнучкий і точний контроль над фільтрацією трафіку. Основні розглянуті пункти цього типу ACL включають:

Опис та характеристики:

- Розширені ACL можуть використовуватись для фільтрації трафіку на основі джерела та призначення IP-адрес, портів, протоколів та інших параметрів.
- Вони оперують на рівні мережевого та транспортного шарів (шари 3 і 4) моделі OSI.
- Розширені ACL забезпечують більш гнучкий контроль над трафіком, дозволяючи встановлювати детальніші правила фільтрації.

Використання та обмеження:

- Розширені ACL широко використовуються для складніших сценаріїв фільтрації, де потрібен точний контроль над джерелом, призначенням, портами, протоколами тощо.
- Однак, через більшу гнучкість і складність встановлення правил, розширені ACL можуть впливати на продуктивність мережі, і тому важливо уважно проектувати та конфігурувати їх.

Інші типи ACL.

Поза стандартними та розширеними ACL існують інші типи, які можуть бути використані для специфічних потреб мережі. Деякі з цих типів ACL включають:

- Динамічні ACL. Це ACL, які автоматично створюються або змінюються на основі специфічних умов або подій у мережі.
- VLAN ACL. Використовуються для фільтрації трафіку на основі VLAN-ідентифікаторів.
- MAC ACL. Фільтрують трафік на основі MAC-адрес.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		14

– Time-based ACL. Дозволяють встановлювати правила фільтрації, активні лише протягом певного періоду часу.

Використання та обмеження:

– Інші типи ACL можуть бути корисними для специфічних сценаріїв або вимог мережі.

– Важливо уважно вивчати та розуміти характеристики кожного типу ACL та їх можливості, а також враховувати обмеження, пов'язані з використанням цих типів ACL.

Використання ACL може стикатися з рядом проблем та викликів, які варто враховувати при їх конфігурації, впровадженні та підтримці. Деякі з найпоширеніших проблем та викликів включають:

1. Складність конфігурації та управління ACL;
2. Практичні проблеми та виклики, що виникають під час реалізації та підтримки ACL;
3. Забезпечення відповідності та контроль доступу.

Складність конфігурації та управління ACL:

– ACL може мати складну синтаксичну структуру та вимагати детального розуміння мережевих протоколів і параметрів.

– Конфігурація ACL на багатьох мережевих пристроях може бути часозатратною та потребувати ретельного планування.

– Підтримка та управління великими наборами ACL може бути складною і вимагати системи керування доступом.

Практичні проблеми та виклики, що виникають під час реалізації та підтримки ACL:

– Велика кількість правил ACL може впливати на продуктивність мережі, збільшувати затримки та споживання ресурсів пристроїв.

– Некоректно налаштовані ACL можуть блокувати легітимний трафік або допускати несанкціонований доступ, що може призвести до проблем з безпекою та недоступністю послуг.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				15
Змн.	Арк.	№ докум.	Підпис	Дата		

– При великій кількості ACL на мережевих пристроях може виникати складність у відладці, аналізі та зміні правил.

Забезпечення відповідності та контроль доступу:

– Виконання вимог щодо безпеки та управління трафіком може вимагати докладного аналізу потоків даних та розробки складних ACL, що може бути викликом для адміністраторів мережі.

– Важливо забезпечити постійне відслідковування та оновлення правил ACL для відповідності змінюючимся вимогам та політикам безпеки.

Для подолання цих проблем рекомендується:

1. Детально спланувати використання ACL, визначити потреби та політики безпеки перед його конфігурацією.

2. Використовувати стандартизовані імена та описи для правил ACL, щоб полегшити їх розуміння та управління.

3. Регулярно переглядати та аудитувати ACL для виявлення непотрібних або застарілих правил.

4. Використовувати інструменти автоматизації та управління для спрощення конфігурації та моніторингу ACL.

Правильне розуміння та управління проблемами та викликами, пов'язаними з використанням ACL, допоможе забезпечити ефективне та безпечне функціонування мережі.

ACL продовжують розвиватися, пристосовуючись до змін у сфері мережевої безпеки, продуктивності та управління мережею. Ось деякі перспективи та тренди, пов'язані з майбутнім розвитком ACL:

1. Розширення функціональності. Очікується, що ACL будуть набувати нових можливостей та функцій, що розширять їх здатність контролювати доступ та управляти трафіком. Наприклад, можуть бути додані нові типи ACL, які дозволяють керувати трафіком на основі контексту, додаткових атрибутів або стану з'єднання.

2. Інтеграція зі штучним інтелектом та машинним навчанням. Застосування штучного інтелекту та машинного навчання може покращити ефективність та

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		16

точність ACL. Алгоритми машинного навчання можуть допомогти виявити аномальний трафік, ідентифікувати загрози та автоматизувати процес створення та оновлення ACL.

3. Удосконалення управління та аналізу. Очікується, що інструменти управління та аналізу ACL стануть більш потужними та інтуїтивно зрозумілими. Це дозволить адміністраторам мережі швидше та ефективніше налаштовувати, моніторити та аналізувати ACL для забезпечення високої безпеки та продуктивності.

4. Гнучкість та автоматизація. ACL можуть стати більш гнучкими та легкими для конфігурації. Розвиток стандартів та протоколів, які спрощують управління ACL, дозволить швидше реагувати на зміни в мережі та політиках безпеки. Автоматизація процесу створення, встановлення та моніторингу ACL також може значно спростити роботу адміністраторів мережі.

5. Інтеграція з хмарними сервісами. За ростом хмарних сервісів і розподілених мереж очікується, що ACL будуть інтегровані з хмарними платформами та сервісами. Це дозволить контролювати трафік до та з хмарних ресурсів, застосовувати політики безпеки на рівні хмари та забезпечувати єдиноцільний контроль доступу.

Усі ці тренди та вдосконалення покликані покращити безпеку, продуктивність та управління мережею через ACL. Адміністратори мереж повинні бути свідомі цих перспектив та готові до адаптації своїх практик та інфраструктури до нових вимог та можливостей, що надаються ACL.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		17

1.2. Алгоритми роботи та правила формування списків доступу

Алгоритми роботи списків доступу визначають послідовність кроків, які виконуються при обробці пакетів мережевого трафіку на мережевих пристроях. Основною метою є визначення, які пакети будуть прийняті або відхилені залежно від умов, встановлених у списку доступу.

Процес обробки списків доступу зазвичай включає наступні етапи:

1. Отримання пакета. Мережевий пристрій отримує пакет мережевого трафіку.

2. Виявлення протоколу. Проаналізований пакет використовується для виявлення типу протоколу, такого як IP, TCP або UDP.

3. Відповідність критеріям. Пакет порівнюється з правилами, визначеними у списку доступу. Кожне правило містить критерії, такі як джерело, призначення, протокол, порти тощо. Якщо пакет відповідає критеріям, встановленим у правилах, то вживається відповідна дія (дозволити або заборонити).

4. Виконання дії. Залежно від результату порівняння, мережевий пристрій вживає відповідної дії з пакетом. Наприклад, пакет може бути пересланий, заблокований, пропущений або оброблений за іншими правилами.

5. Продовження обробки. Якщо є інші правила у списку доступу, процес переходить до наступного правила та повторюється для наступного пакета.

Важливо пам'ятати, що порядок правил у списку доступу має значення, оскільки вони перевіряються по черзі до знаходження відповідного правила. Тому правила з більш вузькими критеріями (наприклад, конкретне джерело або призначення) зазвичай розміщуються перед загальними правилами.

Синтаксис та формат правил списків доступу визначають спосіб використання ключових параметрів та умов для створення правил, що регулюють рух мережевого трафіку. Точний синтаксис може варіюватися залежно від використовуваного обладнання та програмного забезпечення.

Основні компоненти синтаксису та формату правил ACL включають:

1. Джерело та призначення;

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		18

2. Протокол;
3. Порти;
4. Дія.

Джерело та призначення. Це параметри, що вказують на джерело (відправника) та призначення (одержувача) мережевого трафіку. Можуть використовуватися IP-адреси, підмережі, домени, порти тощо.

Протокол. Вказує на тип протоколу, який буде перевірятися для пакетів. Наприклад, TCP, UDP, ICMP.

Порти. Використовуються для специфікації конкретних портів, що використовуються протоколами TCP або UDP.

Дія. Визначає, яку дію слід вжити з пакетом, якщо він відповідає заданим критеріям. Це може бути дозвіл або заборона (permit або deny), а також додаткові дії, такі як пересилання на інтерфейс, видалення заголовку тощо.

Варто відзначити, що синтаксис та формат можуть залежати від типу ACL, який використовується, таких як стандартні ACL, розширені ACL або інші специфічні типи. Крім того, різні вендори та мережеві пристрої можуть мати власні нюанси та особливості в синтаксисі та форматі правил ACL.

Формування правил списків доступу є важливим етапом в налаштуванні ACL. Під час створення правил слід враховувати вимоги та політики мережі, а також розглядати реальні сценарії та приклади для досягнення потрібного рівня контролю доступу.

Основні кроки формування правил ACL включають:

1. Критерії та параметри для визначення правил;
2. Розробка правил для контролю доступу на основі вимог та політик мережі;
3. Розгляд реальних сценаріїв та прикладів формування правил.

Критерії та параметри для визначення правил. Це включає вибір джерела, призначення, протоколу, портів та інших важливих параметрів, які дозволяють ідентифікувати пакети, які потрібно обробити.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		19

Розробка правил для контролю доступу на основі вимог та політик мережі.

Правила ACL повинні відповідати потребам та вимогам мережі. Наприклад, якщо в мережі необхідно обмежити доступ до певних ресурсів з певних джерел, правила повинні бути налаштовані відповідно.

Розгляд реальних сценаріїв та прикладів формування правил. Для ефективного формування правил ACL, корисно розглянути реальні сценарії та приклади використання. Це можуть бути приклади контролю доступу на основі IP-адрес, портів, протоколів та інших критеріїв.

Виконання цих кроків допоможе забезпечити ефективність та точність правил списків доступу, що відповідають потребам мережі та політикам безпеки.

Розташування списків доступу у мережевих пристроях, таких як маршрутизатори, комутатори, файрволи та інші, має значний вплив на ефективність та безпеку мережі. Правильне розміщення списків доступу допомагає досягти оптимального контролю трафіку і забезпечити безпеку мережевих ресурсів.

Основні аспекти, які слід враховувати при розміщенні списків доступу.

Вплив розташування списків доступу на ефективність та безпеку мережі. Розміщення списків доступу на певних точках мережі може мати великий вплив на продуктивність та безпеку. Наприклад, розміщення ACL на ранніх етапах мережі може допомогти зменшити непотрібний трафік та покращити загальну ефективність мережі.

Рекомендації щодо розташування списків доступу на маршрутизаторах, комутаторах, файрволах тощо. Для досягнення оптимального контролю доступу рекомендується розташовувати списки доступу на мережевих пристроях з урахуванням потреб мережі. Наприклад, на маршрутизаторах можна розмістити списки доступу для контролю маршрутизації, а на файрволах – для забезпечення безпеки мережі.

Врахування трафіку в разі використання IPv4 або IPv6. Розташування списків доступу повинно враховувати особливості трафіку, якщо мережа використовує протоколи IPv4 або IPv6. Врахування типу адресування та

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		20

особливостей кожного протоколу допоможе забезпечити ефективний контроль доступу.

Управління розміщенням списків доступу. Розміщення списків доступу не є статичним процесом і може вимагати періодичного оновлення та налаштування. Управління розміщенням списків доступу забезпечить їхню актуальність та безпеку.

Врахування цих аспектів та дотримання рекомендацій щодо розміщення списків доступу допоможе забезпечити ефективний контроль трафіку та безпеку мережі.

Комбіновані списки доступу є розширенням стандартних та розширених списків доступу, які дозволяють використовувати більш складні умови та правила для контролю доступу. Вони надають можливість комбінувати різні типи правил та використовувати логічні операції для задання умов доступу.

Основні аспекти, які слід розглянути щодо використання комбінованих списків доступу.

Опис комбінованих списків доступу та їхньої синтаксис. Комбіновані списки доступу складаються з набору правил, де кожне правило має визначені умови та дії. Синтаксис комбінованих списків доступу залежить від конкретної реалізації та виробника мережевих пристроїв, але вони часто базуються на розширених списках доступу.

Переваги використання комбінованих списків доступу. Використання комбінованих списків доступу дозволяє більш гнучкий та точний контроль доступу до мережевих ресурсів. Завдяки можливості комбінувати умови та використовувати логічні операції, можна створювати складніші правила для задання специфічних вимог до доступу.

Недоліки використання комбінованих списків доступу. Використання комбінованих списків доступу може призвести до складнішої конфігурації та управління. При створенні складних комбінованих правил необхідно бути уважним, щоб уникнути помилок та непередбачених наслідків.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				21
Змн.	Арк.	№ докум.	Підпис	Дата		

Розгляд прикладів застосування комбінованих списків доступу. Комбіновані списки доступу можуть бути застосовані в різних сценаріях, таких як фільтрація трафіку, обмеження доступу до певних ресурсів, реалізація політик безпеки тощо. Приклади використання комбінованих списків доступу можуть включати заборону або дозвіл на певні протоколи, порти, IP-адреси, а також задання специфічних умов для доступу до мережевих послуг.

1.3. Засоби автоматизованого розрахунку та формуванню списків доступу

В сучасних мережевих технологіях засоби автоматизованого розрахунку та формування списків доступу відіграють важливу роль у налаштуванні та контролі доступу до різних мережевих пристроїв. За допомогою цих засобів інженери мереж можуть ефективно керувати рухом даних в мережі, встановлюючи правила та обмеження для передачі пакетів.

Загальною метою засобів автоматизованого розрахунку та формування списків доступу є спрощення процесу налаштування контролю доступу в мережах та забезпечення швидкого та точного розрахунку необхідних параметрів для створення списків доступу. Ці засоби дають можливість інженерам мереж ефективно використовувати ресурси мережі, забезпечуючи контроль трафіку на основі визначених правил та обмежень.

Засоби автоматизованого розрахунку та формування списків доступу допомагають виконувати складні задачі налаштування з високою швидкістю та точністю, що робить їх важливими компонентами при розгортанні та управлінні мережевими інфраструктурами. Використання таких засобів дозволяє зменшити ймовірність помилок під час налаштування та забезпечити консистентність правил контролю доступу на різних пристроях мережі.

Розроблений засіб автоматизованого розрахунку та формування списків доступу є розширеним інструментом, який пропонує широкі функціональні можливості для налаштування контролю доступу в мережах. Він підтримує

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		22

автоматичний розрахунок параметрів та генерацію списків доступу для IPv4, IPv6, а також підтримує різні типи списків доступу, включаючи нумеровані та іменовані списки.

Переваги розробленого засобу порівняно з іншими аналогами включають:

1. Універсальність. Наш засіб підтримує різні виробники мережевого обладнання, що дозволяє легко налаштовувати списки доступу для різних пристроїв, використовуючи один інструмент.

2. Підтримка IPv4 і IPv6. Засіб дозволяє налаштовувати списки доступу як для IPv4, так і для IPv6 мереж, забезпечуючи гнучкість в налаштуванні контролю доступу для різних типів мережевих протоколів.

3. Підтримка нумерованих та іменованих списків доступу. Засіб надає можливість створювати як нумеровані списки доступу, які ідентифікуються за номерами, так і іменовані списки доступу, що дозволяє зручно керувати трафіком.

Наш засіб відзначається своєю універсальністю, підтримкою IPv6 та можливістю створення нумерованих та іменованих списків доступу, що робить його зручним та потужним інструментом для автоматизації налаштування списків доступу.

Розроблений засіб автоматизованого розрахунку та формування списків доступу забезпечує повну підтримку як IPv4, так і IPv6-адресації, що дозволяє його використовувати для налаштування контролю доступу в різних типах мережевих протоколів.

Основні можливості засобу щодо роботи з IPv4- та IPv6-адресацією включають:

1. Розрахунок IP-адрес та масок підмереж;
2. Формування правил доступу;
3. Перевірка сумісності з мережевими пристроями.

Розрахунок IP-адрес та масок підмереж. Засіб автоматично розраховує IP-адреси та маски підмереж для використання в списку доступу. Він забезпечує введення IP-адреси або діапазону IP-адрес із зручними форматами, а також дозволяє вказувати бітові маски або префікси підмереж.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Формування правил доступу. Засіб дозволяє встановлювати правила доступу для як IPv4, так і IPv6-адресації. Це включає визначення джерела та призначення IP-адрес, протоколів, портів та інших параметрів, що використовуються для контролю трафіку в мережі.

Перевірка сумісності з мережевими пристроями. Засіб підтримує різні виробники мережевого обладнання, що дозволяє перевірити сумісність з наявними пристроями в мережі. Він забезпечує можливість генерації списків доступу, які відповідають вимогам різних пристроїв, включаючи підтримку IPv4 та IPv6-протоколів.

Засіб автоматизованого розрахунку та формування списків доступу забезпечує гнучкість в налаштуванні контролю доступу для різних типів мережевих протоколів. Він дозволяє інженерам мереж ефективно використовувати ресурси мережі, забезпечуючи контроль трафіку на основі визначених правил та обмежень, як для IPv4, так і для IPv6-мереж.

Нумеровані та іменовані списки доступу є важливою функціональною складовою засобу автоматизованого розрахунку та формування списків доступу. Вони використовуються для організації та управління правилами контролю доступу в мережах.

Нумеровані списки доступу (Numbered Access Control Lists) визначаються за допомогою номерів, які призначаються кожному списку. Кожне правило в нумерованому списку доступу має унікальний номер, за яким його можна ідентифікувати. Нумеровані списки доступу використовуються для фільтрації трафіку на основі вказаних правил.

Іменовані списки доступу (Named Access Control Lists) використовують поняття імен для ідентифікації списку. Кожен іменований список доступу має унікальне ім'я, за яким його можна викликати та використовувати в мережі. Іменовані списки доступу надають зручність у керуванні трафіком та можуть бути легко змінені або видалені.

Переваги використання нумерованих та іменованих списків доступу включають:

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		24

1. Легке визначення та ідентифікація правил;
2. Гнучкість в керуванні трафіком;
3. Легкість управління та змін;
4. Читабельність та розуміння.

Легке визначення та ідентифікація правил. Нумеровані списки доступу можна швидко ідентифікувати за їхніми номерами, тоді як іменовані списки доступу можна викликати за допомогою їхніх унікальних імен. Це спрощує визначення та керування правилами контролю доступу.

Гнучкість в керуванні трафіком. Нумеровані та іменовані списки доступу дозволяють гнучко налаштовувати правила фільтрації трафіку. Інженери мереж можуть встановлювати правила, що відповідають конкретним вимогам мережі та забезпечують потрібний рівень безпеки.

Легкість управління та змін. Іменовані списки доступу дозволяють просте перейменування, зміну або видалення списків за допомогою їхніх імен. Це полегшує управління списками та забезпечує швидкі зміни в правилах контролю доступу.

Читабельність та розуміння. Як нумеровані, так і іменовані списки доступу можуть бути легко читабельними та зрозумілими для адміністраторів мережі. Правила контролю доступу можуть бути документовані та зрозуміло описані, спрощуючи розуміння їх функціональності.

Використання нумерованих та іменованих списків доступу надає зручність, гнучкість та ефективність у налаштуванні контролю доступу в мережах. Вони є потужними інструментами для управління трафіком та забезпечення безпеки мережі.

Для порівняння з розробленим засобом автоматизованого розрахунку та формування списків доступу буде використано аналогічний інструмент DragonElf.NET, доступний за адресою: <https://www.dragonelf.net/cidr-acl/acl+iptables-form.php?formtype=IOS>. Обидва засоби мають схожі функціональні можливості та призначення.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		25

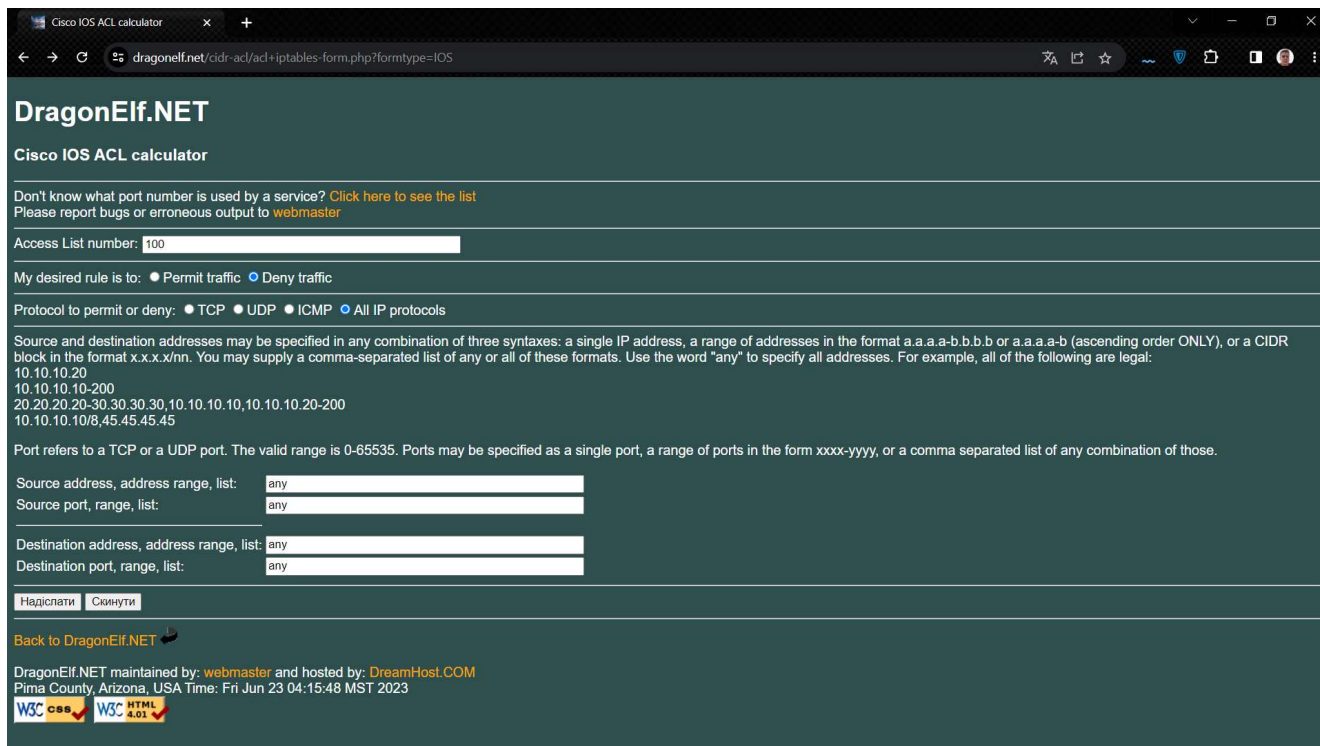


Рисунок 1.1 – інтерфейс інструменту DragonElf.NET

Аналізуючи існуючі засоби автоматизованого розрахунку та формування списків доступу, можна виділити такі переваги розробленого засобу порівняно з аналогами:

Інтуїтивний і простий інтерфейс. Розроблений засіб має зручний та простий у використанні інтерфейс. Користувачам не потрібно мати глибокі знання мережевих технологій для його використання. Він дозволяє швидко та ефективно налаштовувати правила контролю доступу.

Підтримка IPv4 та IPv6. Розроблений засіб підтримує як IPv4, так і IPv6 адресацію, що дозволяє ефективно працювати з різними типами мережевих протоколів. Це важливо в контексті переходу до IPv6 та забезпечення сумісності з різними поколіннями мережевих пристроїв.

Гнучкість в налаштуванні контролю доступу. Розроблений засіб дозволяє гнучко налаштовувати правила контролю доступу для різних типів мережевих протоколів. Це дозволяє адаптувати його до конкретних потреб і вимог мережі.

Можливість використання нумерованих та іменованих списків доступу. Розроблений засіб підтримує як нумеровані, так і іменовані списки доступу. Це

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		26

надає користувачам гнучкість та зручність у використанні та управлінні списками правил контролю доступу.

Автоматизація розрахунку та формування списків доступу. Розроблений засіб надає можливість автоматизованого розрахунку та формування списків доступу на основі заданих параметрів. Це дозволяє значно економити час та зусилля при створенні та налаштуванні правил контролю доступу.

Загалом, розроблений засіб автоматизованого розрахунку та формування списків доступу має численні переваги порівняно з аналогами, забезпечуючи простоту використання, гнучкість налаштувань та підтримку різних мережевих протоколів.

Розглянемо конкретні сценарії та приклади використання розробленого засобу автоматизованого розрахунку та формування списків доступу.

Контроль доступу до мережевих ресурсів. Засіб може бути використаний для налаштування правил контролю доступу до мережевих ресурсів, наприклад, серверів, маршрутизаторів або фаїрволів. Користувач може визначити дозволений або заборонений доступ для певних IP-адрес, діапазонів IP-адрес або мереж.

Фільтрація мережевого трафіку. Засіб дозволяє встановлювати правила фільтрації мережевого трафіку для контролю та обмеження передачі певних типів даних або протоколів. Наприклад, користувач може налаштувати фільтри для блокування певних портів або типів пакетів.

Захист від зловживання та зламів. Засіб може допомогти у захисті мережі від зловживання та зламів, встановлюючи правила доступу, які обмежують підозрілий трафік або блокують IP-адреси зі списків відомих атакуючих джерел.

Приклад використання засобу. Наприклад, припустимо, що користувач бажає дозволити доступ лише з обмеженого діапазону IP-адрес до певного сервера. Засіб дозволить йому встановити правило, яке дозволяє тільки зазначені IP-адреси підключатися до сервера, блокуючи доступ з інших адрес. Користувач зможе швидко та зручно налаштувати це правило з використанням розробленого засобу.

Ці приклади демонструють різноманітність та практичну цінність засобу автоматизованого розрахунку та формування списків доступу. Він може бути

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		27

використаний для різних сценаріїв контролю доступу та забезпечення безпеки в мережевому середовищі.

Засіб автоматизованого розрахунку та формування списків доступу має вплив як на ефективність, так і на безпеку мережі.

Вплив на ефективність мережі. Засіб дозволяє налаштовувати правила контролю доступу та фільтрації трафіку ефективним способом. Це сприяє оптимальному використанню мережевих ресурсів і підвищує продуктивність. Автоматизований розрахунок дозволяє швидко та точно налаштовувати правила доступу, спрощуючи процес управління мережевими політиками.

Вплив на безпеку мережі та контроль доступу. Засіб забезпечує ефективний контроль доступу до мережевих ресурсів, що поліпшує безпеку мережі. Користувач може точно визначати, які користувачі, IP-адреси або мережі мають дозвіл або обмеження доступу до різних ресурсів. Це допомагає запобігти несанкціонованому доступу, зловживанням або атакам на мережу.

Аналіз впливу засобу на ефективність та безпеку мережі є важливим етапом при впровадженні такого засобу. Варто враховувати потреби мережі, обсяг трафіку, складність правил доступу та наявність ресурсів для обробки таких правил. Детальний аналіз може допомогти забезпечити оптимальну ефективність та безпеку мережі з використанням розробленого засобу автоматизованого розрахунку та формування списків доступу.

1.4. Висновки до розділу 1

У цьому розділі було проведено аналіз побудови та застосування списків доступу (ACL). Оглянуто історію використання ACL, проаналізували сучасний стан їх застосування і розглянули класифікацію та характеристики різних типів ACL. Також розглянуто проблеми і виклики, що виникають у використанні ACL, та розглянули перспективи їх розвитку.

Також детально розглянуто алгоритми роботи та правила формування списків доступу, включаючи синтаксис і формат правил ACL. Розглянули процес

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		28

формування правил і розміщення списків доступу у мережевих пристроях, а також вивчили використання комбінованих списків доступу.

Додатково, було досліджено засоби автоматизованого розрахунку та формування списків доступу. Вивчили їх загальну роль у сучасних мережевих технологіях, особливості розробленого засобу та його функціональні можливості. Порівняли розроблений засіб з існуючими аналогами, а також розглянули приклади його застосування.

Крім того, було проаналізовано вплив засобу автоматизованого розрахунку та формування списків доступу на ефективність та безпеку мережі. Встановлено, що засіб може покращити ефективність мережі через оптимізацію використання ресурсів та спрощення процесу управління мережевими політиками. Водночас, засіб забезпечує ефективний контроль доступу та підвищує безпеку мережі, запобігаючи несанкціонованому доступу та атакам.

Висновком можна сказати, що використання списків доступу є важливим елементом управління мережевою безпекою та контролю доступу. Автоматизовані засоби розрахунку та формування списків доступу пропонують швидке та точне налаштування правил, що спрощує управління мережевими політиками та забезпечує оптимальну ефективність та безпеку мережі.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієвський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		29

РОЗДІЛ 2. ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ РОЗРАХУНКУ ОПТИМАЛЬНИХ СПИСКІВ ДОСТУПУ

2.1. Проєктування загальної структури застосунку

Наш застосунок призначений для розрахунку оптимальних списків доступу (ACL) для мережевих пристроїв. Він надає зручний інтерфейс користувача для введення параметрів та генерації відповідних команд ACL, які можуть бути використані для налаштування мережевих пристроїв різних виробників, таких як Cisco, Huawei та MikroTik.

Основними функціями застосунку є:

1. Налаштування параметрів:

- Вибір виробника мережевого пристрою: користувач може обрати виробника зі списку доступних опцій (2.1а).
- Вибір версії IP: користувач може вибрати версію IP-протоколу (IPv4 або IPv6), для якої будуть генеруватись команди ACL (2.1б).
- Вибір типу списків доступу: користувач може обрати між нумерованими або іменованими списками доступу (2.1в).

Налаштування

Виробник (а)

☒ Cisco ☐ Huawei ☐ MikroTik

Версія IP (б)

☒ IPv4 ☐ IPv6

Типи списків доступу (в)

☒ Нумеровані ☐ Іменовані

Зберегти

Рисунок 2.1 – Налаштування параметрів

2. Введення вхідних даних (2.2а):

- Введення номера чи імені ACL: користувач може ввести номер або ім'я ACL, в залежності від обраного типу списку доступу (доступні опції залежать від виробника мережевого пристрою).
- Вибір правила: користувач може вибрати правило, яке буде застосовуватись до ACL.
- Вибір протоколу: користувач може вибрати протокол, який буде застосовуватись до ACL
- Вибір типу трафіку: користувач може вибрати тип трафіку, який буде враховуватись при генерації команд ACL (доступні опції залежать від виробника мережевого пристрою).

3. Введення параметрів ACL (2.2б):

- Введення адреси джерела: користувач може ввести IP-адресу, діапазон адрес, список або завантажити її з текстового файлу.
- Введення порта джерела: користувач може ввести номер порта, діапазон портів, список або завантажити його з текстового файлу.
- Введення адреси призначення: користувач може ввести IP-адресу, діапазон адрес, список або завантажити її з текстового файлу.
- Введення порта призначення: користувач може ввести номер порта, діапазон портів, список або завантажити його з текстового файлу.

4. Обчислення команд ACL (2.2в):

- Кнопка «Обчислити» виконує розрахунок та генерацію команд ACL з введених параметрів.
- Результати виводяться у вікні, де користувач може переглянути сформовані команди ACL.

5. Збереження результатів (2.2в):

- Кнопка «Зберегти ACL у файл» дозволяє зберегти результати, сформовані команди ACL, у текстовий файл.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				31
Змн.	Арк.	№ докум.	Підпис	Дата		

Рисунок 2.2 – Головний екран

Цілі нашого застосунку:

- Забезпечити користувачу зручний та інтуїтивно зрозумілий інтерфейс для налаштування списків доступу.
- Автоматизувати процес генерації команд ACL для різних мережевих пристроїв.
- Забезпечити можливість збереження результатів для подальшого використання.

Після введення вхідних даних та обчислення команд ACL, користувач зможе скопіювати результати або зберегти їх у текстовому файлі для подальшого використання при конфігуруванні мережевих пристроїв.

У нашому застосунку існують декілька основних компонентів, які взаємодіють між собою, щоб забезпечити необхідний функціонал:

1. Форма налаштування (Settings Form):

- Відповідає за відображення налаштувань застосунку, таких як виробник, версія IP та тип списків доступу.

- Містить кнопку «Зберегти», яка передає обрані налаштування в головну форму (Main Form).

2. Головна форма (Main Form):

- Відображає основний інтерфейс застосунку, де користувач може вводити параметри для розрахунку списків доступу.
- Містить кнопку «Налаштування», яка відкриває форму налаштування (Settings Form).
- Містить текстові поля, які відображають обраного виробника та протокол.
- Містить поля для введення даних, такі як номер чи ім'я ACL, правило, трафік, протокол, адреса джерела та призначення, порт джерела та призначення.
- Містить кнопку «Обчислити», яка запускає розрахунок та генерацію команд ACL.
- Містить вікно, де виводяться результати генерації команд ACL.
- Містить кнопку «Зберегти ACL у файл», яка дозволяє зберегти результати в текстовий файл.

3. Модуль розрахунку та формування списків доступу:

- Відповідає за обчислення та генерацію команд ACL на основі введених параметрів користувачем.
- Отримує вхідні дані з головної форми (Main Form) і виконує необхідні розрахунки для формування команд ACL.
- Повертає результати генерації команд ACL у вигляді текстового списку.

Взаємодія між компонентами відбувається наступним чином:

- Користувач відкриває форму налаштування (Settings Form) з головної форми (Main Form).
- Користувач обирає налаштування, які потрібно змінити, і натискає кнопку «Зберегти».

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		33

- Обрані налаштування передаються з форми налаштування до головної форми.
- Користувач вводить вхідні дані та обирає необхідні параметри на головній формі.
- Користувач натискає кнопку «Обчислити», що запускає модуль розрахунку та формування списків доступу.
- Модуль отримує вхідні дані з головної форми і виконує необхідні розрахунки.
- Результати генерації команд ACL повертаються до головної форми та відображаються у вікні.
- Користувач має можливість скопіювати результати або зберегти їх у текстовому файлі за допомогою кнопки «Зберегти ACL у файл».

Ця взаємодія компонентів забезпечує коректну роботу застосунку та забезпечує зручний інтерфейс для користувача.

2.2. Проектування модулів розрахунку та формування списків доступу

У цьому розділі детально розглядається процес проектування модулів, які відповідають за розрахунок та формування списків доступу для нашого програмного застосунку. Один з важливих етапів цього процесу – аналіз вхідних даних та взаємодія з користувачем. Для ефективного розрахунку та формування списків доступу, необхідно чітко зрозуміти, які саме дані потрібно вводити та які формати запису доступні.

Наша програма використовує наступні типи вхідних даних для розрахунку списків доступу:

1. Номер чи ім'я ACL – це ідентифікатор, який визначає конкретний список доступу, до якого будуть додаватись правила. В залежності від обраного типу списків доступу (нумеровані або іменовані), користувач може вводити номер або ім'я ACL.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		34

2. Правило – це компонент, що визначає дію, яку необхідно застосувати до певного трафіку в списку доступу.

3. Трафік – це характеристика певного потоку даних, до якого застосовуються правила зі списку доступу. Він може бути визначений за допомогою параметрів, таких як джерело, призначення, протокол, порти та інші.

4. Протокол – це тип мережевого протоколу, який використовується для комунікації між джерелом і призначенням. Можливі протоколи, які підтримуються нашим застосунком, включають TCP, UDP, ICMP та інші.

5. Адреса джерела та призначення – це мережеві адреси, які вказують на початкове та кінцеве місце, між якими відбувається передача даних.

6. Порт джерела та призначення – це номери портів, які використовуються для ідентифікації конкретних додатків або служб, які спілкуються між собою.

Тепер детальніше розглянемо кожен з них.

Номер чи ім'я ACL. Номер чи ім'я ACL – це ідентифікатор, що визначає конкретний список доступу, до якого будуть додаватися правила. Це поле недоступне, якщо в налаштуваннях було обрано виробника MikroTik.

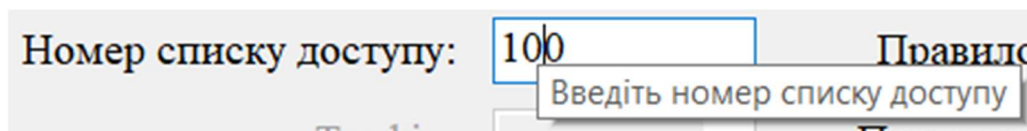


Рисунок 2.3 – Поле вводу номера чи ім'я ACL

Що стосується іменованих списків доступу, то для них немає обмежень. Користувач може вільно вводити імена, які найкраще відповідають їх потребам.

Підказка, яка з'являється при наведенні на поле вводу номера чи імені ACL, може надати користувачеві додаткову інформацію щодо обмежень, які встановлені для конкретного виробника або типу списку доступу. Наприклад, підказка може відображати текст, який вказує на обмеження Cisco та Huawei для нумерованих списків доступу, а також на відсутність обмежень для іменованих списків доступу.

З урахуванням вищезазначених обмежень та підказок, користувач зможе вводити номер чи ім'я ACL відповідно до вимог конкретного виробника та типу списку доступу, що сприятиме правильному функціонуванню застосунку.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росініський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Правило. Для визначення правила користувач може вибрати один з доступних варіантів або ввести власне правило вручну.

The image shows two side-by-side screenshots of a configuration interface. On the left, the 'Правило:' (Rule) dropdown is set to 'permit', and the 'Протокол:' (Protocol) dropdown shows 'permit' and 'deny' as options. On the right, the 'Правило:' dropdown is set to 'accept', and the 'Протокол:' dropdown shows 'accept' and 'drop' as options.

Рисунок 2.4 – Поле вибору/вводу правила

Якщо виробником, обраним у налаштуваннях, є Cisco або Huawei, то доступні варіанти для правила – "permit" та "deny". Вибір "permit" дозволяє пропускати трафік, а "deny" – блокувати його.

У випадку, коли виробником є MikroTik, доступні варіанти для правила – "accept" та "drop". Вибір "accept" дозволяє приймати трафік, а "drop" – відкидати його.

Крім варіантів, які надаються за замовчуванням, користувач також має можливість вручну ввести своє правило в поле. Це дозволяє більш гнучко налаштовувати список доступу та використовувати власні кастомні правила.

Таким чином, користувач має різноманітні варіанти вибору або введення правила в залежності від виробника, що дозволяє налаштовувати роботу застосунку згідно зі своїми потребами та вимогами безпеки мережі.

При наведенні курсору на поле вводу правила, з'являється підказка, що надає короткий опис дій, які виконуються при виборі певних варіантів. Ці підказки допомагають користувачеві краще розуміти вплив вибору правила на обробку трафіку та допомагають зробити відповідний вибір відповідно до вимог та цілей налаштування списків доступу.

Трафік. Трафік – це поле, яке визначає напрямок, до якого застосовуються правила списку доступу. Це поле доступне лише у випадку, коли в налаштуваннях було обрано виробника MikroTik.

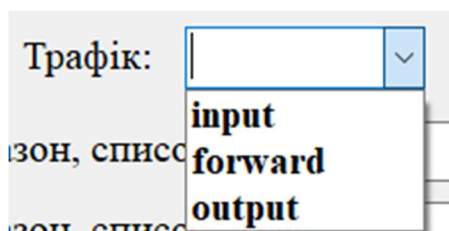


Рисунок 2.5 – Поле вибору/вводу трафіка

Доступні варіанти вибору для поля «Трафік» включають:

1. «input» – якщо обрано «input», то правила списку доступу будуть застосовуватись до вхідного трафіку, що надходить на пристрій.
2. «forward» – при виборі «forward», правила списку доступу будуть застосовуватись до трафіку, який пропускається через пристрій та передається на інші пристрої.
3. «output» – якщо обрано «output», то правила списку доступу будуть застосовуватись до вихідного трафіку, який виходить з пристрою.

Крім вибору з варіантів, також є можливість ввести трафік вручну у це поле, якщо жоден з варіантів не відповідає конкретним потребам.

При наведенні на поле «Трафік» з'являється підказка, яка надає короткий опис доступних варіантів. Ця підказка допомагає користувачеві краще зрозуміти значення кожного варіанту та вибрати відповідний варіант залежно від вимог та потреб в налаштуванні списків доступу. *Протокол.* Протокол – це поле, яке визначає тип протоколу, до якого будуть застосовуватись правила списку доступу. У цьому полі користувач має можливість ввести власний варіант протоколу або обрати один з запропонованих варіантів.

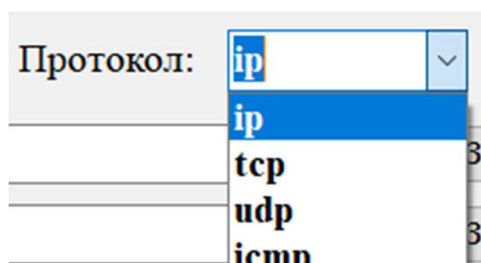


Рисунок 2.6 – Поле вибору/вводу протоколу

Якщо йому відомий конкретний протокол, який він бажає використовувати, він може ввести його вручну. Це дає користувачеві гнучкість у виборі протоколу, що відповідає його потребам.

Крім того, він може обрати протокол зі списку запропонованих варіантів. Цей список містить найпоширеніші протоколи, які можуть використовуватись у списку доступу. Вибір з цього списку спрощує введення правильного протоколу та забезпечує правильну ідентифікацію трафіку.

При наведенні на поле «Протокол» з'являється підказка, яка нагадує користувачеві, що він має можливість обрати протокол зі списку або ввести свій варіант. Ця підказка надає зрозумілу інформацію та спрощує процес вибору та введення протоколу у поле.

Адреса джерела та призначення. Адреса джерела та призначення – це поле, в якому користувач може вказати адреси або мережі, що використовуються для визначення джерела та призначення трафіку в правилах списку доступу. Це поле приймає різні формати запису адрес та мереж залежно від обраного типу адреси (IPv4 або IPv6).

The screenshot shows a configuration window with four input fields, each followed by a 'Завантажити' (Load) button. The fields are:

- Адреса джерела, діапазон, список: (Source address, range, list) with 'any' entered.
- Порт джерела, діапазон, список: (Source port, range, list) with 'any' entered.
- Адреса призначення, діапазон, список: (Destination address, range, list) with 'any' entered.
- Порт призначення, діапазон, список: (Destination port, range, list) with 'any' entered.

 A dropdown menu is open for the first field, showing 'Приклади допустимих форматів:' (Examples of acceptable formats:) with the following text:

- 0.0.0.0, 0.0.0.5
- 0.0.0.0-0.0.0.5, 0.0.0.10
- 0.0.0.0-15, 10.0.0.0/24
- 192.168.1.0 255.255.255.0, 192.168.3.1-20

Рисунок 2.7 – Поля вводу адрес джерела та призначення з підказкою і кнопки «завантажити»

Якщо обрано тип адреси IPv4, то користувач може використовувати такі формати запису:

- Окрема адреса: можна ввести окрему IPv4-адресу у вигляді чотирьох чисел, розділених крапками (наприклад, 192.168.0.1).
- Діапазон адрес: можна вказати діапазон IPv4-адрес, використовуючи дві окремі адреси, розділені дефісом (наприклад, 192.168.0.1-192.168.0.10).

– Мережа з префіксом: можна вказати IPv4-мережу з префіксом, використовуючи формат адреси та числа після косої риски, що вказує на довжину префіксу (наприклад, 192.168.0.0/24).

– Мережа з маскою: можна вказати IPv4-мережу з маскою, використовуючи формат адреси та маски у вигляді числа (наприклад, 192.168.0.0 255.255.255.0).

Також можна вказати адреси або мережі одночасно, розділяючи їх комою. Наприклад, користувач може ввести кілька окремих адрес, діапазонів, мереж з префіксом або мереж з маскою, розділивши їх комами. Це дозволяє йому швидко вказати багато вхідних даних без повторення поля вводу для кожного окремого значення.

Якщо обрано тип адреси IPv6, користувач може використовувати ті ж самі формати запису, що й для IPv4, за винятком мережі з маскою, яка не доступна для IPv6.

При наведенні на поле «Адреса джерела та призначення» з'являється підказка, яка показує приклади доступних форматів введення. Ця підказка допомагає користувачеві правильно ввести адреси або мережі у поле.

Також, поруч з полем є кнопка «Завантажити», яка дає змогу завантажити список адрес з обраного текстового файлу. Користувач може підготувати список адрес у текстовому файлі, де кожен абзац містить одну адресу або мережу. При завантаженні, кожен абзац у файлі буде замінений комою, щоб розділити адреси у полі.

Ці можливості форматування та завантаження допомагають спростити введення та управління списком адрес та мереж у застосунку.

Порт джерела та призначення. Поле «Порт джерела та призначення» дозволяє користувачу ввести окремий порт або діапазон портів. Він може вказати кілька портів або діапазонів, розділивши їх комами. Це дозволяє швидко визначити багато портів без необхідності повторення поля вводу для кожного окремого значення.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Адреса джерела, діапазон, список:	any	Завантажити
Порт джерела, діапазон, список:	any	Завантажити
Адреса призначення, діапазон, список:	any	Завантажити
Порт призначення, діапазон, список:	any	Завантажити

Діапазон: 0-65535
 Приклад допустимого формату: 10-20, 36, 50-100

Рисунок 2.8 – Поля вводу портів джерела та призначення з підказкою і кнопки «завантажити»

Наприклад, користувач може ввести окремі порти, наприклад, 80, 443, 8080, або діапазони портів, такі як 1000-2000, 5000-6000. Він також може комбінувати окремі порти та діапазони, розділяючи їх комами.

Для зручності, при наведенні на поле з'являється підказка, яка показує приклади доступних форматів введення, що допомагає правильно вказати порти або діапазони.

Крім того, після полем знаходиться кнопка «Завантажити», яка дозволяє завантажити список портів з обраного текстового файлу. У файлі користувач може вказати порти через абзац, і при завантаженні кожен абзац буде замінений комою, щоб зручно відображати різні порти.

Алгоритм розрахунку команд ACL дозволяє генерувати відповідні команди з урахуванням вхідних даних та налаштувань користувача, забезпечуючи правильне формування списків доступу.

Перед початком формування команд ACL, спочатку перевіряється виробник мережного пристрою, для якого необхідно згенерувати команди. Це важливо, оскільки різні виробники можуть використовувати різні формати команд ACL та їх параметрів. Далі проводиться зчитування та обробка вхідних даних, таких як адреси та порти джерела та призначення.

Зчитування адрес джерела та призначення передбачає проведення перевірки формату введених адрес. Якщо формат адреси некоректний, програма повідомляє користувача про помилку. Після перевірки формату, адреси та мережі перетворюються у діапазони адрес та сортуються за початковою адресою.

Послідовні або накладені діапазони адрес об'єднуються. Потім з них формуються мережі з оберненою маскою, якщо йде робота з IPv4, а якщо IPv6 – формуються мережі з префіксом

Схожий підхід застосовується і до портів джерела та призначення. Порти перевіряються на відповідність формату та сортуються за початковим значенням у разі, якщо вказано діапазон. Також проводиться об'єднання послідовних або накладених діапазонів портів.

Після обробки вхідних даних розпочинається формування відповідних команд ACL на основі отриманих результатів. У цьому кроці враховуються вхідні дані та налаштування користувача, що дозволяє згенерувати правильні команди ACL для заданого мережного пристрою.

Надамо наочне уявлення про алгоритм розрахунку команд ACL, розглянемо обробку вхідних даних для полів адреси джерела та адреси призначення.

```

List<string> result = new List<string>();
string[] elements = input.Split(new[] { "," }, StringSplitOptions.RemoveEmptyEntries);
foreach (string element in elements)
{
    string trimmedElement = element.Trim();
    if (IsValidIPAddress(trimmedElement))
    {
        result.Add(ExpandIPv6Address(trimmedElement));
    }
    else if (IsValidAddressRange(trimmedElement))
    {
        string range = FormatAddressRange(trimmedElement);
        result.Add(range);
    }
    else if (IsValidNetworkWithMask(trimmedElement))
    {
        string range = ConvertNetworkWithMaskToRange(trimmedElement);
        result.Add(range);
    }
    else if (IsValidNetworkWithPrefix(trimmedElement))
    {
        string range = ConvertNetworkWithPrefixToRange(trimmedElement);
        result.Add(range);
    }
    else
    {
        MessageBox.Show("Неправильний формат: " + trimmedElement);
        error = true;
    }
}

```

Рисунок 2.9 – Фрагмент коду, на якому відбувається перевірка форматів та формування діапазонів з них

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		41

На першому етапі розрахунку проводиться зчитування вхідних даних та їх перевірка, а також формування діапазонів на їх основі. Після цього важливим кроком є сортування цих даних від меншої адреси до більшої. Таке сортування необхідне для подальшого об'єднання адрес та діапазонів.

```
result.Sort((a, b) =>
{
    IPAddress ipA = IPAddress.Parse(a.Split('-')[0]);
    IPAddress ipB = IPAddress.Parse(b.Split('-')[0]);
    return CompareIPAddresses(ipA, ipB);
});
```

Рисунок 2.10 – Фрагмент коду, на якому відбувається сортування адрес та діапазонів

Після виконання сортування адрес і діапазонів, наступним кроком є об'єднання цих адрес та діапазонів для уникнення повторень і оптимізації процесу. Це дозволяє зменшити кількість елементів і забезпечити більш ефективну обробку даних.

Після об'єднання адрес та діапазонів формуються мережі на основі кожної утвореної адреси та діапазону. Якщо користувач працює з протоколом IPv4, то ці мережі утворюються з оберненою маскою, тоді як у випадку IPv6 використовується формування мереж з префіксом. Цей процес має на меті утворення мереж, які виключно належать заданим адресам та діапазонам. Це створює передумови для більш точного визначення доступу до ресурсів. При формуванні мереж з використанням оберненої маски (для IPv4) або префікса (для IPv6), застосунок забезпечує, що кожна мережа містить усі адреси та діапазони, які до неї належать. Таким чином, забезпечується відповідність мережевих налаштувань зазначеним вхідним даним, що сприяє більш точному та контрольованому керуванню доступом до ресурсів у мережевому середовищі. Проте, для виробника MikroTik, не потрібно формувати окремі мережі, тому цей процес не проводиться.

Натомість, код, пов'язаний з процесом об'єднання та утворення мереж, є досить об'ємним. Тому, замість наведення фрагментів коду, весь код цих процесів буде представлений в додатку до роботи. Важливими методами,

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		42

використовуваними для цих процесів, є «MergeRanges» (об'єднання діапазонів) та «ConvertRangesToNetworkWithInverseMask» (утворення мереж).

Для обробки вхідних даних для полів порт джерела та порт призначення відбуваються схожі процеси, а саме: зчитування вхідних даних, перевірка формату, сортування та об'єднання портів та діапазонів портів. Метод «ReadPortRanges» відповідає цей процес.

2.3. Особливості формування списків доступу для різних міжмережних екранів

В сучасних комп'ютерних мережах широко застосовуються різноманітні мережні пристрої, такі як маршрутизатори та комутатори, від виробників Cisco, Huawei та MikroTik. Кожен з цих виробників має свої особливості щодо форматів команд Access Control List (ACL) та їх параметрів. Для успішної реалізації програмного засобу для розрахунку оптимальних списків доступу, необхідно провести аналіз підтримуваних форматів команд ACL кожним з цих виробників та визначити їх особливості.

В ході огляду форматів команд ACL для виробника Cisco було виявлено, що вони базуються на розширеному форматі стандартних команд ACL, що надає додаткові можливості для встановлення деталей правил доступу. Виробник Cisco також підтримує розширений набір параметрів, які можна використовувати для точного налаштування правил доступу на їхніх мережних пристроях.

```
Router(config)#access-list 101 permit ip 10.0.0.0 0.255.255.255 host 50.0.0.10
Router(config)#access-list 101 deny ip 10.0.0.0 0.255.255.255 50.0.0.0 0.255.255.255
Router(config)#access-list 101 permit ip 10.0.0.0 0.255.255.255 20.0.0.0 0.255.255.255
Router(config)#access-list 101 deny ip 10.0.0.0 0.255.255.255 any
```

Рисунок 2.11 – Приклад команди ACL на обладнанні Cisco

У разі виробника Huawei було встановлено, що вони використовують свій власний формат команд ACL, який має схожість зі стандартним форматом, але містить деякі відмінності. Формат команд ACL від Huawei також підтримує розширений набір параметрів для більш гнучкого налаштування правил доступу.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		43

```
[Huawei-Router] display acl 3500
```

Basic ACL 3500, 4 rules

ACL's step in 5

```
rule 5 deny source 10.10.5.0/24 destination 172.16.1.0/24
```

```
rule 10 deny tcp source 10.10.6.0/24 destination 172.16.2.10 destination-port eq 21
```

```
rule 15 permit tcp source 10.10.5.0/24 destination 172.16.2.10 destination-port eq 21
```

```
rule 20 permit source 10.10.6.0/24 destination 172.16.1.0/24
```

Рисунок 2.12 – Приклад команди ACL на обладнанні Huawei

У випадку з виробником MikroTik, було виявлено, що їхні команди ACL мають свої особливості та відрізняються від форматів Cisco та Huawei. MikroTik також підтримує розширений набір параметрів для встановлення правил доступу та надає можливості для гнучкого конфігурування списків доступу на їхніх мережних пристроях.

```
/ip firewall filter  
add chain=input action=accept src-address=192.168.1.0/24  
dst-address=0.0.0.0/0 protocol=tcp dst-port=80
```

Рисунок 2.13 – Приклад команди ACL на обладнанні Huawei

Результати аналізу форматів команд ACL виробників Cisco, Huawei та MikroTik дають уявлення про їхні особливості та відмінності. Ця інформація є важливою для подальшого проектування програмного засобу, оскільки вона дозволяє врахувати специфіку кожного виробника та забезпечити коректне формування команд ACL для обраного мережного пристрою.

Таким чином, аналіз форматів команд ACL для виробників Cisco, Huawei та MikroTik є важливим етапом у проектуванні програмного засобу для розрахунку оптимальних списків доступу та дозволяє враховувати особливості кожного виробника для належного конфігурування мережних пристроїв.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		44

При формуванні команд спершу перевіряється обраний виробник. Це важлива перевірка, оскільки кожен виробник має свій формат команд ACL.

```
private void buttonResult_Click(object sender, EventArgs e)
{
    listBoxResult.Items.Clear();
    List<string> result = new List<string>();
    if (radioCisco)
    {
        result = resultCisco();
    }
    else if (radioHuawei)
    {
        result = resultHuawei();
    }
    else
    {
        result = resultMikroTik();
    }
    if (error)
    {
    }
    else listBoxResult.Items.AddRange(result.ToArray());
    error = false;
}
```

Рисунок 2.14 – Фрагмент коду, на якому відбувається перевірка виробника перед виводом результатів

Розглянемо варіант, коли було обрано виробника Cisco. Перед формуванням команди застосунку потрібно отримати вхідні дані. Перевірка поля вводу з номером або ім'ям ACL є необхідною для зчитування, коли вибрано нумеровані списки, і поля введення повинні містити числове значення. Поля правила та протоколу не перевіряється.

```
int number;
string accessList = textAccessListNumber.Text.Trim();
if (radioNumber)
{
    if (int.TryParse(accessList, out number))
    {}
    else
    {
        MessageBox.Show("Неправильний формат: " + accessList);
        error = true;
    }
}
```

Рисунок 2.15 – Фрагмент коду, на якому відбувається отримання вхідних даних, а саме: номер чи ім'я ACL, правило, протокол

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Після цього програма переходить до отримання даних з полів адреси та порту джерела та призначення, і виконуємо необхідні операції, які були описані раніше.

```
string input = textAddressSource.Text.Trim();
List<string> addressSourceResult = new List<string>();
if (input == "any")
{
    addressSourceResult.Clear();
    addressSourceResult.Add("any");
}
else addressSourceResult = ReadSortNetworkRanges(input);

input = textPortSource.Text.Trim();
List<string> portSourceResultCisco = new List<string>();
if (input == "any")
{
    portSourceResultCisco.Clear();
    portSourceResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();
    portResult = ReadPortRanges(input);
    foreach (string element in portResult)
    {
        string[] portRange = element.Split('-');
        if (portRange.Length == 2)
        {
            portSourceResultCisco.Add("range " + portRange[0] + " " + portRange[1]);
        }
        else portSourceResultCisco.Add("eq " + portRange[0]);
    }
}
```

Рисунок 2.16 – Фрагмент коду, на якому відбувається отримання вхідних даних, а саме: адреси та порту джерела (для адреси та порту призначення відбуваються аналогічні дії)

Після отримання всіх вхідних даних, застосунок починає формування команд в залежності від двох факторів: тип списку доступу (нумерований чи іменний) та версія протоколу (IPv4 або IPv6). Нижче наведено приклад коду для нумерованого списку доступу (ACL) для IPv4. Повний код програми буде наведений у додатку.

```

if (radioNumber)
{
    for (int i = 0; i < addressSourceResult.Count; i++)
    {
        for (int j = 0; j < portSourceResultCisco.Count; j++)
        {
            for (int k = 0; k < addressDestinationResult.Count; k++)
            {
                for (int n = 0; n < portDestinationResultCisco.Count; n++)
                {
                    string item = string.Format("access-list {0} {1} {2} {3} {4} {5} {6}", accessList,
                        rule, protocol, addressSourceResult[i], portSourceResultCisco[j],
                        addressDestinationResult[k], portDestinationResultCisco[n]);
                    result.Add(item.Replace(" ", " "));
                }
            }
        }
    }
}
else

```

Рисунок 2.17 – Фрагмент коду, на якому відбувається формування команд для нумерованих ACL для IPv4

2.4. Висновки до розділу 2

У розділі 2 було проведено детальний аналіз та опис основних аспектів проектування програмного засобу для розрахунку оптимальних списків доступу (ACL) у мережевому обладнанні трьох виробників: Cisco, Huawei та MikroTik. Зазначені виробники були обрані як основні для розробки програми.

В рамках алгоритму програми було розглянуто кроки, необхідні для правильного формування команд ACL з урахуванням вхідних даних. Перевірка обраного виробника відіграє ключову роль у подальшому формуванні команд. Зчитування та обробка вхідних даних включає перевірку форматів адрес джерела та призначення, а також портів джерела та призначення. Некоректні формати викликають відповідні повідомлення про помилку. Далі адреси та порти перетворюються у діапазони та сортуються для подальшого об'єднання послідовних або накладених діапазонів.

Наступним кроком є формування відповідних команд ACL на основі отриманих результатів. З урахуванням типу списку доступу (нумерований чи

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		47

іменний) та версії протоколу (IPv4 або IPv6) програма генерує відповідні команди ACL.

Загалом, розділ 2 надає необхідний теоретичний та практичний фундамент для розробки програмного засобу, що забезпечує оптимальний розрахунок списків доступу для мережевого обладнання Cisco, Huawei та MikroTik.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		48

РОЗДІЛ 3. ВСТАНОВЛЕННЯ, НАЛАГОДЖЕННЯ, ТА ТЕСТУВАННЯ РОБОТИ ЗАСТОСУНКУ

3.1. Встановлення та налагодження та видалення застосунку

Встановлення застосунку:

Крок 1: Завантаження завантажувального файлу

1. Користувач має мати доступ до завантажувального файлу setup.exe. Цей файл містить інсталяційний пакет застосунку.

Ім'я	Дата змінення	Тип
Application Files	21.06.2023 18:17	Папка файлів
ACL calculator.application	21.06.2023 18:17	Application Mani
setup.exe	21.06.2023 18:17	Застосунок

Рисунок 3.1 – Папка з завантажувальним файлом

Крок 2: Запуск інсталяційного пакету

1. Знаходячись у відповідній папці, подвійним клацанням на файлі setup.exe потрібно запустити процес інсталяції.

2. Підтвердити встановлення, натиснувши на «Install»

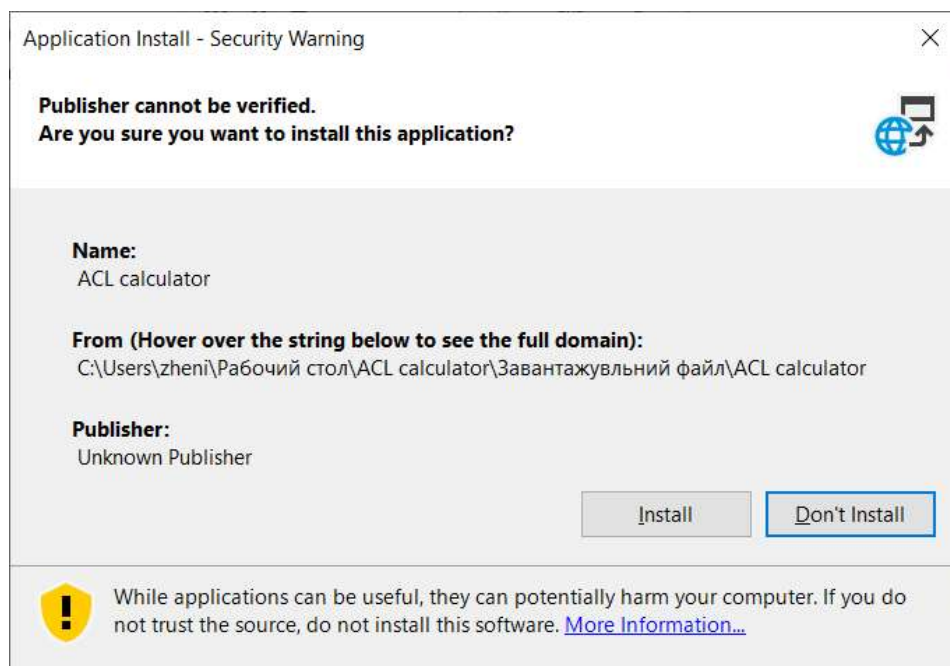


Рисунок 3.2 – Вікно підтвердження завантаження

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				49
Змн.	Арк.	№ докум.	Підпис	Дата		

Крок 3: Очікування завершення інсталяції

1. Для завершення процесу інсталяції треба дочекатися, коли встановлення застосунку буде завершено.

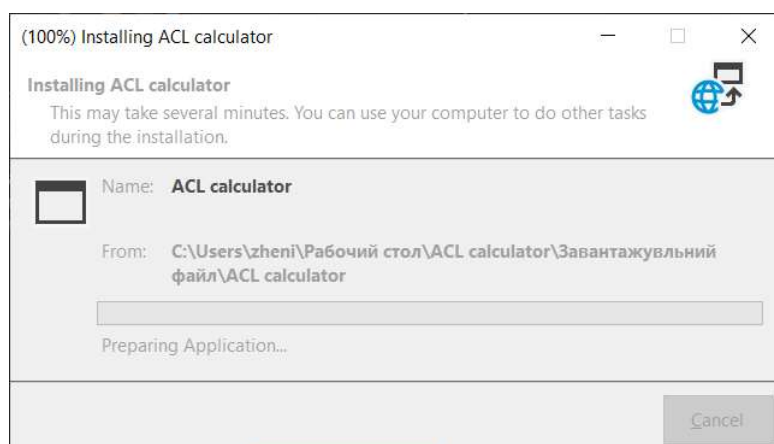


Рисунок 3.3 – Процес встановлення застосунку

Крок 4: Перевірка встановлення

1. Після успішного завершення процесу інсталяції застосунок буде доступний у списку встановлених програм на вашому комп'ютері.

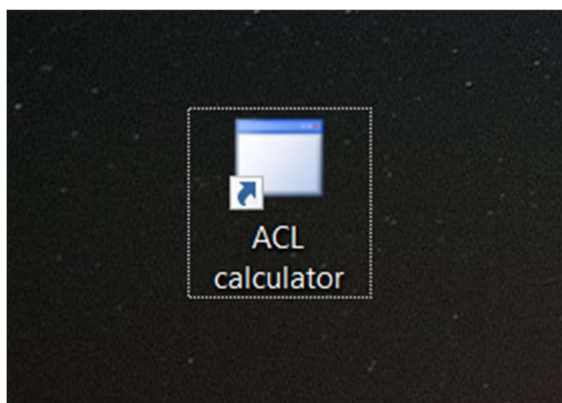


Рисунок 3.4 – Встановлений застосунок на робочому столі

Налагодження застосунку:

Після встановлення застосунку, потрібно виконати наступні кроки для початку роботи з ним:

1. Користувач має запустити застосунок, подвійним клацанням на його ярлику або шляхом пошуку та вибору його в меню Пуск.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		50

2. Після запуску застосунку відкриється його головний екран, який має стандартні налаштування. У цьому випадку, виробник встановлений як Cisco, версія IP – IPv4, а тип списків доступу – нумеровані.

3. Щоб змінити налаштування застосунку, користувач має натиснути на відповідну кнопку «Налаштування». Це відкриє вікно налаштувань.

4. У вікні налаштувань можна змінити виробника, версію IP та тип списків доступу. Наприклад, можна обрати іншого виробника, такий як Huawei або MicroTik, змінити версію IP на Ipv6 або обрати іменні списки доступу замість нумерованих.

5. Після внесення змін у налаштування потрібно зберегти їх натиснувши відповідну кнопку «Зберегти». Якщо закрити вікно, то внесені зміни в налаштуваннях не зберуться.

6. Тепер користувач зможе працювати з застосунком, використовуючи нові налаштування, що було встановлено. Потрібно звернути увагу, що головний екран застосунку тепер відображатиме вибрані параметри, такі як виробник, версія IP та тип списків доступу.

За допомогою цих налаштувань можна персоналізувати роботу застосунку відповідно до потреб та вимог користувач.

ACL calculator

Налаштування

Виробник: Cisco
Версія IP-протоколу: IPv4

Номер списку доступу: 100 Правило: permit

Трафік: Протокол: ip

Адреса джерела, діапазон, список: апу Завантажити

Порт джерела, діапазон, список: апу Завантажити

Адреса призначення, діапазон, список: апу Завантажити

Порт призначення, діапазон, список: апу Завантажити

Обчислити **Зберегти ACL у файл**

Рисунок 3.5 – Запущений застосунок

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		51

Налаштування

Виробник

☐ Cisco ☐ Huawei ☒ MikroTik

Версія IP

☐ IPv4 ☒ IPv6

Типи списків доступу

☐ Нумеровані ☐ Іменовані

Зберегти

Рисунок 3.6 – Вікно налаштувань

ACL calculator

Налаштування **Виробник: MikroTik**
Версія IP-протоколу: IPv6

Номер списку доступу: --- Правило: **accept** ▾

Трафік: **input** ▾ Протокол: **ip** ▾

Адреса джерела, діапазон, список: any Завантажити

Порт джерела, діапазон, список: any Завантажити

Адреса призначення, діапазон, список: any Завантажити

Порт призначення, діапазон, список: any Завантажити

Обчислити Зберегти ACL у файл

Рисунок 3.7 – Головне вікно після збереження налаштувань

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Видалення застосунку:

Щоб видалити застосунок «ACL calculator» потрібно виконати наступні кроки:

1. Відкрити налаштування комп'ютера або пристрою, на якому встановлений застосунок «ACL calculator».
2. Знайти розділ або опцію, пов'язану з керуванням програмами або програмними компонентами. Зазвичай це можна знайти у розділі «Система», «Програми та компоненти» або «Додаткові налаштування».
3. У списку програм та компонентів знайти застосунок «ACL calculator» і клацнути на ньому.
4. Обрати опцію «Видалити» або «Деінсталювати», яка з'явиться після клацання на застосунок.
5. Після вибору опції видалення треба слідувати інструкціям на екрані для підтвердження процесу видалення.
6. Почекаати, доки процес видалення завершиться. Це може зайняти деякий час, в залежності від швидкості комп'ютера.

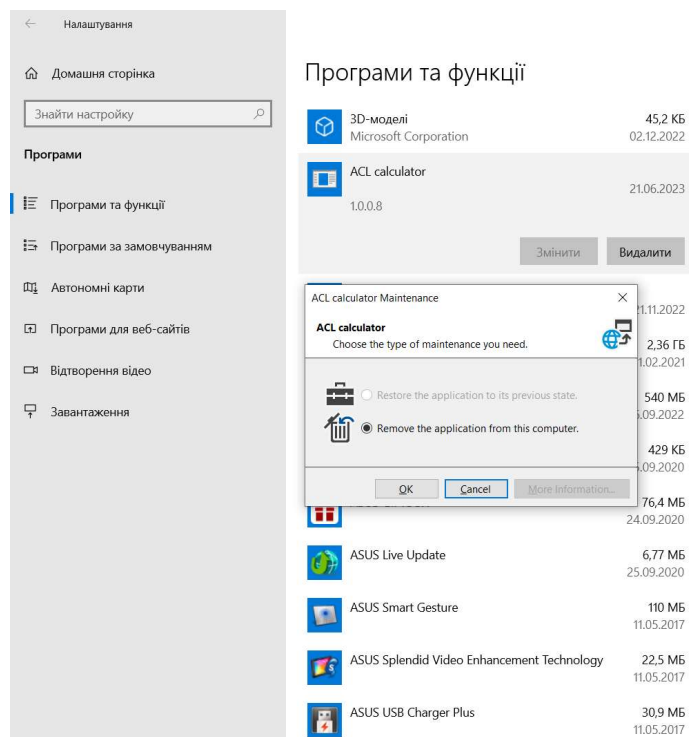


Рисунок 3.8 – Видалення застосунку

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		53

3.2. Тестування роботи застосунку для формування простих списків доступу

Простий список доступу – це метод контролю доступу, який використовується для обмеження або дозволу на певні види мережевого трафіку. Він включає в себе набір правил, які визначають, які джерела, призначення та сервіси мають бути дозволені або заборонені. Простий список доступу може містити окремі IP-адреси, діапазони адрес, порти та інші параметри, що використовуються для ідентифікації та керування мережевим трафіком.

Простий список доступу може мати наступну структуру:

1. Дозволити доступ з джерела А до призначення В на порту Х.
2. Заборонити доступ з джерела С до призначення D на порту Y.

Цей тип списку доступу використовується для простих сценаріїв контролю доступу, де правила є прямими та дозволяють або забороняють специфічні комбінації джерел, призначень та портів.

Хоча простий список доступу може бути менш гнучким, він все ще є ефективним інструментом для базового контролю доступу в мережах. Він дозволяє встановити основні правила для фільтрації мережевого трафіку та забезпечує базовий рівень безпеки та керування ресурсами.

Спробуємо протестувати застосунок за декількома сценаріями:

1. Дозволити доступ з будь-якого джерела до сервера на порту 80.
2. Заборонити доступ з джерела 192.168.1.10 до будь-якого призначення.
3. Дозволити доступ з джерела 2001:0db8:85a3::8a2e:0370:7334 до будь-якого призначення на портах 22 і 443.
4. Заборонити доступ з джерела 172.16.0.0/24 до призначення 192.168.0.0/16.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		54

ACL calculator

Виробник: Cisco
Версія IP-протоколу: IPv4

Налаштування

Номер списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

```
access-list 100 permit ip any any eq 80
```

Рисунок 3.9 – Сценарій 1

ACL calculator

Виробник: MikroTik
Версія IP-протоколу: IPv4

Налаштування

Номер списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

```
/ip firewall filter add chain=input action=drop protocol=ip src-address=192.168.1.10
```

Рисунок 3.10 – Сценарій 2

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		55

ACL calculator

Виробник: Huawei
Версія IP-протоколу: IPv6

Налаштування

Ім'я списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

```

ipv6 acl name My_ACL
rule 1 permit ip source-address 2001:0db8:85a3:0:0:8a2e:0370:7334 destination-port eq 22
rule 2 permit ip source-address 2001:0db8:85a3:0:0:8a2e:0370:7334 destination-port eq 443
return

```

Рисунок 3.11 – Сценарій 3

ACL calculator

Виробник: Cisco
Версія IP-протоколу: IPv4

Налаштування

Ім'я списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

```

ip access-list extended MyACL1
permit ip 172.16.0.0 0.0.0.255 192.168.0.0 0.0.255.255
exit

```

Рисунок 3.12 – Сценарій 4

3.3. Тестування роботи застосунку для оптимальних списків доступу

Оптимальний список доступу відрізняється від простого списку доступу шляхом використання більш розширених та деталізованих правил для контролю доступу до мережних ресурсів. Він забезпечує більш гнучкий та точний контроль над тим, які користувачі або пристрої мають дозвіл на доступ до певних ресурсів та які дії вони можуть виконувати.

Оптимальний список доступу включає в себе більш широкий спектр параметрів для керування доступом, такі як адреси джерела та призначення, порти, протоколи, типи пакетів тощо. Він дозволяє встановлювати більш складні правила, враховуючи різні атрибути трафіку та потреби мережі. Це дозволяє забезпечити більш точне визначення доступу та зменшити ризик небажаного або недозволеного доступу до мережних ресурсів.

Оптимальний список доступу дає можливість адміністратору мережі гнучко налаштовувати політику безпеки та контролю доступу до мережних ресурсів, забезпечуючи ефективну захист від загроз і небажаної активності.

Спробуємо протестувати застосунок за сценарієм:

Ім'я списку доступу: MyFirewall

Правило: дозволити

Протокол: tcp

Адреса джерела:

- 1) 192.168.1.10
- 2) 10.0.0.1-10.0.0.100
- 3) 172.16.0.50-60
- 4) 192.168.2.0/24

Порт джерела:

- 1) 50-60
- 2) 443

Адреса призначення: 10.10.0.0/16

Порт призначення: 204

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		57

Тестування:

1. Ми запишемо адреси джерела у тестовий файл з метою перевірки функції завантаження, а потім завантажимо ці адреси у поле для адрес джерела.

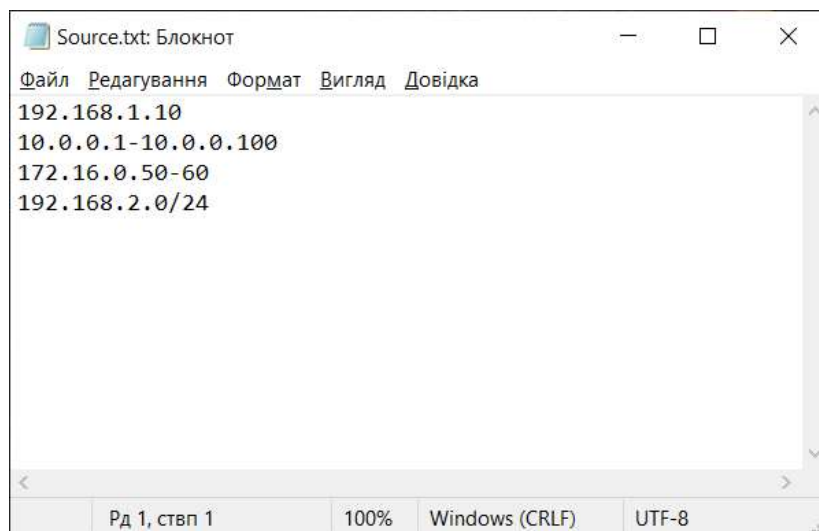


Рисунок 3.13 – Запис адрес джерела у файл

ACL calculator

Виробник: Cisco
Версія IP-протоколу: IPv4

Налаштування

Ім'я списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

Рисунок 3.14 – Успішне завантаження

2. Заповнюємо інші поля вхідних даних і натискаємо кнопку «Обчислити»

ACL calculator

Виробник: Cisco
Версія IP-протоколу: IPv4

Налаштування

Ім'я списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

```
ip access-list extended MyFirewall
permit tcp host 10.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp host 10.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.8 0.0.0.7 range 50 60 10.10.0.0 0.0.255.255 eq 204
```

Рисунок 3.15 – Результат обчислення

3. Зберігаємо файл на робочому столі та дивимося на результат.

ACL calculator

Виробник: Cisco
Версія IP-протоколу: IPv4

Налаштування

Ім'я списку доступу: Правило:

Трафік: Протокол:

Адреса джерела, діапазон, список:

Порт джерела, діапазон, список:

Адреса призначення, діапазон, список:

Порт призначення, діапазон, список:

Файл успішно збережено.

```
ip access-list extended MyFirewall
permit tcp host 10.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp host 10.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.8 0.0.0.7 range 50 60 10.10.0.0 0.0.255.255 eq 204
```

Рисунок 3.16 – Збереження результатів

```

ip access-list extended MyFirewall
permit tcp host 10.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp host 10.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.2 0.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.4 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.8 0.0.0.7 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.8 0.0.0.7 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.16 0.0.0.15 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.16 0.0.0.15 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.32 0.0.0.31 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.32 0.0.0.31 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.64 0.0.0.31 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.64 0.0.0.31 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.96 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp 10.0.0.96 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp host 10.0.0.100 range 50 60 10.10.0.0 0.0.255.255 eq 204
permit tcp host 10.0.0.100 eq 443 10.10.0.0 0.0.255.255 eq 204
permit tcp 172.16.0.50 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204

```

Рисунок 3.17 – Збережений файл

4. Перевіряємо результати на обладнанні Cisco. Для цього копіюємо вміст файлу та вставляємо (Paste) в командному рядку обладнання

```

R_191_11_1(config)#
R_191_11_1(config)#ip access-list extended MyFirewall
R_191_11_1(config-ext-nacl)#permit tcp host 10.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 10.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.2 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.2 0.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.4 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.4 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.8 0.0.0.7 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.8 0.0.0.7 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.16 0.0.0.15 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.16 0.0.0.15 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.32 0.0.0.31 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.32 0.0.0.31 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.64 0.0.0.31 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.64 0.0.0.31 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.96 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 10.0.0.96 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 10.0.0.100 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 10.0.0.100 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.50 0.0.0.1 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.50 0.0.0.1 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.52 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.52 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.56 0.0.0.3 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 172.16.0.56 0.0.0.3 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 172.16.0.60 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 172.16.0.60 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 192.168.1.10 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp host 192.168.1.10 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 192.168.2.0 0.0.0.255 range 50 60 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#permit tcp 192.168.2.0 0.0.0.255 eq 443 10.10.0.0 0.0.255.255 eq 204
R_191_11_1(config-ext-nacl)#exit
R_191_11_1(config)#

```

Рисунок 3.18 – Команди працюють без помилок

3.4. Висновки до розділу 3

У розділі 3 було проведено встановлення, налагодження та тестування роботи нашого застосунку для формування списків доступу.

У підрозділі 3.1 була надана інструкція зі встановлення застосунку, що базується на Microsoft Visual Studio, C# та Windows Forms. Користувачу було пояснено процес запуску застосунку та основні налаштування, які доступні на головному екрані.

У підрозділі 3.2 було проведено тестування роботи застосунку для формування простих списків доступу. Було наведено приклади простих списків доступу із заданими адресами, діапазонами адрес та мережами. Застосунок успішно виконував розрахунок команд ACL на основі цих даних.

У підрозділі 3.3 було проведено тестування роботи застосунку для оптимальних списків доступу. Був наданий приклад оптимального списку доступу, який містить чотири пунктів з адресами, діапазонами адрес та мережами. Застосунок ефективно обробляв цей список та формував відповідні команди ACL.

Загальною висновком розділу 3 є те, що застосунок для формування списків доступу успішно пройшов встановлення та налагодження, а також продемонстрував свою працездатність під час тестування як для простих, так і для оптимальних списків доступу. Це підтверджує його відповідність поставленим завданням та функціональності.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		61

ВИСНОВКИ

У даній кваліфікаційній було проведено аналіз побудови та застосування списків доступу для міжмережних екранів. В розділі 1 було визначено та класифіковано списки доступу, описано алгоритм роботи та правила їх формування. Також було розглянуто існуючі засоби автоматизованого розрахунку та формування списків доступу.

У розділі 2 було розроблено програмний застосунок для розрахунку оптимальних списків доступу для міжмережних екранів. Була розроблена загальна структура застосунку та модулі для розрахунку та формування списків доступу. Особливу увагу було приділено формуванню списків доступу для різних типів міжмережних екранів.

У розділі 3 було проведено встановлення, налагодження та тестування роботи застосунку. Застосунок був реалізований у середовищі Microsoft Visual Studio 2019, і було проведено тестування для перевірки правильності розрахунку та ефективності формування списків доступу.

Аналіз результатів роботи програмного застосунку показав, що він успішно виконує розрахунок та формування оптимальних списків доступу для міжмережних екранів. Застосунок значно спрощує процес формування списків доступу, забезпечуючи ефективність та безпеку мережевої інфраструктури.

На основі результатів роботи можна зробити висновок, що розроблений програмний застосунок є корисним інструментом для адміністраторів мереж, що працюють з міжмережними екранами. Він дозволяє автоматизувати процес розрахунку та формування списків доступу, що приводить до підвищення ефективності та забезпечення більш високого рівня безпеки.

Програмний застосунок може бути рекомендований для використання в реальних умовах, а також для подальшого розвитку і вдосконалення. Потенційні напрямки розвитку включають розширення функціональності застосунку, підтримку більш широкого спектру міжмережних екранів та оптимізацію алгоритмів розрахунку для роботи з більшими мережевими об'єктами.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Загальною висновком кваліфікаційної роботи є те, що розроблений програмний застосунок успішно вирішує поставлені завдання, спрощуючи процес розрахунку та формування списків доступу для міжмережних екранів. Його використання може значно полегшити роботу адміністраторів мереж та покращити безпеку мережевої інфраструктури.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		63

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Cisco Networking Academy CCNAv7: Enterprise Networking, Security, and Automation (v7.0) [Електронний ресурс] – Режим доступу: www.netacad.com.
2. Cisco Networking Academy CCNAv7: Switching, Routing, and Wireless Essentials (v7.0) [Електронний ресурс] – Режим доступу: www.netacad.com.
3. Cisco Networking Academy CCNAv7: Introduction to Networks [Електронний ресурс] – Режим доступу: www.netacad.com.
4. Cisco Networking Academy. CyberOps Associate (v1.0) [Електронний ресурс] – Режим доступу: www.netacad.com.
5. Configure and Filter IP Access Lists [Електронний ресурс] – Режим доступу до ресурсу: <https://www.cisco.com/c/en/us/support/docs/security/ios-firewall/23602-confaccesslists.html>
6. What Is an Access Control List (ACL) [Електронний ресурс] – Режим доступу до ресурсу: <https://support.huawei.com/enterprise/en/doc/EDOC1100086647>
7. MikroTik. Firewall and Quality of Service [Електронний ресурс] – Режим доступу до ресурсу: <https://help.mikrotik.com/docs/display/ROS/Firewall+and+Quality+of+Service>
8. Conlan J., Patrik. Cisco Network Professional. Advanced Internetworking Guide. / Patrik J. Conlan. – Wiley Publishing, 2009. – 854 p.
9. Paquet, Catherine. Implementing Cisco IOS Network Security (IINS) / Catherine Paquet. – Cisco Press, 2009. – 600 p.
10. Грайворонський М.В., Новіков О.М. Безпека інформаційнокомунікаційних систем. – К.: Видавнича група BVH, 2009. – 608 с.
11. Буров Є.В. Комп'ютерні мережі. Підручник. Том 1. / Є.В. Буров, М.М. Митник. – Львів: «Магнолія 2006», 2021. – 334 с.
12. Буров Є.В. Комп'ютерні мережі. Підручник. Том 2. / Є.В. Буров, М.М. Митник. – Львів: «Магнолія 2006», 2021. – 204 с.
13. Odom Wendell. CCNA 200-301 Official Cert Guide. Volume 1. / Wendell Odom. Cisco Press, 2020. – 1095 p

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		64

14. Odom Wendell. CCNA 200-301 Official Cert Guide. Volume 2. / Wendell Odom. Cisco Press, 2020. – 774 p.

15. Основи побудови локальних комп'ютерних мереж Ethernet на базі керованих комутаторів компанії Cisco: навчальний посібник. [Текст] / А.А. Єфіменко. – Житомир : Державний університет «Житомирська політехніка», 2021. – 116 с.

16. Todd Lamle. CCNA Routing and Switching Complete Study Guide, 2016

17. Тарнавський Ю. А., Кузьменко І. М. Організація комп'ютерних мереж / Ю. А. Тарнавський, І. М. Кузьменко. – Київ : КІІ ім. Ігоря Сікорського, 2018. – 259 с.

18. Barker, Keith. CCNA Security 640-554. Official Cert Guide / Keith Barker, Scott Morris. – Cisco Press, 2013. – 740 p.

19. Santos, Omar. CCNA Security 210-260. Official Cert Guide / Omar Santos, John Stuppi. – Cisco Press, 2015. – 658 p.

20. Єфіменко А.А. Захист інформації в інформаційно-комунікаційних системах: методичні рекомендації для виконання лабораторних робіт. Ч. 2 / підг. А. А. Єфіменко. – Житомир: ЖВІ, 2017. – 176 с.

21. Єфіменко А.А. Захист інформації в інформаційно-комунікаційних системах: методичні рекомендації для виконання лабораторних робіт. Ч. 1 / підг. А. А. Єфіменко. – Житомир: ЖДТУ, 2018. – 112 с.

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		65

ДОДАТКИ

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		66

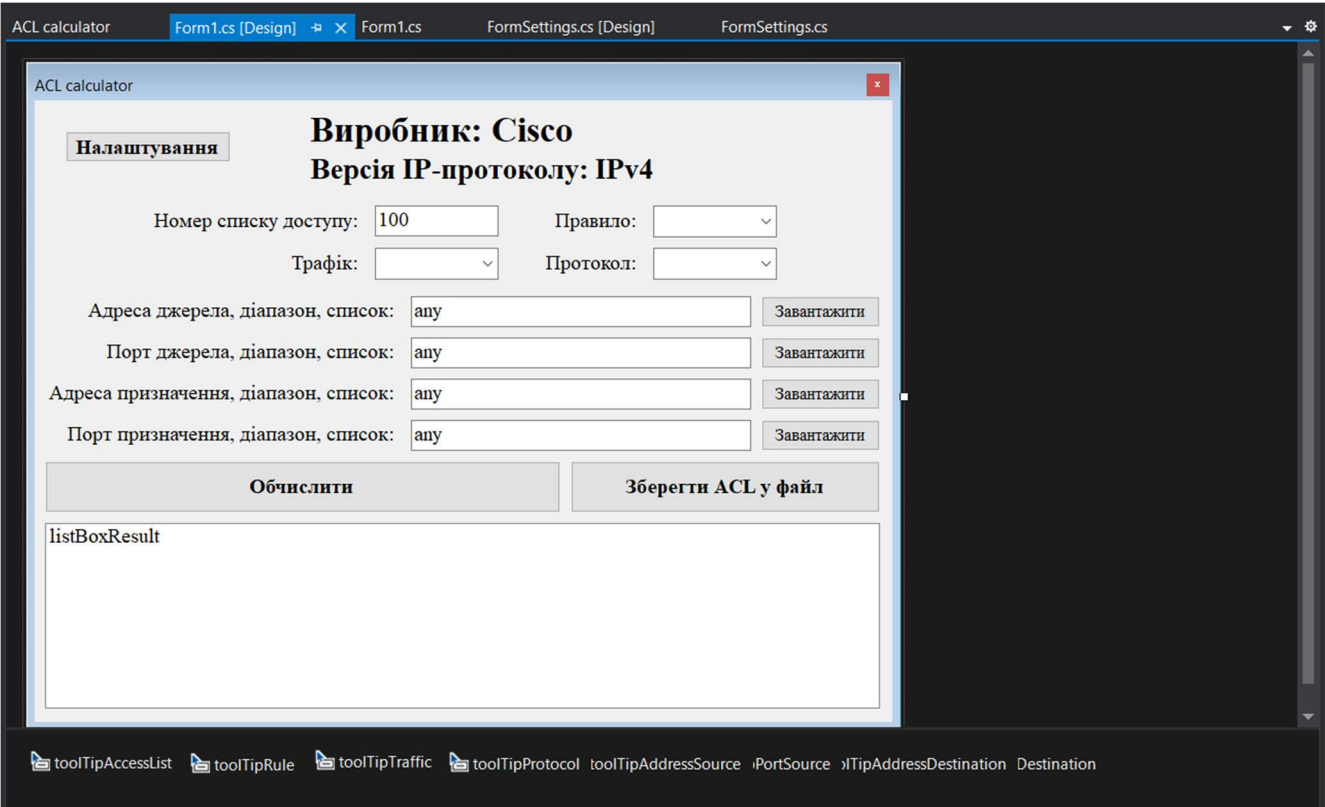


Рисунок А.1 – Проєкт головного вікна

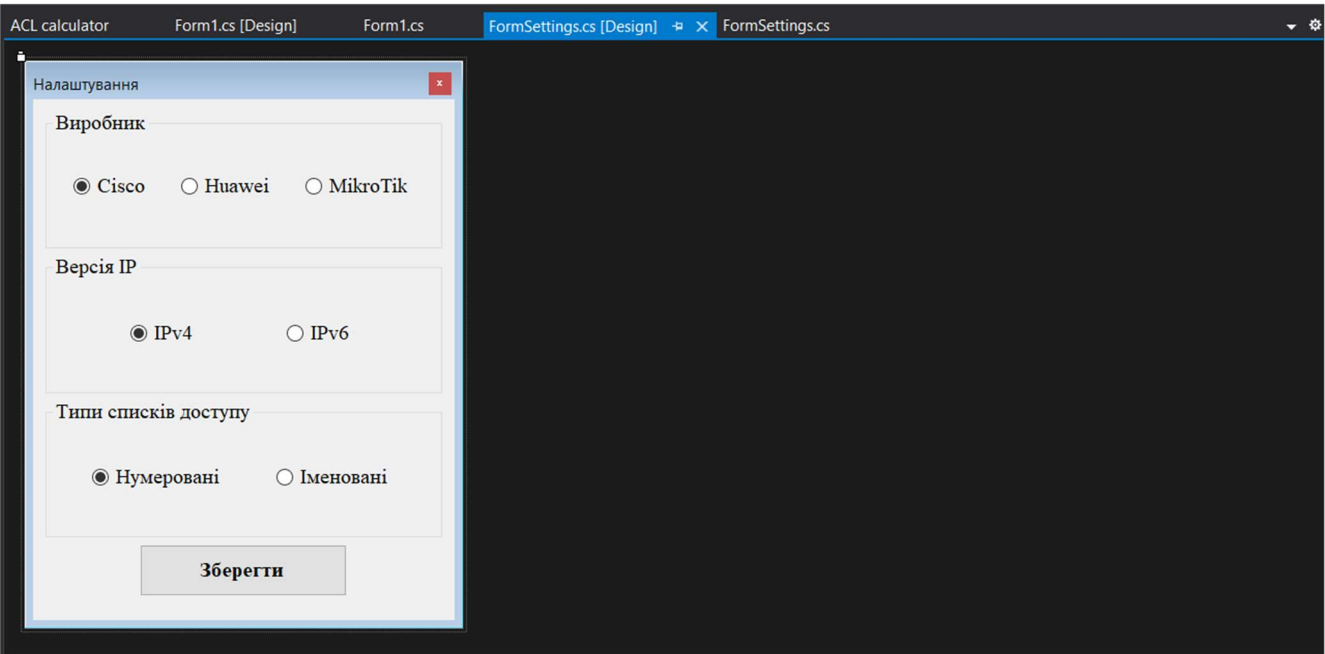


Рисунок А.2 – Проєкт вікна налаштування

Код для головного вікна програми:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Net.Sockets;
using System.Text.RegularExpressions;
namespace ACL_calculator
{
    public partial class Main_Form : Form
    {
        string rule = "permit";
        string protocol = "ip";
        bool radioCisco = true;
        bool radioHuawei;
        bool radioIPv4 = true;
        bool radioNumber = true;
        bool error = false;
        public Main_Form()
        {
            InitializeComponent();
        }
        private void buttonResult_Click(object sender, EventArgs e)
        {
            listBoxResult.Items.Clear();
            List<string> result = new List<string>();
            if (radioCisco)
            {
                result = resultCisco();
            }
            else if (radioHuawei)
            {
                result = resultHuawei();
            }
            else
            {
                result = resultMikroTik();
            }
            if (error)
            {
            }
            else listBoxResult.Items.AddRange(result.ToArray());
            error = false;
        }
        private List<string> ReadPortRanges(string input)
        {
            List<string> result = new List<string>();
            string[] elements = input.Split(new[] { "," }, StringSplitOptions.RemoveEmptyEntries);
            foreach (string element in elements)
            {
                string trimmedElement = element.Trim();
                int numberStart;
                int numberEnd;
                if (int.TryParse(trimmedElement, out numberStart))
                {
                    if (numberStart >= 0 && numberStart <= 65535) result.Add(trimmedElement);
                }
            }
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М				
Змн.	Арк.	№ докум.	Підпис	Дата		68

```

        else
        {
            error = true;
            MessageBox.Show("Неправильний формат: " + trimmedElement);
        }
    }
    else
    {
        string[] portRange = trimmedElement.Split('-');
        if (portRange.Length == 2)
        {
            if ((int.TryParse(portRange[0], out numberStart) && (int.TryParse(portRange[1], out numberEnd))))
            {
                if (numberEnd > numberStart)
                {
                    if (numberStart >= 0 && numberStart <= 65535 && numberEnd >= 0 && numberEnd <= 65535)
                    result.Add(trimmedElement);
                }
                else
                {
                    MessageBox.Show("Неправильний формат: " + trimmedElement);
                    error = true;
                }
            }
            else
            {
                MessageBox.Show("Неправильний формат: " + trimmedElement);
                error = true;
            }
        }
        else
        {
            MessageBox.Show("Неправильний формат: " + trimmedElement);
            error = true;
        }
    }
    else
    {
        MessageBox.Show("Неправильний формат: " + trimmedElement);
        error = true;
    }
}
}
}
result.Sort((a, b) =>
{
    int portA = int.Parse(a.Split('-')[0]);
    int portB = int.Parse(b.Split('-')[0]);
    return ComparePorts(portA, portB);
});
List<string> mergedResult = MergePorts(result);
return mergedResult;
}
private int ComparePorts(int portA, int portB)
{
    if (portA > portB) return 1;
    else if (portA < portB) return -1;
    else return 0;
}
private List<string> ReadSortNetworkRanges(string input)
{
    List<string> result = new List<string>();
    string[] elements = input.Split(new[] { "," }, StringSplitOptions.RemoveEmptyEntries);
    foreach (string element in elements)

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росіньський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

```

{
    string trimmedElement = element.Trim();
    if (IsValidIPAddress(trimmedElement))
    {
        result.Add(ExpandIPv6Address(trimmedElement));
    }
    else if (IsValidAddressRange(trimmedElement))
    {
        string range = FormatAddressRange(trimmedElement);
        result.Add(range);
    }
    else if (IsValidNetworkWithMask(trimmedElement))
    {
        string range = ConvertNetworkWithMaskToRange(trimmedElement);
        result.Add(range);
    }
    else if (IsValidNetworkWithPrefix(trimmedElement))
    {
        string range = ConvertNetworkWithPrefixToRange(trimmedElement);
        result.Add(range);
    }
    else
    {
        MessageBox.Show("Неправильний формат: " + trimmedElement);
        error = true;
    }
}
result.Sort((a, b) =>
{
    IPAddress ipA = IPAddress.Parse(a.Split('-')[0]);
    IPAddress ipB = IPAddress.Parse(b.Split('-')[0]);
    return CompareIPAddresses(ipA, ipB);
});
List<string> mergedResult = MergeRanges(result);
if (radioCisco || radioHuawei)
{
    List<string> addressSourceResult = ConvertRangesToNetworkWithInverseMask(mergedResult);
    return addressSourceResult;
}
else return mergedResult;
}
private bool IsValidIPAddress(string address)
{
    IPAddress ipAddress;
    if (IPAddress.TryParse(address, out ipAddress))
    {
        if (((ipAddress.AddressFamily == AddressFamily.InterNetwork) && radioIPv4) || (ipAddress.AddressFamily == AddressFamily.InterNetworkV6 && radioIPv6 == false))
        {
            return true;
        }
    }
    return false;
}
private bool IsValidAddressRange(string address)
{
    string[] parts = address.Split('-');
    if (parts.Length == 2)
    {
        string startIP = ExpandIPv6Address(parts[0].Trim());
        string endIP = ExpandIPv6Address(parts[1].Trim());
        if (startIP.Split('.').Count() == 4 || startIP.Split(':').Count() == 8)

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росіньський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

```

{
    IPAddress ipStart = IPAddress.Parse(startIP);
    int range, lastIpStart;
    if (ipStart.AddressFamily == AddressFamily.InterNetwork && radioIPv4)
    {
        if (endIP.Split('.').Count() == 4)
        {
            IPAddress ipEnd = IPAddress.Parse(endIP);
            return IsValidIPAddress(startIP) && IsValidIPAddress(endIP) && CompareIPAddresses(ipEnd, ipStart)
        }
        else if ((int.TryParse(parts[1], out range)) && int.Parse(parts[0].Split('.')[3]) < int.Parse(parts[1]) &&
int.Parse(parts[1]) < 256) return IsValidIPAddress(startIP);
        else if (ipStart.AddressFamily == AddressFamily.InterNetworkV6 && radioIPv4 == false)
        {
            if (endIP.Contains(':'))
            {
                IPAddress ipEnd = IPAddress.Parse(endIP);
                return IsValidIPAddress(startIP) && IsValidIPAddress(endIP) && CompareIPAddresses(ipEnd, ipStart)
            }
            else
            {
                bool isHex = int.TryParse(endIP, System.Globalization.NumberStyles.HexNumber, null, out range);
                isHex = int.TryParse(startIP.Split(':')[7], System.Globalization.NumberStyles.HexNumber, null, out
lastIpStart);
                if (int.TryParse(endIP, System.Globalization.NumberStyles.HexNumber, null, out range) &&
endIP.Length <= 4 && range > lastIpStart) return IsValidIPAddress(startIP);
            }
        }
    }
    return false;
}

private string FormatAddressRange(string address)
{
    string[] parts = address.Split('-');
    string startIP = ExpandIPv6Address(parts[0].Trim());
    string endIP = ExpandIPv6Address(parts[1].Trim());
    if (endIP.Contains(".") || endIP.Contains(":"))
    {
        return ExpandIPv6Address(address.Replace(" ", "").Split('-')[0]) + "-" + ExpandIPv6Address(address.Replace(" ",
""), Split('-')[1]);
    }
    else
    {
        if (startIP.Contains('.'))
        {
            string[] startOctets = startIP.Split('.');
            string endOctets = endIP;
            string formattedEndIP = startOctets[0] + "." + startOctets[1] + "." + startOctets[2] + "." + endOctets;
            return startIP + "-" + formattedEndIP;
        }
        else
        {
            string[] startOctets = startIP.Split(':');
            string endOctets = endIP;
            string formattedEndIP = startOctets[0] + ":" + startOctets[1] + ":" + startOctets[2] + ":" + startOctets[3] + ":" +
startOctets[4] + ":" + startOctets[5] + ":" + startOctets[6] + ":" + endOctets;
            return startIP + "-" + formattedEndIP;
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росіньський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

```

    }
}
private bool IsValidNetworkWithMask(string address)
{
    if (radioIPv4)
    {
        string[] parts = address.Split(new[] { " " }, StringSplitOptions.RemoveEmptyEntries);
        if (parts.Length == 2)
        {
            bool maskTrue = IsValidSubnetMask(parts[1]);
            if (maskTrue) return IsValidIPAddress(parts[0]) && IsValidIPAddress(parts[1]);
        }
        return false;
    }
    return false;
}
private bool IsValidSubnetMask(string subnetMask)
{
    IPAddress maskIP;
    if (!IPAddress.TryParse(subnetMask, out maskIP))
    {
        return false;
    }
    byte[] maskBytes = maskIP.GetAddressBytes();
    string binaryMask = string.Join("", maskBytes.Select(b => Convert.ToString(b, 2).PadLeft(8, '0')));
    bool hasInvalidSequence = false;
    bool foundZeroAfterOne = false;
    foreach (char bit in binaryMask)
    {
        if (bit == '1' && foundZeroAfterOne)
        {
            hasInvalidSequence = true;
            break;
        }
        else if (bit == '0')
        {
            foundZeroAfterOne = true;
        }
    }
    int prefixLength = binaryMask.IndexOf('0');
    int significantBitsCount = binaryMask.Count(bit => bit == '1');
    return !hasInvalidSequence && prefixLength == significantBitsCount;
}
private bool IsValidNetworkWithPrefix(string address)
{
    string[] parts = address.Split('/');
    if (parts.Length == 2)
    {
        int prefix;
        parts[1].Trim();
        if (radioIPv4)
        {
            if (int.TryParse(parts[1], out prefix))
            {
                if ((int.Parse(parts[1]) <= 32) && (int.Parse(parts[1]) >= 0)) return IsValidIPAddress(parts[0]) &&
                parts[1].All(char.IsDigit);
            }
        }
        else
        {
            if (int.TryParse(parts[1], out prefix))

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

```

        {
            if ((int.Parse(parts[1]) <= 128) && (int.Parse(parts[1]) >= 0)) return IsValidIPAddress(parts[0]) &&
parts[1].All(char.IsDigit);
        }
    }
}
return false;
}
private string ConvertNetworkWithMaskToRange(string address)
{
    string[] parts = address.Split(new[] { " " }, StringSplitOptions.RemoveEmptyEntries);
    string networkAddress = parts[0];
    string subnetMask = parts[1];
    string endAddress = CalculateEndAddress(networkAddress, subnetMask);
    string startAddress = CalculateStartAddress(endAddress, subnetMask);
    return $" {startAddress} - {endAddress} ";
}
private string ConvertNetworkWithPrefixToRange(string address)
{
    string[] parts = address.Split("/");
    string networkAddress = parts[0];
    string prefix = parts[1];
    string subnetMask = GetSubnetMaskFromPrefix(prefix);
    string endAddress = CalculateEndAddress(networkAddress, subnetMask);
    string startAddress = CalculateStartAddress(endAddress, subnetMask);
    if (startAddress == endAddress) return $" {ExpandIPv6Address(startAddress)} ";
    return $" {ExpandIPv6Address(startAddress)} - {ExpandIPv6Address(endAddress)} ";
}
private string CalculateStartAddress(string endAddress, string subnetMask)
{
    IPAddress endIP = IPAddress.Parse(endAddress);
    IPAddress maskIP = IPAddress.Parse(subnetMask);
    byte[] endBytes = endIP.GetAddressBytes();
    byte[] maskBytes = maskIP.GetAddressBytes();
    byte[] startBytes = new byte[maskBytes.Count()];
    for (int i = 0; i < maskBytes.Count(); i++)
    {
        startBytes[i] = (byte)(endBytes[i] ^ (byte)~maskBytes[i]);
    }
    return new IPAddress(startBytes).ToString();
}
private string CalculateEndAddress(string networkAddress, string subnetMask)
{
    IPAddress networkIP = IPAddress.Parse(networkAddress);
    IPAddress maskIP = IPAddress.Parse(subnetMask);
    byte[] networkBytes = networkIP.GetAddressBytes();
    byte[] maskBytes = maskIP.GetAddressBytes();
    byte[] endBytes = new byte[maskBytes.Count()];
    for (int i = 0; i < maskBytes.Count(); i++)
    {
        endBytes[i] = (byte)(networkBytes[i] | (byte)~maskBytes[i]);
    }
    return new IPAddress(endBytes).ToString();
}
private string GetSubnetMaskFromPrefix(string prefix)
{
    int prefixLength = int.Parse(prefix);
    byte[] maskBytes;
    if (radioIPv4)
    {
        uint mask = prefixLength == 0 ? 0 : ~(1u << (32 - prefixLength)) - 1;
        maskBytes = BitConverter.GetBytes(mask);
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

```

    }
    else
    {
        int fullBytes = prefixLength / 8;
        int remainingBits = prefixLength % 8;
        maskBytes = new byte[16];
        for (int i = 0; i < fullBytes; i++)
        {
            maskBytes[i] = 255;
        }
        if (remainingBits > 0)
        {
            maskBytes[fullBytes] = (byte)(255 << (8 - remainingBits));
        }
        Array.Reverse(maskBytes);
    }
    if (BitConverter.IsLittleEndian)
    {
        Array.Reverse(maskBytes);
    }
    return new IPAddress(maskBytes).ToString();
}
private int CompareIPAddresses(IPAddress ipA, IPAddress ipB)
{
    byte[] bytesA = ipA.GetAddressBytes();
    byte[] bytesB = ipB.GetAddressBytes();
    if (ipA.AddressFamily == AddressFamily.InterNetwork && ipB.AddressFamily == AddressFamily.InterNetwork)
    {
        for (int i = 0; i < 4; i++)
        {
            if (bytesA[i] < bytesB[i])
            {
                return -1;
            }
            else if (bytesA[i] > bytesB[i])
            {
                return 1;
            }
        }
    }
    else if (ipA.AddressFamily == AddressFamily.InterNetworkV6 && ipB.AddressFamily == AddressFamily.InterNetworkV6)
    {
        for (int i = 0; i < 16; i++)
        {
            if (bytesA[i] < bytesB[i])
            {
                return -1;
            }
            else if (bytesA[i] > bytesB[i])
            {
                return 1;
            }
        }
    }
    return 0;
}
private List<string> MergePorts(List<string> ranges)
{
    List<string> mergedResult = new List<string>();
    for (int i = 0; i < ranges.Count; i++)
    {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

```

if (i == ranges.Count - 1)
{
    mergedResult.Add(ranges[i]);
}
else
{
    string[] currentRangeParts = ranges[i].Split('-');
    string[] nextRangeParts = ranges[i + 1].Split('-');
    string[] nextNextRangeParts;
    if (currentRangeParts.Length == 2 && nextRangeParts.Length == 2)
    {
        int currentEnd = int.Parse(currentRangeParts[1]);
        int nextStart = int.Parse(nextRangeParts[0]);
        int nextEnd = int.Parse(nextRangeParts[1]);
        int nextNextStart;
        if (ComparePorts(currentEnd + 1, nextStart) >= 0)
        {
            string mergedRange;
            if (ComparePorts(currentEnd, nextEnd) < 0)
            {
                mergedRange = currentRangeParts[0] + "-" + nextRangeParts[1];
            }
            else mergedRange = currentRangeParts[0] + "-" + currentRangeParts[1];
            if (i != ranges.Count - 2)
            {
                nextNextRangeParts = ranges[i + 2].Split('-');
                nextNextStart = int.Parse(nextNextRangeParts[0]);
                if ((ComparePorts(currentEnd + 1, nextNextStart) >= 0) || (ComparePorts(nextEnd + 1, nextNextStart)
>= 0))
                {
                    ranges[i + 1] = mergedRange;
                }
            }
            else
            {
                mergedResult.Add(mergedRange);
                i++;
            }
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else mergedResult.Add(ranges[i]);
}
else if (currentRangeParts.Length == 1 && nextRangeParts.Length == 2)
{
    int currentStart = int.Parse(currentRangeParts[0]);
    int nextStart = int.Parse(nextRangeParts[0]);
    int nextEnd = int.Parse(nextRangeParts[1]);
    int nextNextStart;
    if (ComparePorts(currentStart + 1, nextStart) == 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[1];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStart = int.Parse(nextNextRangeParts[0]);
            if ((ComparePorts(currentStart + 1, nextNextStart) >= 0) || (ComparePorts(nextEnd + 1, nextNextStart)
>= 0))
            {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		75

```

        ranges[i + 1] = mergedRange;
    }
    else
    {
        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(mergedRange);
    i++;
}
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else if (currentRangeParts.Length == 2 && nextRangeParts.Length == 1)
{
    int currentEnd = int.Parse(currentRangeParts[1]);
    int nextStart = int.Parse(nextRangeParts[0]);
    int nextNextStart;
    if (ComparePorts(currentEnd + 1, nextStart) == 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[0];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStart = int.Parse(nextNextRangeParts[0]);
            if ((ComparePorts(currentEnd + 1, nextNextStart) >= 0) || (ComparePorts(nextStart + 1, nextNextStart)
>= 0))
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {
                mergedResult.Add(mergedRange);
                i++;
            }
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else if (ComparePorts(currentEnd + 1, nextStart) > 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + currentRangeParts[1];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStart = int.Parse(nextNextRangeParts[0]);
            if (ComparePorts(currentEnd + 1, nextNextStart) >= 0)
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {
                mergedResult.Add(mergedRange);
            }
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

```

        i++;
    }
}
else
{
    mergedResult.Add(mergedRange);
    i++;
}
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else if (currentRangeParts.Length == 1 && nextRangeParts.Length == 1)
{
    int currentEnd = int.Parse(currentRangeParts[0]);
    int nextStart = int.Parse(nextRangeParts[0]);
    int nextNextStart;
    if (ComparePorts(currentEnd, nextStart) == 0)
    {
        string mergedRange = currentRangeParts[0];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStart = int.Parse(nextNextRangeParts[0]);
            if (ComparePorts(currentEnd + 1, nextNextStart) >= 0)
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {
                mergedResult.Add(mergedRange);
                i++;
            }
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else if (ComparePorts(currentEnd + 1, nextStart) >= 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[0];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStart = int.Parse(nextNextRangeParts[0]);
            if ((ComparePorts(currentEnd + 1, nextNextStart) >= 0) || (ComparePorts(nextStart + 1, nextNextStart)
>= 0))
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {
                mergedResult.Add(mergedRange);
                i++;
            }
        }
        else
        {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

```

        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else mergedResult.Add(ranges[i]);
}
}
return mergedResult;
}
private List<string> MergeRanges(List<string> ranges)
{
    List<string> mergedResult = new List<string>();
    for (int i = 0; i < ranges.Count; i++)
    {
        if (i == ranges.Count - 1)
        {
            mergedResult.Add(ranges[i]);
        }
        else
        {
            string[] currentRangeParts = ranges[i].Split('-');
            string[] nextRangeParts = ranges[i + 1].Split('-');
            string[] nextNextRangeParts;
            if (currentRangeParts.Length == 2 && nextRangeParts.Length == 2)
            {
                IPAddress currentEndIP = IPAddress.Parse(currentRangeParts[1]);
                IPAddress nextStartIP = IPAddress.Parse(nextRangeParts[0]);
                IPAddress nextEndIP = IPAddress.Parse(nextRangeParts[1]);
                IPAddress nextNextStartIP;
                IPAddress adjustedCurrentEndIP = SubtractIPAddress(currentEndIP);
                IPAddress adjustedNextEndIP = SubtractIPAddress(nextEndIP);
                if (CompareIPAddresses(adjustedCurrentEndIP, nextStartIP) >= 0)
                {
                    string mergedRange;
                    if (CompareIPAddresses(currentEndIP, nextEndIP) < 0)
                    {
                        mergedRange = currentRangeParts[0] + "-" + nextRangeParts[1];
                    }
                    else mergedRange = currentRangeParts[0] + "-" + currentRangeParts[1];
                    if (i != ranges.Count - 2)
                    {
                        nextNextRangeParts = ranges[i + 2].Split('-');
                        nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
                        if ((CompareIPAddresses(adjustedCurrentEndIP, nextNextStartIP) >= 0) ||
                            (CompareIPAddresses(adjustedNextEndIP, nextNextStartIP) >= 0))
                        {
                            ranges[i + 1] = mergedRange;
                        }
                    }
                    else
                    {
                        mergedResult.Add(mergedRange);
                        i++;
                    }
                }
                else
                {
                    mergedResult.Add(mergedRange);
                }
            }
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

```

        i++;
    }
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else if (currentRangeParts.Length == 1 && nextRangeParts.Length == 2)
{
    IPAddress currentStartIP = IPAddress.Parse(currentRangeParts[0]);
    IPAddress nextStartIP = IPAddress.Parse(nextRangeParts[0]);
    IPAddress nextEndIP = IPAddress.Parse(nextRangeParts[1]);
    IPAddress nextNextStartIP;
    IPAddress adjustedCurrentStartIP = SubtractIPAddress(currentStartIP);
    IPAddress adjustedNextEndIP = SubtractIPAddress(nextEndIP);
    if (CompareIPAddresses(adjustedCurrentStartIP, nextStartIP) >= 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[1];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
            if ((CompareIPAddresses(adjustedCurrentStartIP, nextNextStartIP) >= 0) ||
(CompareIPAddresses(adjustedNextEndIP, nextNextStartIP) >= 0))
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {
                mergedResult.Add(mergedRange);
                i++;
            }
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else
    {
        mergedResult.Add(ranges[i]);
    }
}
else if (currentRangeParts.Length == 2 && nextRangeParts.Length == 1)
{
    IPAddress currentEndIP = IPAddress.Parse(currentRangeParts[1]);
    IPAddress nextStartIP = IPAddress.Parse(nextRangeParts[0]);
    IPAddress nextNextStartIP;
    IPAddress adjustedCurrentEndIP = SubtractIPAddress(currentEndIP);
    IPAddress adjustedNextStartIP = SubtractIPAddress(nextStartIP);
    if (CompareIPAddresses(adjustedCurrentEndIP, nextStartIP) == 0)
    {
        string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[0];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
            if ((CompareIPAddresses(adjustedCurrentEndIP, nextNextStartIP) >= 0) ||
(CompareIPAddresses(adjustedNextStartIP, nextNextStartIP) >= 0))
            {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

```

        ranges[i + 1] = mergedRange;
    }
    else
    {
        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(mergedRange);
    i++;
}
}
else if (CompareIPAddresses(adjustedCurrentEndIP, nextStartIP) > 0)
{
    string mergedRange = currentRangeParts[0] + "-" + currentRangeParts[1];
    if (i != ranges.Count - 2)
    {
        nextNextRangeParts = ranges[i + 2].Split('-');
        nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
        if (CompareIPAddresses(adjustedCurrentEndIP, nextNextStartIP) >= 0)
        {
            ranges[i + 1] = mergedRange;
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else
    {
        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else if (currentRangeParts.Length == 1 && nextRangeParts.Length == 1)
{
    IPAddress currentEndIP = IPAddress.Parse(currentRangeParts[0]);
    IPAddress nextStartIP = IPAddress.Parse(nextRangeParts[0]);
    IPAddress nextNextStartIP;
    IPAddress adjustedCurrentEndIP = SubtractIPAddress(currentEndIP);
    IPAddress adjustedNextStartIP = SubtractIPAddress(nextStartIP);
    if (CompareIPAddresses(currentEndIP, nextStartIP) == 0)
    {
        string mergedRange = currentRangeParts[0];
        if (i != ranges.Count - 2)
        {
            nextNextRangeParts = ranges[i + 2].Split('-');
            nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
            if (CompareIPAddresses(adjustedCurrentEndIP, nextNextStartIP) >= 0)
            {
                ranges[i + 1] = mergedRange;
            }
            else
            {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

```

        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(mergedRange);
    i++;
}
}
else if (CompareIPAddresses(adjustedCurrentEndIP, nextStartIP) >= 0)
{
    string mergedRange = currentRangeParts[0] + "-" + nextRangeParts[0];
    if (i != ranges.Count - 2)
    {
        nextNextRangeParts = ranges[i + 2].Split('-');
        nextNextStartIP = IPAddress.Parse(nextNextRangeParts[0]);
        if ((CompareIPAddresses(adjustedCurrentEndIP, nextNextStartIP) >= 0) ||
(CompareIPAddresses(adjustedNextStartIP, nextNextStartIP) >= 0))
        {
            ranges[i + 1] = mergedRange;
        }
        else
        {
            mergedResult.Add(mergedRange);
            i++;
        }
    }
    else
    {
        mergedResult.Add(mergedRange);
        i++;
    }
}
else
{
    mergedResult.Add(ranges[i]);
}
}
else
{
    mergedResult.Add(ranges[i]);
}
}
}
return mergedResult;
}
private IPAddress SubtractIPAddress(IPAddress ipAddress)
{
    byte[] ipAddressBytes = ipAddress.GetAddressBytes();
    int limit = 0;
    foreach (byte element in ipAddressBytes)
    {
        if (element == 255) limit++;
    }
    if ((limit == 4 && radioIPv4) || limit == 16)
    {
        return ipAddress;
    }
    for (int i = ipAddressBytes.Length - 1; i >= 0; i--)
    {
        ipAddressBytes[i]++;
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

```

        if (ipAddressBytes[i] != 0)
        {
            break;
        }
    }
    return new IPAddress(ipAddressBytes);
}
private List<string> ConvertRangesToNetworkWithInverseMask(List<string> ranges)
{
    List<string> networkResult = new List<string>();
    if (radioIPv4)
    {
        for (int i = 0; i < ranges.Count; i++)
        {
            string[] rangeParts = ranges[i].Split('-');
            IPAddress startRangeIP = IPAddress.Parse(rangeParts[0]);
            if (rangeParts.Length == 1)
            {
                networkResult.Add("host " + rangeParts[0]);
            }
            else
            {
                IPAddress networkEndIP = CalculateEndAddressStartRange(startRangeIP);
                IPAddress endRangeIP = IPAddress.Parse(rangeParts[1]);
                if (CompareIPAddresses(networkEndIP, endRangeIP) > 0)
                {
                    IPAddress newInverseMaskIP = CalculateNewInverseMaskBytes(startRangeIP, endRangeIP);
                    IPAddress newMaskIP = InverseMack(newInverseMaskIP);
                    IPAddress newEndIP = IPAddress.Parse(CalculateEndAddress(startRangeIP.ToString(),
newMaskIP.ToString()));
                    if (CompareIPAddresses(newEndIP, startRangeIP) == 0)
                    {
                        networkResult.Add("host " + startRangeIP.ToString());
                    }
                    else
                    {
                        networkResult.Add(startRangeIP.ToString() + " " + newInverseMaskIP.ToString());
                    }
                }
                if (CompareIPAddresses(newEndIP, endRangeIP) != 0)
                {
                    ranges[i] = SubtractIPAddress(newEndIP).ToString() + "-" + endRangeIP.ToString();
                    i--;
                }
            }
        }
        else if (CompareIPAddresses(networkEndIP, endRangeIP) == 0)
        {
            networkResult.Add(startRangeIP.ToString() + " " + CalculateInverseMask(startRangeIP,
networkEndIP).ToString());
        }
        else if (CompareIPAddresses(networkEndIP, endRangeIP) < 0)
        {
            if (CompareIPAddresses(networkEndIP, startRangeIP) == 0)
            {
                networkResult.Add("host " + startRangeIP.ToString());
            }
            else
            {
                networkResult.Add(startRangeIP.ToString() + " " + CalculateInverseMask(startRangeIP,
networkEndIP).ToString());
            }
            ranges[i] = SubtractIPAddress(networkEndIP).ToString() + "-" + endRangeIP.ToString();
            i--;
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

    }
    }
}
else
{
    for (int i = 0; i < ranges.Count; i++)
    {
        string[] rangeParts = ranges[i].Split('-');
        IPAddress startRangeIP = IPAddress.Parse(rangeParts[0]);
        if (rangeParts.Length == 1)
        {
            networkResult.Add("host " + ExpandIPv6Address(rangeParts[0]));
        }
        else
        {
            IPAddress networkEndIP = CalculateEndAddressStartRange(startRangeIP);
            IPAddress endRangeIP = IPAddress.Parse(rangeParts[1]);
            if (CompareIPAddresses(networkEndIP, endRangeIP) > 0)
            {
                IPAddress newInverseMaskIP = CalculateNewInverseMaskBytes(startRangeIP, endRangeIP);
                IPAddress newMaskIP = InverseMack(newInverseMaskIP);
                IPAddress newEndIP = IPAddress.Parse(CalculateEndAddress(startRangeIP.ToString(),
newMaskIP.ToString()));
                if (CompareIPAddresses(newEndIP, startRangeIP) == 0)
                {
                    networkResult.Add("host " + ExpandIPv6Address(startRangeIP.ToString()));
                }
                else
                {
                    networkResult.Add(ExpandIPv6Address(startRangeIP.ToString())
InverseMaskToPrefix(newInverseMaskIP));
                }
                if (CompareIPAddresses(newEndIP, endRangeIP) != 0)
                {
                    ranges[i] = SubtractIPAddress(newEndIP).ToString() + "-" + endRangeIP.ToString();
                    i--;
                }
            }
            else if (CompareIPAddresses(networkEndIP, endRangeIP) == 0)
            {
                networkResult.Add(ExpandIPv6Address(startRangeIP.ToString())
InverseMaskToPrefix(CalculateInverseMask(startRangeIP, networkEndIP)));
            }
            else if (CompareIPAddresses(networkEndIP, endRangeIP) < 0)
            {
                if (CompareIPAddresses(networkEndIP, startRangeIP) == 0)
                {
                    networkResult.Add("host " + ExpandIPv6Address(startRangeIP.ToString()));
                }
                else
                {
                    networkResult.Add(ExpandIPv6Address(startRangeIP.ToString())
InverseMaskToPrefix(CalculateInverseMask(startRangeIP, networkEndIP)));
                }
                ranges[i] = SubtractIPAddress(networkEndIP).ToString() + "-" + endRangeIP.ToString();
                i--;
            }
        }
    }
}
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

        return networkResult;
    }
    private IPAddress InverseMack(IPAddress maskIP)
    {
        byte[] maskBytes = maskIP.GetAddressBytes();
        for (int i = 0; i < maskBytes.Count(); i++)
        {
            maskBytes[i] = (byte)~maskBytes[i];
        }
        return new IPAddress(maskBytes);
    }
    private IPAddress CalculateEndAddressStartRange(IPAddress startRangeIP)
    {
        byte[] bytes = startRangeIP.GetAddressBytes();
        string binaryString = string.Concat(bytes.Select(b => Convert.ToString(b, 2).PadLeft(8, '0')));
        int index = binaryString.LastIndexOf('1');
        if (index != -1)
        {
            binaryString = binaryString.Substring(0, index + 1) + binaryString.Substring(index + 1).Replace('0', '1');
        }
        if (!binaryString.Contains('1'))
        {
            binaryString = new string('1', binaryString.Length);
        }
        byte[] newBytes = Enumerable.Range(0, binaryString.Length / 8)
            .Select(i => Convert.ToByte(binaryString.Substring(i * 8, 8), 2))
            .ToArray();
        return new IPAddress(newBytes);
    }
    private IPAddress CalculateNewInverseMaskBytes(IPAddress startAddress, IPAddress endAddress)
    {
        byte[] startBytes = startAddress.GetAddressBytes();
        byte[] endBytes = endAddress.GetAddressBytes();
        byte[] inverseMaskBytes = new byte[endBytes.Count()];
        for (int i = 0; i < endBytes.Count(); i++)
        {
            inverseMaskBytes[i] = (byte)(startBytes[i] ^ (byte)endBytes[i]);
        }
        string binaryString = string.Concat(inverseMaskBytes.Select(b => Convert.ToString(b, 2).PadLeft(8, '0')));
        int index = binaryString.IndexOf('1');
        if ((binaryString.Length - index == binaryString.Count(c => c == '1')) || (!binaryString.Contains('1'))) return new
        IPAddress(inverseMaskBytes);
        binaryString = binaryString.Substring(0, index) + binaryString.Substring(index, 1).Replace('1', '0') +
        binaryString.Substring(index + 1).Replace('0', '1');
        byte[] newInverseMaskBytes = Enumerable.Range(0, binaryString.Length / 8)
            .Select(i => Convert.ToByte(binaryString.Substring(i * 8, 8), 2))
            .ToArray();
        return new IPAddress(newInverseMaskBytes);
    }
    private IPAddress CalculateInverseMask(IPAddress startAddress, IPAddress endAddress)
    {
        byte[] startBytes = startAddress.GetAddressBytes();
        byte[] endBytes = endAddress.GetAddressBytes();
        byte[] inverseMaskBytes = new byte[endBytes.Count()];
        for (int i = 0; i < endBytes.Count(); i++)
        {
            inverseMaskBytes[i] = (byte)(startBytes[i] ^ (byte)endBytes[i]);
        }
        return new IPAddress(inverseMaskBytes);
    }
    public string ExpandIPv6Address(string address)
    {

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

if (!address.Contains("::"))
{
    return address;
}
if (Regex.IsMatch(address, "::[^:]+::") == false)
{
    string[] groups = address.Split(':');
    int doubleColonIndex = Array.IndexOf(groups, "");
    int missingGroupCount = 8 - (groups.Length - 1);
    string[] missingGroups = new string[missingGroupCount];
    for (int i = 0; i < missingGroupCount; i++)
    {
        missingGroups[i] = "0";
    }
    List<string> expandedGroups = new List<string>(groups);
    expandedGroups.RemoveAt(doubleColonIndex);
    expandedGroups.InsertRange(doubleColonIndex, missingGroups);
    return string.Join(":", expandedGroups.Select(group => group == "" ? "0" : group));
}
else return "0";
}

public string InverseMaskToPrefix(IPAddress inverseMask)
{
    byte[] addressBytes = inverseMask.GetAddressBytes();
    int bitCount = 0;
    foreach (byte b in addressBytes)
    {
        for (int i = 7; i >= 0; i--)
        {
            if ((b & (1 << i)) != 0)
            {
                bitCount++;
            }
        }
    }
    return "/" + ((bitCount - 128) * (-1)).ToString();
}

private void Main_Form_KeyDown(object sender, KeyEventArgs e)
{
    if (e.Control && e.KeyCode == Keys.C)
    {
        Control focusedControl = ActiveControl;
        if (focusedControl is TextBoxBase textBox)
        {
            if (!string.IsNullOrEmpty(textBox.SelectedText))
            {
                Clipboard.SetText(textBox.SelectedText);
            }
        }
        else if (focusedControl is ListBox listBox)
        {
            StringBuilder selectedItemsText = new StringBuilder();
            foreach (object selectedItem in listBox.SelectedItems)
            {
                selectedItemsText.AppendLine(selectedItem.ToString());
            }
            if (selectedItemsText.Length > 0)
            {
                Clipboard.SetText(selectedItemsText.ToString());
            }
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

    }
}
private void Main_Form_Load(object sender, EventArgs e)
{
    this.Left = (Screen.PrimaryScreen.Bounds.Width - this.Width) / 2;
    this.Top = (Screen.PrimaryScreen.Bounds.Height - this.Height) / 2;
    comboBoxRule.SelectedItem = comboBoxRule.Items[0];
    comboBoxProtocol.SelectedItem = comboBoxProtocol.Items[0];
    tooltipAccessList.SetToolTip(textAccessListNumber, "Введіть номер списку доступу");
    tooltipRule.SetToolTip(comboBoxRule, "permit (дозволено): вказує, що пакети, що відповідають умовам ACL,
будуть дозволені пройти через маршрутизатор або пристрій\ndeny(заборонено): вказує, що пакети, що відповідають
умовам ACL, будуть відкинуті і не будуть пропущені через маршрутизатор або пристрій");
    tooltipTraffic.SetToolTip(comboBoxTraffic, "input: цей ланцюжок застосовується до вхідного трафіку, що
приходить на маршрутизатор або на інтерфейс маршрутизатора\output: цей ланцюжок застосовується до вихідного
трафіку, що відправляється з маршрутизатора або через його інтерфейси\nforward: цей ланцюжок застосовується до
трафіку, який проходить через маршрутизатор, але не призначений для маршрутизатора самого по собі");
    tooltipProtocol.SetToolTip(comboBoxProtocol, "Оберіть протокол серед запропонованих, або введіть своє");
    tooltipAddressSource.SetToolTip(textAddressSource, "0.0.0.0, 0.0.0.5\n0.0.0.0-0.0.0.5, 0.0.0.10\n0.0.0.0-15,
10.0.0.0/24\n192.168.1.0 255.255.255.0, 192.168.3.1-20");
    tooltipPortSource.SetToolTip(textPortSource, "Приклад допустимого формату:\n10-20, 36, 50-100");
    tooltipAddressDestination.SetToolTip(textAddressDestination, "0.0.0.0, 0.0.0.5\n0.0.0.0-0.0.0.5, 0.0.0.10\n0.0.0.0-
15, 10.0.0.0/24\n192.168.1.0 255.255.255.0, 192.168.3.1-20");
    tooltipPortDestination.SetToolTip(textPortDestination, "Приклад допустимого формату:\n10-20, 36, 50-100");
}
private void textAccessListNumber_Leave(object sender, EventArgs e)
{
    if (textAccessListNumber.Text.Trim() == "") textAccessListNumber.Text = "100";
}
private void comboBoxRule_Leave(object sender, EventArgs e)
{
    if (comboBoxRule.Text.Trim() == "") comboBoxRule.SelectedItem = comboBoxRule.Items[0];
}
private void comboBoxTraffic_Leave(object sender, EventArgs e)
{
    if (comboBoxTraffic.Text.Trim() == "") comboBoxTraffic.SelectedItem = comboBoxTraffic.Items[0];
}
private void comboBoxProtocol_Leave(object sender, EventArgs e)
{
    if (comboBoxProtocol.Text.Trim() == "") comboBoxProtocol.SelectedItem = comboBoxProtocol.Items[0];
}
private void textAddressSource_Leave(object sender, EventArgs e)
{
    if (textAddressSource.Text.Trim() == "") textAddressSource.Text = "any";
}
private void textPortSource_Leave(object sender, EventArgs e)
{
    if (textPortSource.Text.Trim() == "") textPortSource.Text = "any";
}
private void textAddressDestination_Leave(object sender, EventArgs e)
{
    if (textAddressDestination.Text.Trim() == "") textAddressDestination.Text = "any";
}
private void textPortDestination_Leave(object sender, EventArgs e)
{
    if (textPortDestination.Text.Trim() == "") textPortDestination.Text = "any";
}
private void buttonWriteToFile_Click(object sender, EventArgs e)
{
    if (listBoxResult.Items.Count == 0)
    {
        MessageBox.Show("Список порожній.");
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росіньський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

else
{
    SaveFileDialog saveFileDialog = new SaveFileDialog();
    saveFileDialog.Filter = "Текстові файли (*.txt)|*.txt";
    saveFileDialog.Title = "Зберегти файл";
    if (saveFileDialog.ShowDialog() == DialogResult.OK)
    {
        string filePath = saveFileDialog.FileName;
        try
        {
            using (StreamWriter writer = new StreamWriter(filePath))
            {
                foreach (var item in listBoxResult.Items)
                {
                    writer.WriteLine(item.ToString());
                }
            }
            MessageBox.Show("Файл успішно збережено.");
        }
        catch (Exception ex)
        {
            MessageBox.Show("Сталася помилка під час запису файлу: " + ex.Message);
        }
    }
}
}

private void buttonAddressSourceLoad_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Filter = "Текстові файли (*.txt)|*.txt";
    openFileDialog.Title = "Виберіть файл для завантаження";
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        string filePath = openFileDialog.FileName;
        try
        {
            string fileContent = File.ReadAllText(filePath);
            string processedContent = fileContent.Replace(Environment.NewLine, ", ").TrimEnd(' ', ',');
            textAddressSource.Text = processedContent;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Сталася помилка під час завантаження файлу: " + ex.Message);
        }
    }
}

private void buttonPortSourceLoad_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Filter = "Текстові файли (*.txt)|*.txt";
    openFileDialog.Title = "Виберіть файл для завантаження";
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        string filePath = openFileDialog.FileName;
        try
        {
            string fileContent = File.ReadAllText(filePath);
            string processedContent = fileContent.Replace(Environment.NewLine, ", ").TrimEnd(' ', ',');
            textPortSource.Text = processedContent;
        }
        catch (Exception ex)
        {
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

        MessageBox.Show("Сталася помилка під час завантаження файлу: " + ex.Message);
    }
}
}
private void buttonAddressDestinationLoad_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Filter = "Текстові файли (*.txt)|*.txt";
    openFileDialog.Title = "Виберіть файл для завантаження";
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        string filePath = openFileDialog.FileName;
        try
        {
            string fileContent = File.ReadAllText(filePath);
            string processedContent = fileContent.Replace(Environment.NewLine, ", ").TrimEnd(' ', ',');
            textAddressDestination.Text = processedContent;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Сталася помилка під час завантаження файлу: " + ex.Message);
        }
    }
}
private void buttonPortDestinationLoad_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Filter = "Текстові файли (*.txt)|*.txt";
    openFileDialog.Title = "Виберіть файл для завантаження";
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        string filePath = openFileDialog.FileName;
        try
        {
            string fileContent = File.ReadAllText(filePath);
            string processedContent = fileContent.Replace(Environment.NewLine, ", ").TrimEnd(' ', ',');
            textPortDestination.Text = processedContent;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Сталася помилка під час завантаження файлу: " + ex.Message);
        }
    }
}
private void buttonSettings_Click(object sender, EventArgs e)
{
    this.Enabled = false;
    FormSettings settingsForm = new FormSettings(this);
    settingsForm.Owner = this;
    settingsForm.RadioCisco = radioCisco;
    settingsForm.RadioHuawei = radioHuawei;
    settingsForm.RadioIPv4 = radioIPv4;
    settingsForm.RadioNumber = radioNumber;
    settingsForm.FormClosed += FormSettings_FormClosed;
    settingsForm.Show();
    settingsForm.Top = this.Top+65;
    settingsForm.Left = this.Left+212;
}
private void FormSettings_FormClosed(object sender, FormClosedEventArgs e)
{
    FormSettings settingsForm = (FormSettings)sender;
    radioCisco = settingsForm.RadioCisco;

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

radioHuawei = settingsForm.RadioHuawei;
radioIPv4 = settingsForm.RadioIPv4;
radioNumber = settingsForm.RadioNumber;
if (radioIPv4)
{
    toolTipAddressSource.SetToolTip(textAddressSource, "0.0.0.0, 0.0.0.5\n0.0.0.0-0.0.0.5, 0.0.0.10\n0.0.0.0-15, 10.0.0.0/24\n192.168.1.0 255.255.255.0, 192.168.3.1-20");
    toolTipAddressDestination.SetToolTip(textAddressDestination, "0.0.0.0, 0.0.0.5\n0.0.0.0-0.0.0.5, 0.0.0.10\n0.0.0.0-15, 10.0.0.0/24\n192.168.1.0 255.255.255.0, 192.168.3.1-20");
}
else
{
    toolTipAddressSource.SetToolTip(textAddressSource, "1a2b:4fdf:6541:984f:ac54::fd, 1a2b:4fdf:6541:984f:ac54::1fd\n1a2b:4fdf:6541:984f:ac54::fd-1a2b:4fdf:6541:984f:ac54::fffd\n1a2b:4fdf:6541:984f:ac54::fd-ffff\n::43fd, f51::/64");
    toolTipAddressDestination.SetToolTip(textAddressDestination, "1a2b:4fdf:6541:984f:ac54::fd, 1a2b:4fdf:6541:984f:ac54::1fd\n1a2b:4fdf:6541:984f:ac54::fd-1a2b:4fdf:6541:984f:ac54::fffd\n1a2b:4fdf:6541:984f:ac54::fd-ffff\n::43fd, f51::/64");
}
if (radioCisco || radioHuawei)
{
    toolTipRule.SetToolTip(comboBoxRule, "permit (дозволено): вказує, що пакети, що відповідають умовам ACL, будуть дозволені пройти через маршрутизатор або пристрій\ndeny(заборонено): вказує, що пакети, що відповідають умовам ACL, будуть відкинуті і не будуть пропущені через маршрутизатор або пристрій");
    comboBoxTraffic.Items.Clear();
    comboBoxTraffic.Text = "";
    string[] rules = { "permit", "deny" };
    comboBoxRule.Items.Clear();
    comboBoxRule.Items.AddRange(rules);
    if (radioCisco)
    {
        labelProducer.Text = "Виробник: Cisco";
    }
    else
    {
        labelProducer.Text = "Виробник: Huawei";
    }
    labelTraffic.Enabled = false;
    comboBoxTraffic.Enabled = false;
    labelAccessList.Enabled = true;
    textAccessListNumber.Enabled = true;
    if (radioNumber)
    {
        labelAccessList.Text = "Номер списку доступу:";
        labelAccessList.Left = textAccessListNumber.Left - labelAccessList.Size.Width - 10;
        toolTipAccessList.SetToolTip(textAccessListNumber, "Введіть номер списку доступу");
    }
    else
    {
        labelAccessList.Text = "Ім'я списку доступу:";
        labelAccessList.Left = textAccessListNumber.Left - labelAccessList.Size.Width - 10;
        toolTipAccessList.SetToolTip(textAccessListNumber, "Введіть ім'я ACL");
    }
}
else
{
    toolTipRule.SetToolTip(comboBoxRule, "accept: трафік буде прийнятий і пропущений через мережеве обладнання\ndrop: трафік буде відкинутий і не буде пропущений через мережеве обладнання");
    string[] rules = { "accept", "drop" };
    comboBoxRule.Items.Clear();
    comboBoxRule.Items.AddRange(rules);
    string[] traffics = { "input", "forward", "output" };
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

        comboBoxTraffic.Items.Clear();
        comboBoxTraffic.Items.AddRange(traffics);
        comboBoxTraffic.SelectedItem = comboBoxTraffic.Items[0];
        labelProducer.Text = "Виробник: MikroTik";
        labelTraffic.Enabled = true;
        comboBoxTraffic.Enabled = true;
        labelAccessList.Enabled = false;
        textAccessListNumber.Enabled = false;
    }

    if (settingsForm.CloseSetting)
    {
        if (radioCisco) textAccessListNumber.Text = "100";
        else if (radioHuawei) textAccessListNumber.Text = "2000";
        else textAccessListNumber.Text = "---";
        comboBoxRule.SelectedItem = comboBoxRule.Items[0];
        comboBoxProtocol.SelectedItem = comboBoxProtocol.Items[0];
        textAddressSource.Text = "any";
        textPortSource.Text = "any";
        textAddressDestination.Text = "any";
        textPortDestination.Text = "any";
        listBoxResult.Items.Clear();
        if (radioIPv4) labelVersionIP.Text = "Версія IP-протоколу: IPv4";
        else labelVersionIP.Text = "Версія IP-протоколу: IPv6";
    }
    this.BringToFront();
}
private List<string> resultCisco()
{
    List<string> result = new List<string>();
    int number;
    string accessList = textAccessListNumber.Text.Trim();
    if (radioNumber)
    {
        if (int.TryParse(accessList, out number))
        {
        }
        else
        {
            MessageBox.Show("Неправильний формат: " + accessList);
            error = true;
        }
    }
    rule = comboBoxRule.Text;
    protocol = comboBoxProtocol.Text;
    string input = textAddressSource.Text.Trim();
    List<string> addressSourceResult = new List<string>();
    if (input == "any")
    {
        addressSourceResult.Clear();
        addressSourceResult.Add("any");
    }
    else addressSourceResult = ReadSortNetworkRanges(input);
    input = textPortSource.Text.Trim();
    List<string> portSourceResultCisco = new List<string>();
    if (input == "any")
    {
        portSourceResultCisco.Clear();
        portSourceResultCisco.Add("");
    }
    else
    {
        List<string> portResult = new List<string>();

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		90

```

portResult = ReadPortRanges(input);
foreach (string element in portResult)
{
    string[] portRange = element.Split('-');
    if (portRange.Length == 2)
    {
        portSourceResultCisco.Add("range " + portRange[0] + " " + portRange[1]);
    }
    else portSourceResultCisco.Add("eq " + portRange[0]);
}
}
input = textAddressDestination.Text.Trim();
List<string> addressDestinationResult = new List<string>();
if (input == "any")
{
    addressDestinationResult.Clear();
    addressDestinationResult.Add("any");
}
else addressDestinationResult = ReadSortNetworkRanges(input);
input = textPortDestination.Text.Trim();
List<string> portDestinationResultCisco = new List<string>();
if (input == "any")
{
    portDestinationResultCisco.Clear();
    portDestinationResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();
    portResult = ReadPortRanges(input);
    foreach (string element in portResult)
    {
        string[] portRange = element.Split('-');
        if (portRange.Length == 2)
        {
            portDestinationResultCisco.Add("range " + portRange[0] + " " + portRange[1]);
        }
        else portDestinationResultCisco.Add("eq " + portRange[0]);
    }
}
if (radioNumber)
{
    for (int i = 0; i < addressSourceResult.Count; i++)
    {
        for (int j = 0; j < portSourceResultCisco.Count; j++)
        {
            for (int k = 0; k < addressDestinationResult.Count; k++)
            {
                for (int n = 0; n < portDestinationResultCisco.Count; n++)
                {
                    string item = string.Format("access-list {0} {1} {2} {3} {4} {5} {6}", accessList,
                        rule, protocol, addressSourceResult[i], portSourceResultCisco[j],
                        addressDestinationResult[k], portDestinationResultCisco[n]);
                    result.Add(item.Replace(" ", " "));
                }
            }
        }
    }
}
else
{
    if (radioIPv4)

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```

{
    string item = string.Format("ip access-list extended {0}", accessList);
    result.Add(item);
    for (int i = 0; i < addressSourceResult.Count; i++)
    {
        for (int j = 0; j < portSourceResultCisco.Count; j++)
        {
            for (int k = 0; k < addressDestinationResult.Count; k++)
            {
                for (int n = 0; n < portDestinationResultCisco.Count; n++)
                {
                    item = string.Format("{0} {1} {2} {3} {4} {5}", rule, protocol, addressSourceResult[i],
portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                    result.Add(item.Replace(" ", " "));
                }
            }
        }
    }
    result.Add("exit");
}
else
{
    string item = string.Format("ipv6 access-list {0}", accessList);
    result.Add(item);
    for (int i = 0; i < addressSourceResult.Count; i++)
    {
        for (int j = 0; j < portSourceResultCisco.Count; j++)
        {
            for (int k = 0; k < addressDestinationResult.Count; k++)
            {
                for (int n = 0; n < portDestinationResultCisco.Count; n++)
                {
                    item = string.Format("{0} {1} {2} {3} {4} {5}", rule, protocol, addressSourceResult[i],
portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                    result.Add(item.Replace(" ", " "));
                }
            }
        }
    }
    result.Add("exit");
}
}
return result;
}
private List<string> resultHuawei()
{
    List<string> result = new List<string>();
    int number;
    string accessList = textAccessListNumber.Text.Trim();
    if (radioNumber)
    {
        if (int.TryParse(accessList, out number))
        {}
        else
        {
            MessageBox.Show("Неправильний формат: " + accessList);
            error = true;
        }
    }
    rule = comboBoxRule.Text;
    protocol = comboBoxProtocol.Text;
    string input = textAddressSource.Text.Trim();

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		92

```

List<string> addressSourceResult = new List<string>();
if (input == "any")
{
    addressSourceResult.Clear();
    addressSourceResult.Add("");
}
else
{
    List<string> addressResult = new List<string>();
    addressResult = ReadSortNetworkRanges(input);
    foreach (string element in addressResult)
    {
        addressSourceResult.Add("source-address " + element.Replace("host", ""));
    }
}
input = textPortSource.Text.Trim();
List<string> portSourceResultCisco = new List<string>();
if (input == "any")
{
    portSourceResultCisco.Clear();
    portSourceResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();
    portResult = ReadPortRanges(input);
    foreach (string element in portResult)
    {
        string[] portRange = element.Split('-');
        if (portRange.Length == 2)
        {
            portSourceResultCisco.Add("source-port range " + portRange[0] + " " + portRange[1]);
        }
        else portSourceResultCisco.Add("source-port eq " + portRange[0]);
    }
}
input = textAddressDestination.Text.Trim();
List<string> addressDestinationResult = new List<string>();
if (input == "any")
{
    addressDestinationResult.Clear();
    addressDestinationResult.Add("");
}
else
{
    List<string> addressResult = new List<string>();
    addressResult = ReadSortNetworkRanges(input);
    foreach (string element in addressResult)
    {
        addressDestinationResult.Add("destination-address " + element.Replace("host", ""));
    }
}
input = textPortDestination.Text.Trim();
List<string> portDestinationResultCisco = new List<string>();
if (input == "any")
{
    portDestinationResultCisco.Clear();
    portDestinationResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();

```

```

portResult = ReadPortRanges(input);
foreach (string element in portResult)
{
    string[] portRange = element.Split('-');
    if (portRange.Length == 2)
    {
        portDestinationResultCisco.Add("destination-port range " + portRange[0] + " " + portRange[1]);
    }
    else portDestinationResultCisco.Add("destination-port eq " + portRange[0]);
}
}
if (radioNumber)
{
    if (radioIPv4)
    {
        string item = string.Format("acl number {0}", accessList);
        result.Add(item);
        for (int i = 0; i < addressSourceResult.Count; i++)
        {
            for (int j = 0; j < portSourceResultCisco.Count; j++)
            {
                for (int k = 0; k < addressDestinationResult.Count; k++)
                {
                    for (int n = 0; n < portDestinationResultCisco.Count; n++)
                    {
                        item = string.Format("rule {0} {1} {2} {3} {4} {5} {6}", (n + k + j + i + 1), rule, protocol,
addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                        result.Add(item.Replace(" ", ""));
                    }
                }
            }
        }
        result.Add("return");
    }
    else
    {
        string item = string.Format("ipv6 acl number {0}", accessList);
        result.Add(item);
        for (int i = 0; i < addressSourceResult.Count; i++)
        {
            for (int j = 0; j < portSourceResultCisco.Count; j++)
            {
                for (int k = 0; k < addressDestinationResult.Count; k++)
                {
                    for (int n = 0; n < portDestinationResultCisco.Count; n++)
                    {
                        item = string.Format("rule {0} {1} {2} {3} {4} {5} {6}", (n + k + j + i + 1), rule, protocol,
addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                        result.Add(item.Replace(" ", ""));
                    }
                }
            }
        }
        result.Add("return");
    }
}
}
else
{
    if (radioIPv4)
    {
        string item = string.Format("acl name {0}", accessList);
        result.Add(item);
    }
}
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

```

        for (int i = 0; i < addressSourceResult.Count; i++)
        {
            for (int j = 0; j < portSourceResultCisco.Count; j++)
            {
                for (int k = 0; k < addressDestinationResult.Count; k++)
                {
                    for (int n = 0; n < portDestinationResultCisco.Count; n++)
                    {
                        item = string.Format("rule {0} {1} {2} {3} {4} {5} {6}", (n + k + j + i + 1), rule, protocol,
addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                        result.Add(item.Replace(" ", " "));
                    }
                }
            }
        }
        result.Add("return");
    }
    else
    {
        string item = string.Format("ipv6 acl name {0}", accessList);
        result.Add(item);
        for (int i = 0; i < addressSourceResult.Count; i++)
        {
            for (int j = 0; j < portSourceResultCisco.Count; j++)
            {
                for (int k = 0; k < addressDestinationResult.Count; k++)
                {
                    for (int n = 0; n < portDestinationResultCisco.Count; n++)
                    {
                        item = string.Format("rule {0} {1} {2} {3} {4} {5} {6}", (n + k + j + i + 1), rule, protocol,
addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k], portDestinationResultCisco[n]);
                        result.Add(item.Replace(" ", " "));
                    }
                }
            }
        }
        result.Add("return");
    }
}
return result;
}
private List<string> resultMikroTik()
{
    List<string> result = new List<string>();

    rule = comboBoxRule.Text;
    protocol = comboBoxProtocol.Text;
    string traffic = comboBoxTraffic.Text;
    string input = textAddressSource.Text.Trim();
    List<string> addressSourceResult = new List<string>();
    if (input == "any")
    {
        addressSourceResult.Clear();
        addressSourceResult.Add("");
    }
    else
    {
        List<string> addressResult = new List<string>();
        addressResult = ReadSortNetworkRanges(input);
        foreach (string element in addressResult)
        {
            addressSourceResult.Add("src-address=" + element);
        }
    }
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

```

    }
}
input = textPortSource.Text.Trim();
List<string> portSourceResultCisco = new List<string>();
if (input == "any")
{
    portSourceResultCisco.Clear();
    portSourceResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();
    portResult = ReadPortRanges(input);
    foreach (string element in portResult)
    {
        string[] portRange = element.Split('-');
        if (portRange.Length == 2)
        {
            portSourceResultCisco.Add("src-port-range=" + portRange[0] + "-" + portRange[1]);
        }
        else portSourceResultCisco.Add("src-port=" + portRange[0]);
    }
}
input = textAddressDestination.Text.Trim();
List<string> addressDestinationResult = new List<string>();
if (input == "any")
{
    addressDestinationResult.Clear();
    addressDestinationResult.Add("");
}
else
{
    List<string> addressResult = new List<string>();
    addressResult = ReadSortNetworkRanges(input);
    foreach (string element in addressResult)
    {
        addressDestinationResult.Add("dst-address=" + element);
    }
}
input = textPortDestination.Text.Trim();
List<string> portDestinationResultCisco = new List<string>();
if (input == "any")
{
    portDestinationResultCisco.Clear();
    portDestinationResultCisco.Add("");
}
else
{
    List<string> portResult = new List<string>();
    portResult = ReadPortRanges(input);
    foreach (string element in portResult)
    {
        string[] portRange = element.Split('-');
        if (portRange.Length == 2)
        {
            portDestinationResultCisco.Add("dst-port-range=" + portRange[0] + "-" + portRange[1]);
        }
        else portDestinationResultCisco.Add("dst-port=" + portRange[0]);
    }
}
}
if (radioIPv4)
{

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

```

for (int i = 0; i < addressSourceResult.Count; i++)
{
    for (int j = 0; j < portSourceResultCisco.Count; j++)
    {
        for (int k = 0; k < addressDestinationResult.Count; k++)
        {
            for (int n = 0; n < portDestinationResultCisco.Count; n++)
            {
                string item = string.Format("/ip firewall filter add chain={0} action={1} protocol={2} {3} {4} {5} {6}",
traffic, rule, protocol, addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k],
portDestinationResultCisco[n]);
                result.Add(item.Replace(" ", " "));
            }
        }
    }
}
else
{
    for (int i = 0; i < addressSourceResult.Count; i++)
    {
        for (int j = 0; j < portSourceResultCisco.Count; j++)
        {
            for (int k = 0; k < addressDestinationResult.Count; k++)
            {
                for (int n = 0; n < portDestinationResultCisco.Count; n++)
                {
                    string item = string.Format("/ipv6 firewall filter add chain={0} action={1} protocol={2} {3} {4} {5}
{6}", traffic, rule, protocol, addressSourceResult[i], portSourceResultCisco[j], addressDestinationResult[k],
portDestinationResultCisco[n]);
                    result.Add(item.Replace(" ", " "));
                }
            }
        }
    }
}
return result;
}
private void Main_Form_FormClosed(object sender, FormClosedEventArgs e)
{
    Application.Exit();
}
}
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Россінський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

Код для вікна налаштувань:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace ACL_calculator
{
    public partial class FormSettings : Form
    {
        private Main_Form mainForm;
        public bool RadioCisco { get; set; }
        public bool RadioHuawei { get; set; }
        public bool RadioIPv4 { get; set; }
        public bool RadioNumber { get; set; }
        public bool CloseSetting { get; set; }
        public FormSettings(Main_Form mainForm)
        {
            InitializeComponent();
            this.mainForm = mainForm;
        }
        private void FormSettings_FormClosed(object sender, FormClosedEventArgs e)
        {
            mainForm.Enabled = true;
        }
        private void buttonSave_Click(object sender, EventArgs e)
        {
            RadioCisco = radioCisco.Checked;
            RadioHuawei = radioHuawei.Checked;
            RadioIPv4 = radioIPv4.Checked;
            RadioNumber = radioNumber.Checked;
            CloseSetting = true;
            this.Close();
        }
        private void radioIPv6_CheckedChanged(object sender, EventArgs e)
        {
            if (radioHuawei.Checked == false)
            {
                radioNumber.Enabled = false;
                if (radioCisco.Checked == true) radioName.Checked = true;
            }
        }
        private void radioIPv4_CheckedChanged(object sender, EventArgs e)
        {
            if (radioCisco.Checked == true || radioHuawei.Checked == true) radioNumber.Enabled = true;
        }
        private void FormSettings_Load(object sender, EventArgs e)
        {
            if (RadioCisco) radioCisco.Checked = true;
            else if (RadioHuawei) radioHuawei.Checked = true;
            else radioMikroTik.Checked = true;
            if (RadioIPv4) radioIPv4.Checked = true;
            else radioIPv6.Checked = true;
            if (RadioNumber) radioNumber.Checked = true;
            else if (radioMikroTik.Checked == false) radioName.Checked = true;
        }
    }
}
```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		98

```

    }
private void radioMikroTik_CheckedChanged(object sender, EventArgs e)
{
    if (radioCisco.Checked || radioHuawei.Checked)
    {
        radioNumber.Checked = true;
        radioNumber.Enabled = true;
        radioName.Enabled = true;
    }
    else
    {
        radioNumber.Checked = false;
        radioName.Checked = false;
        radioNumber.Enabled = false;
        radioName.Enabled = false;
    }
}
private void radioHuawei_CheckedChanged(object sender, EventArgs e)
{
    radioNumber.Enabled = true;
    radioName.Enabled = true;
}
private void radioCisco_CheckedChanged(object sender, EventArgs e)
{
    if (radioIPv4.Checked) radioNumber.Enabled = true;
    else
    {
        radioNumber.Enabled = false;
        radioName.Checked = true;
    }

    radioName.Enabled = true;
}
}
}
}

```

		Мисюк Є.Ю.			«Житомирська політехніка».23.125.11.000	Арк.
		Росієнський Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		99