ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ

Пояснювальна записка

до магістерської роботи
на тему:

«ТЕХНОЛОГІЯ ЗАБЕЗПЕЧЕННЯ ЗАХИЩЕНОГО ОБМІНУ ДАНИМИ В КОПРОПАТИВНІЙ МЕРЕЖІ З ВИКОРИСТАННЯМ РІШЕННЯ WIREGUARD»

Виконав студент 6 курсу, групи БСДМ-62
спеціальності 125 Кібербезпека
освітньо-професійної програми«Інформаційна та кібернетична безпека»
(шифр і назва спеціальності)
Добреля В.О.
(прізвище та ініціали)
Керівник Гахов С.О.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтролер Чумак Н.С.
(прізвище та ініціали)

КИЇВ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Інститут ННІЗІ
Кафедра Інформаційної та кібернетичної безпеки
Ступінь вищої освіти Магістр
Спеціальність 125 Кібербезпека
Освітньо-професійна програма Інформаційна та кібернетична безпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ІКБ
_____ Гайдур Г.І.
"___"_____ 2021 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Добрелі Віталію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема магістерської роботи: «Технологія забезпечення захищеного обміну даними в корпоративній мережі з використанням рішення WireGuard»

керівник магістерської роботи Гахов Сергій Олександрович, к.військ.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від «11» жовтня 2021 року №170.

2. Строк подання студентом магістерської роботи 15.12.2021 р.

3. Вихідні дані до магістерської роботи
концепція побудови хмарних сервісів VPN
науково-технічна література експлуатаційна
документація, нормативні документи, міжнародні стандарти.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1.Актуальність проблеми забезпечення захищеного обміну даними в корпоративній мережі.
2.Аналіз методів та засобів побудови VPN на базі протоколу WireGuard.
3.Нові механізми роботи протоколу VPN WireGuard на базі Linux.
4.Технологія розгортання VPN WireGuard на базі Linux з використанням різних топологій мереж.

5. Перелік графічного матеріалу

1.
2.
3.
5.
6.
7.
8.
9.
10.

6. Дата видачі завдання 27.09.2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1.	Визначення актуальності проблеми забезпечення захищеного обміну даними в корпоративній мережі.	27.09.2021 р.	
2.	Аналіз наукової та технічної літератури з питань теми магістерської роботи.	12.10.2021 р.	
3.	Аналіз методів та засобів побудови VPN на базі протоколу WireGuard	23.10.2021 р.	
4.	Розроблення технології розгортання VPN WireGuard на базі Linux з використанням різних топологій мереж	07.11.2021 р.	
5.	Розроблення рекомендацій щодо застосування рішення WireGuard для забезпечення захищеного обміну даними в корпоративній мережі	27.11.2021 р.	
6.	Оформлення результатів дослідження. Проходження плагіату	02.122021 р.	
7.	Підготовка доповіді до захисту.	15.12.2021 р.	

Студент Добреля В.О.
(підпис) прізвище та ініціали

Керівник магістерської роботи Гахов С.С
(підпис) прізвище та ініціали

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ОС – операційна система.

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

ЦОД – (Data center) – центр опрацювання даних.

IP – (Internet Protocol) – міжмережевий протокол.

IPSec – (IP Security) – протокол захисту даних, що передається через IP.

LAN – (Local Area Network) – локальна мережа.

MTU – (Maximum transmission unit) – максимальний розмір корисного блоку даних одного пакета, який може бути переданий протоколом без фрагментації.

PPTP – (Point-to-Point Tunneling Protocol) – тунельний протокол точка-точка

TCP – (Transmission Control Protocol) – протокол керування передачею даних.

UDP – (User Datagram Protocol) – протокол котрий виконує обмін повідомленнями без підтвердження та гарантії доставки.

VPN - (Virtual Private Network) – віртуальна приватна мережа.

VLAN – (Virtual Local Area Network) – віртуальна локальна мережа.

VPLS – (Virtual Private Local Area Network Service) – віртуальна приватна локальна мережа.

VPS – (Virtual Private Server) – віртуальний виділений сервер

WAN – (Wide Area Network) – глобальна мережа.

ВСТУП

Об'єкт дослідження – процес забезпечення захищеного обміну даними в корпоративній мережі.

Предмет дослідження – технології забезпечення захисту даних при передачі їх по незахищеним мережам.

Мета роботи – розробити варіант технології безпечного обміну даними в корпоративній мережі за допомогою рішення VPN WireGuard та рекомендації щодо застосування даного протоколу на підприємстві.

Наукові завдання:

дослідити ступінь проблеми забезпечення захищеного обміну даних в корпоративній мережі

встановити сутність завдань захисту обміну даними в корпоративній мережі

проаналізувати існуючі технології захисту обміну даними в корпоративній мережі

проаналізувати методи та засоби забезпечення захищеного обміну даних в корпоративній мережі

проаналізувати основні функції та принципи реалізації захисту обміну даними в корпоративній мережі

Методи дослідження – опрацювання літератури за даною темою, аналіз експлуатаційної документації, міжнародних стандартів та їх порівняння, моделювання процесу безпечного обміну даних в корпоративній мережі.

Практичне значення одержаних результатів: запропоновано порядок застосування технології захисту обміну даними в корпоративній мережі із застосуванням рішення VPN WireGuard, а також розроблено рекомендації щодо застосування рішення WireGuard для забезпечення захищеного обміну даними в корпоративній мережі.

Результати магістерської роботи апробовані на Всеукраїнській науковій конференції «Актуальні проблеми кібербезпеки» яка відбулася 27 жовтня 2021 року в Державному університеті телекомунікацій, м. Київ.

1 АНАЛІЗ ПРОБЛЕМИ ЗАБЕЗПЕЧЕННЯ ЗАХИЩЕНОГО ОБМІНУ ДАННИМИ В КОРПОРАТИВНІЙ МЕРЕЖІ

1.1 Тенденція зростання популярності використання співробітниками власних пристроїв в корпоративних мережах

Мільйони споживачів, багато з яких також є співробітниками, купують сучасні мобільні пристрої, такі як смартфони та планшети, для особистого використання, а потім завантажують на них додатки, які допомагають краще керувати своїм життям. Ці потужні пристрої мають інтуїтивно зрозумілий інтерфейс користувача, оснащені камерами двостороннього відеозв'язку і можуть отримувати доступ до сотень тисяч програм не тільки для особистого використання і розваг, але і для бізнесу.

Все частіше люди беруть ці пристрої на роботу та інтегрують їх у свій повсякденний робочий процес. Цю тенденцію часто називають "принеси свій власний пристрій" (BYOD).

BYOD може мати серйозні наслідки для того, як компанії керують своїми мережами, мобільними пристроями та навіть своїми співробітниками, які переглядають поняття «бути в офісі». Спираючись на дослідження проведене Cisco IBSG в процесі якого було опитано близько 4900 IT-керівників у 18 галузях на підприємствах з 1000 і більше співробітників, так і в компаніях середнього розміру з кількістю співробітників від 500 до 999 осіб у восьми країнах з трьох регіонів, в результаті було заявлено про фундаментальне зрушення у тому, як корпорації підтримують та надають пристрої своїм співробітникам. 95% керівників IT-служб заявили, що їхні компанії у тій чи іншій формі підтримують BYOD. Це має серйозні наслідки для того, як компанії надають пристрої та регулюють доступ до мережі. Співробітники хочуть мати більший контроль над тим, де і як вони працюють і які інструменти вони використовують для виконання роботи. Вони хочуть менше приділяти уваги речам, які є другорядними для

досягнення успіху в їхніх службових ролях, наприклад, проводити особистий час в офісі або вибирати між вузьким колом пристроїв та додатків.

Мобільні пристрої тепер настільки складні і в той же час прості у використанні, а вибір мобільних додатків та хмарних сервісів настільки широкий, що вимоги використовувати запропоновані компанією інструменти не підтримуються багатьма працівниками.

Вони хочуть, щоб компанії довіряли їм виконувати роботу так, як вони вважають за потрібне, і в найбільш зручний час дня. Прагнення співробітників до більшої свободи, контролю та гнучкості може змусити деякі компанії замислитись, оскільки це потенційно ускладнює як надання IT-послуг, так і управління ними. Але взаємно посилюючі переваги для компанії та співробітників є суттєвими. BYOD може допомогти співробітнику збалансувати роботу з соціальними та сімейними зобов'язаннями. Це природно узгоджується з сімейною політикою, такою як віддалена робота та гнучкий графік. Компанії можуть використовувати переваги BYOD, подібні до цих, для залучення та утримання талановитих співробітників, а також для забезпечення їх продуктивності, де б вони не знаходилися.

Крім цих переваг, надання працівникам свободи вибору пристроїв, додатків та хмарних сервісів, які вони використовують, дозволяє змінити робочі процеси. Інновації під керівництвом співробітників виходять далеко за межі того, коли і де їм працювати. За допомогою BYOD співробітники можуть постійно додавати нововведення безліччю способів, таких як використання хмарних сервісів для аналізу та візуалізації даних на мобільному пристрої, відкриття ідеального інструменту для керування складними робочими процесами або запис відеоконференцій для спрощення виконання завдань.

Мобільність - робота віддалено від традиційного офісу або фіксованого розташування - стала звичайною вимогою для сьогоднішніх працівників розумової праці. 47% співробітників опитаних компаній офіційно визнано «віддаленими працівниками». Але 60% співробітників використовують мобільні пристрої для роботи навіть перебуваючи в офісі, що на 13% більше, ніж було

заявлено офіційних віддалених співробітників.

Ці додаткові пристрої здебільшого є результатом ініціативи співробітників: навіть якщо їм офіційно не потрібні мобільні пристрої для виконання обов'язків, вони інтегрують їх у повсякденну роботу. Зростання кількості пристроїв на одного користувача значною мірою є результатом BYOD. Наприклад, 42% смартфонів та 38% ноутбуків, які використовуються на робочому місці, тепер належать співробітникам. Це показує, що BYOD – далеко не нова тенденція, і вже міцно укорінилася у корпораціях по всьому світу.

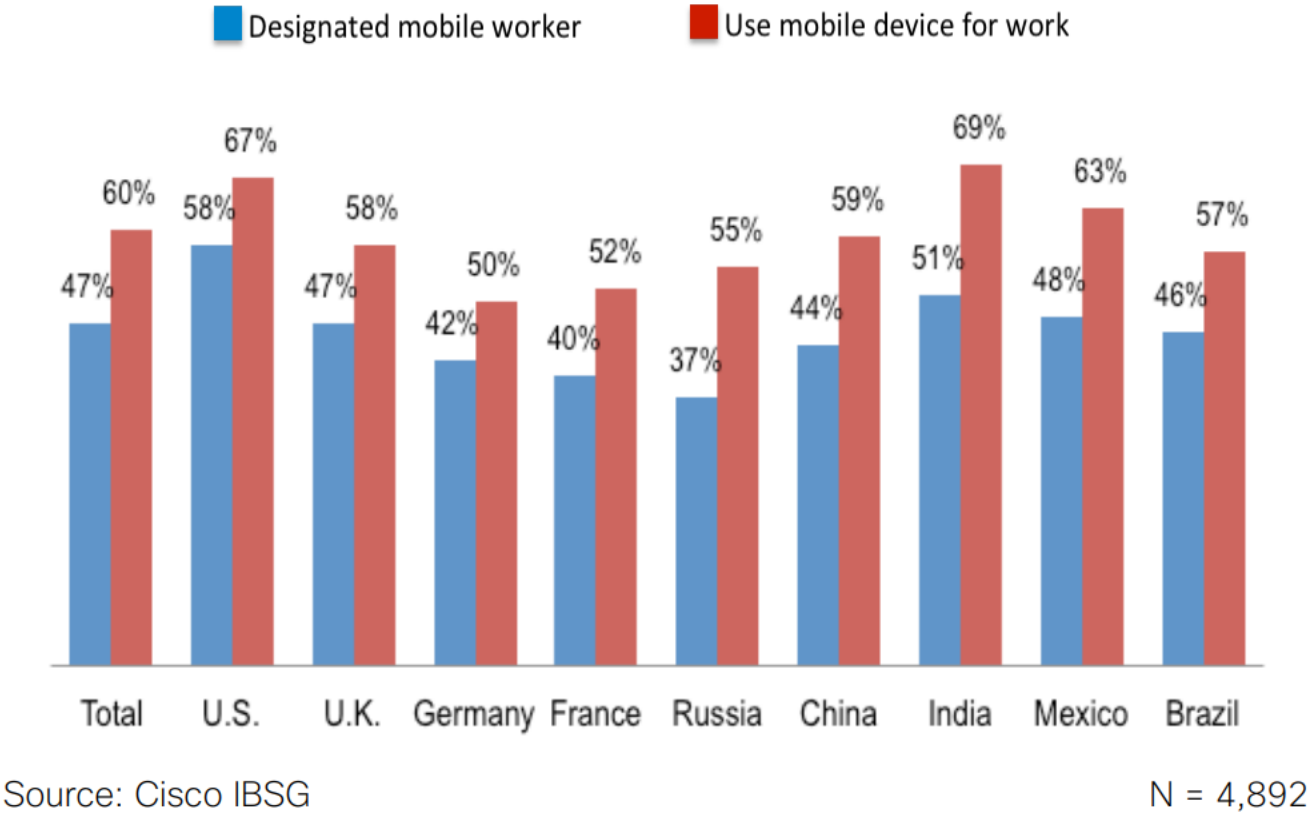

Рис. 1.1. Відсоток віддалених працівників, порівняно з офісними працівниками, які використовують мобільні пристрої для роботи [2]

Після того, як використання пристроїв, що належать співробітникам, було дозволено та отримало підтримку, кількість мобільних пристроїв, підключених до корпоративної мережі, збільшилася більш ніж удвічі. Поряд із зростанням кількості пристроїв, що належать співробітникам, зростає використання сторонніх додатків та хмарних сервісів. Є сенс підтримувати використання сторонніх

додатків, оскільки співробітники хочуть використовувати не просто пристрій на свій вибір, але також програмне забезпечення та хмарні сервіси, яким вони віддають перевагу. Який би набір інструментів не надавали компанії, існують тисячі мобільних програм та онлайн-сервісів, які можуть допомогти співробітникам у спільній роботі, аналізі даних та представленні своїх ідей. У магазинах програм Apple та Google співробітники можуть переглядати та завантажувати їх миттєво, часто безкоштовно. Крім того, співробітники хочуть використовувати свої власні пристрої, щоб мати під рукою свої особисті програми та контент. Здатність працівників без проблем поєднувати особисте та робоче життя є однією з ключових переваг BYOD.

Використання співробітниками власних додатків пов'язано з потенційними витратами для компаній. Співробітники все частіше використовують обмін повідомленнями, керування файлами, мобільний відеозв'язок та хмарні служби спільної роботи, щоб доповнити або навіть замінити корпоративні програми для спільної роботи. Хоча співробітникам може бути зручніше користуватися інструментами, які вони вибирають самі, особливо якщо ті, які їм надані компанією обмежені, існує можливість виникнення безлічі самостворених «острівців» співпраці, які мимоволі виключають інших працівників. Для компаній життєво важливо забезпечити інтеграцію сторонніх інструментів спільної роботи в корпоративні системи обміну повідомленнями, корпоративні каталоги, корпоративне соціальне програмне забезпечення та інструменти відеозв'язку для запобігання цьому. Загалом корпоративні IT-лідери схвалюють BYOD. Майже 90% підтримують BYOD у тій чи іншій формі, починаючи від простого дозволу використання особистих пристроїв у корпоративній мережі до надання повної IT підтримки для всіх пристроїв, що належать співробітникам.

Але лише 50% представників великого і 41% середнього бізнесу з тих компаній, що брали участь у дослідженні, мають політики безпеки що визначають, хто знаходиться в корпоративній мережі і з яким пристроєм. Це проблематично, навіть якщо переважна більшість пристроїв належить компанії; ризики мережевої безпеки, інтелектуальної власності та корпоративних даних

зростають у міру того, як співробітники підключають до мережі все більше власних пристроїв. У цьому сенсі політики та інструменти безпеки слід розглядати як такі, що забезпечують BYOD, роблячи мережу безпечнішою.

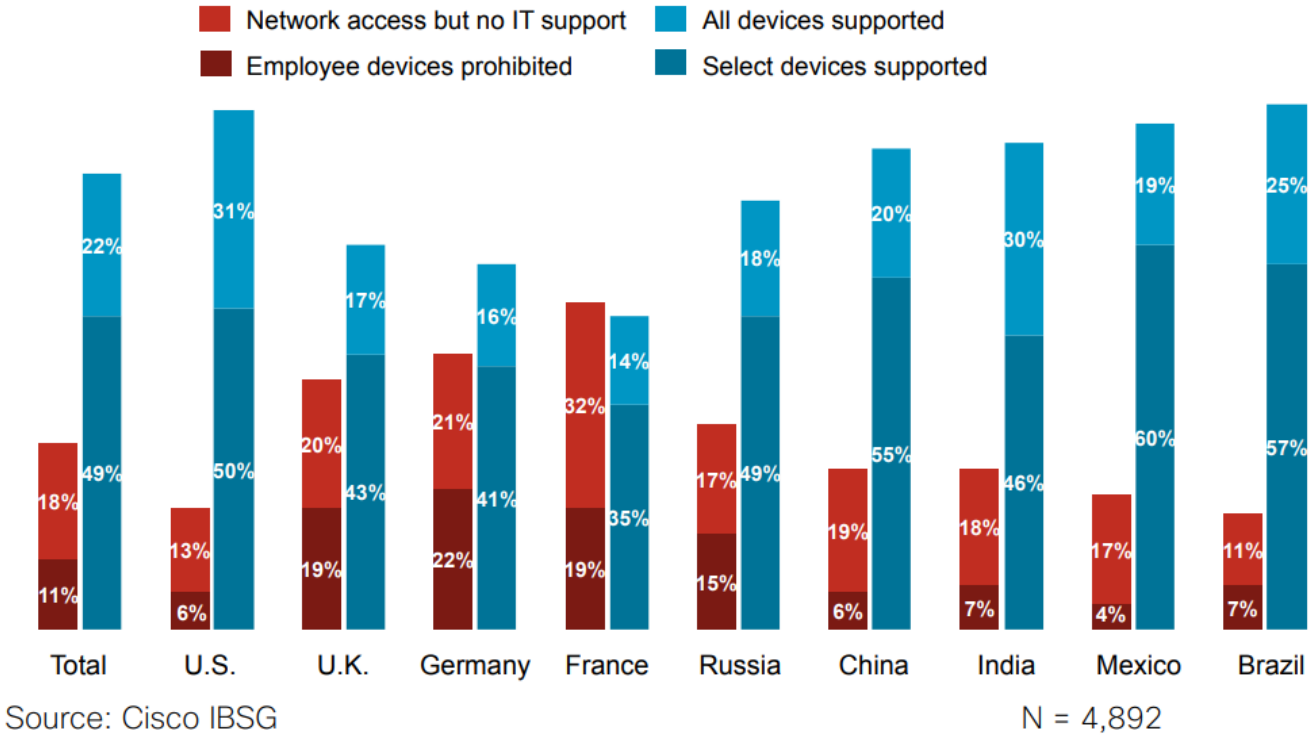

Рис.1.2. Рівні IT-підтримки пристроїв, що належать працівникам [2]

Безпечна передача даних за допомогою шифрування дозволяє компаніям надавати навіть конфіденційні дані мобільним користувачам незалежно від пристрою або розташування.

1.2 VPN – загальні принципи

Головною метою VPN є організація доступу до мережі компанії за допомогою мереж загального користування. VPN забезпечує конфіденційність, цілісність та доступність інформації. Тобто гарантує те, що інформація не буде доступна небажаним особам, буде збережена та доступна лише визначеним користувачам. Дані характеристики забезпечується за допомогою основних компонентів VPN:

Шифрування трафіку;

Аутентифікація;

Підтримка великої кількості протоколів;

Тунелювання.

Шифрування трафіку включає в себе кодування інформації для запобігання отримання даних третіми особами. Доступ до такої інформації будуть мати лише ті користувачі, яким для дешифрування було надано відповідний ключ. Шифрування має бути складним для забезпечення конфіденційності інформації, що передається в межах періоду, до якого вона буде актуальна. Алгоритм шифрування повинен протидіяти протизаконному дешифруванню трафіку на довгий період.

Щодо аутентифікації, то мається на увазі, що на центральному сервері відбувається аутентифікація користувачів. Також може відбуватися взаємна аутентифікація з'єднаних вузлів. Для більшої безпеки рекомендується використовувати двофакторну аутентифікацію. Для отримання доступу до мережі користувач VPN проходить через брандмауер, де здійснюється його аутентифікація і авторизація, завдяки чому VPN гарантує, що доступ до ресурсів мережі отримають лише авторизовані користувачі.

Протоколи VPN визначають рівень захищеності трафіку та те як VPN взаємодіє з іншими системами в Інтернеті.

Тунелювання вказує на те, що VPN створює відокремлений канал між з'єднаними пристроями. При цьому кожен кінцевий вузол VPN здатен забезпечувати кілька одночасних з'єднань з іншими вузлами. Водночас за допомогою шифрування трафік розділяється, і всі вузли є відокремленими один від одного. Це дозволяє зробити приватну інформацію невидимою для інших користувачів Web і забезпечує додатковий захист при передачі.

За своєю суттю, VPN володіє багатьма властивостями виділеної лінії, проте реалізується вона в межах загальнодоступного середовища Інтернет.

Проводячи порівняння між приватними і віртуальними приватними мережами, слід виділити ряд безсумнівних переваг VPN:

технологія VPN дозволяє значно знизити витрати для утримання

працездатності мережі;

користувач сплачує тільки абонентську плату за оренду каналу. До речі, оренда каналів також не викликає будь-яких труднощів внаслідок широкомасштабності мережі Інтернет;

зручність і легкість при організації та перебудови структури мережі.

Розробка єдиної моделі обслуговування віртуальної приватної мережі могла б спростити мережеві операції, але такий підхід не може задовольнити різним вимогам клієнтів, так як вони унікальні. Кожен клієнт висуває свої вимоги до безпеки, числу сайтів, складності маршрутизації, критичних ситуацій, моделям і обсягами трафіку. Для задоволення широкого спектру вимог, постачальники послуг повинні пропонувати клієнтам різні моделі доставки послуг.

Всі мережі VPN умовно можна розділити на три основні види:

Внутрішнькорпоративні VPN (Intranet VPN).

Міжкорпоративні VPN (Extranet VPN).

VPN з віддаленим доступом (Remote Access VPN).

Intranet VPN являє собою найбільш простий варіант VPN, він дозволяє об'єднати в єдину захищену мережу кілька розподілених філій однієї організації, взаємодіючих по відкритих каналах зв'язку. При організації такої схеми підключення необхідна наявність VPN серверів, яке дорівнює кількості пов'язаних офісів.

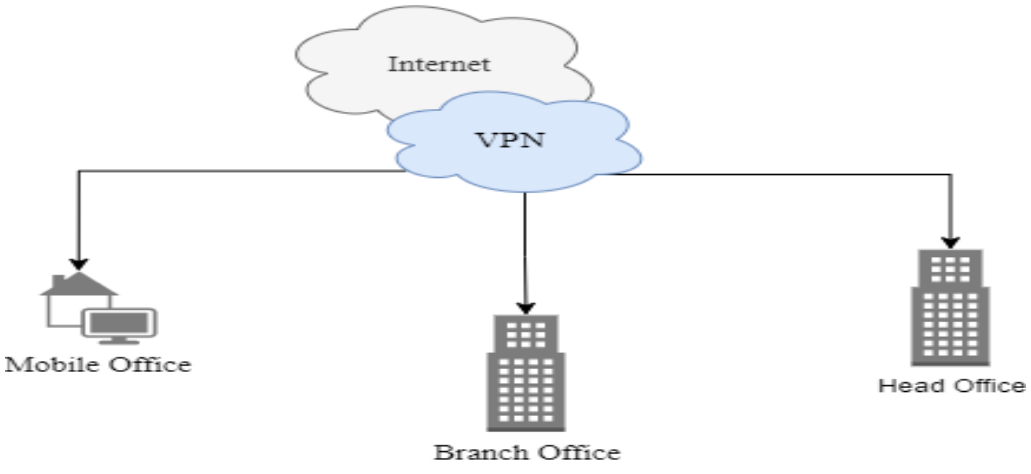

Рис. 1.3. Приклад мережі Intranet VPN

Варіант побудови Extranet VPN призначений для забезпечення доступу з мережі однієї компанії до ресурсів мережі іншого, рівень довіри до якої набагато нижче, ніж до своїх співробітників. Тому, коли кілька компаній приймають рішення працювати разом і відкривають один для одного свої мережі, вони повинні подбати про те, щоб їх нові партнери мали доступ тільки до певної інформації.

При цьому, конфіденційна інформація повинна бути надійно захищена від несанкціонованого використання. Саме тому в міжкорпоративних мережах велике значення має надаватися контролю доступу за допомогою брандмауерів (Firewalling). Важлива і аутентифікація користувачів, яка покликана гарантувати, що доступ до інформації отримують тільки ті, кому він дійсно дозволений.

Принцип роботи VPN з віддаленим доступом простий: користувачі встановлюють з'єднання з місцевої точкою доступу до глобальної мережі (POP), після чого їх виклики тунелюють через Інтернет, що дозволяє уникнути плати за міжміський і міжнародний зв'язок або виставлення рахунків власникам безкоштовних міжміських номерів (Toll-free Numbers).

Потім всі виклики концентруються на відповідних вузлах і передаються в корпоративні мережі. Однак через використання Інтернету в якості об'єднуючої магістралі, механізми захисту інформації стають життєво важливими елементами даної технології.

Як правило, віддалений користувач не має статичної адреси і підключається до захищених ресурсів не через виділений пристрій VPN, а за допомогою спеціального програмного забезпечення, що встановлюється на його комп'ютері.

Важливу роль при побудові VPN грають відносини підприємства з провайдером, зокрема, розподіл між ними функцій по конфігурації і експлуатації VPN-пристроїв.

При створенні захищених каналів VPN-засоби можуть розташовуватися як в середовищі обладнання провайдера, так і в обладнанні підприємства. Залежно від цього, виділяють два варіанти побудови VPN: призначена для користувача схема (Customer Provided VPN) і провайдерська схема (Provider Provisioned VPN).

Крім названої вище класифікації, всі варіанти створення VPN можна розділити на дві категорії: програмні і апаратні.

Програмні рішення являють собою готові програми, які встановлюються на час огляду мережі комп'ютера зі стандартним програмним забезпеченням.

Апаратні VPN-рішення включають в себе комп'ютер, операційну систему, спеціальне програмне забезпечення.

Віртуальні приватні мережі можна вважати повноцінним видом транспорту для передачі трафіку, тільки якщо є гарантії на пропускну здатність та інші параметри продуктивності, а також безпеку переданих даних.

1.3 Класифікація VPN

Існує декілька класифікацій VPN по різноманітним базовим параметрам.

За рівнем захисту:

Довірительні. Використовуються для налаштування віртуальної нової мережі під основною мережею. При цьому не приділяється увага проблемам забезпечення безпеки, оскільки середа для передачі даних є довірчою;

Захищені. Використовуються для налаштування надійних і високозахищених мереж на основі існуючих незахищених мереж.

За реалізацією:

Інтегроване рішення. Використовується програмно-апаратний комплекс, який налаштовує мережевий екран та фільтрує трафік;

Програмно-апаратний комплекс. Реалізується за рахунок спеціалізованих програмно-апаратних засобів;

Програмний комплекс. Використовується комп'ютер клієнта зі спеціальним програмним забезпеченням.

За призначенням:

Intranet. Об'єднує філії організації у високозахищену мережу для обміну інформацією за рахунок створення відкритих каналів;

Extranet. Об'єднує кілька різних компаній в єдину мережу, але для

небажаних користувачів обмежується доступ до конфіденційної інформаці;.

Remote Access. Створюється захищений канал між віддаленим співробітником та мережею компанії;

Client/Server. Поділяє фізичну мережу на декілька логічних підмереж;

Internet. Забезпечує доступ по спільному фізичному каналу для декількох користувачів у мережі провайдера;

За рівнем моделі OSI:

Канальний (протоколи SLIP, PPP, PPTP, L2TP). Здійснює інкапсуляцію декількох типів трафіку, а також створює віртуальні тунелі за принципом точка-точка;

Мережевий (протоколи IPsec, MPLS). Забезпечує інкапсуляцію одного IP пакету в інший;

Транспортний (протоколи SSL/TLS). Забезпечує цілісність і конфіденційність даних, управляє ключами в процесі передачі даних, а також здійснює аутентифікацію відправника та отримувача;

Сеансовий (протокол SOCKS). Забезпечує підтримку необхідних програм, для яких потрібно встановити умови доступу для користувачів та спрямувати інформаційний потік.

За видом протоколу:

Можна виділити VPN під IPX, AppleTalk і TCP/IP. Найбільш популярним є TCP / IP.

1.4 Функції VPN по захисту даних

Підключення будь-якої корпоративної мережі до публічної викликає два типи загроз:

несанкціонований доступ до ресурсів локальної мережі, отриманий в результаті входу в цю мережу;

несанкціонований доступ до даних при передачі трафіку по публічній мережі;

Віртуальні приватні мережі є комбінацією декількох самостійних сервісів (механізмів) безпеки:

шифрування (з використання інфраструктури криптосистем) на виділених шлюзах (шлюз забезпечує обмін даними між обчислювальними мережами, що функціонують по різних протоколах);

екранування (з використанням міжмережевих екранів);

тунелювання.

При цьому узгоджуючим сторонам необхідна платформа підключення, яка не тільки швидко масштабується, а й (в першу чергу) забезпечує конфіденційність даних, цілісність даних і аутентифікацію.

Функції VPN по захисту даних полягають в наступному:

На всі комп'ютери, що мають вихід в Інтернет (замість Інтернет може бути і будь-яка інша мережа загального користування), встановлюється «VPN-агенти» - засіб, що реалізовує VPN, мережевий шлюз, які обробляють IP-пакети, що передаються по мережах;

Для корпоративного зв'язку в великих організаціях або об'єднання віддалених один від одного офісів для реалізації VPN-технологій в ролі шлюзу можуть виступати: сервера Unix, сервера Windows, мережевий маршрутизатор і мережевий шлюз на якому розгорнуто VPN.

Перед відправкою IP-пакету «VPN-агент» виконує наступні операції:

аналізується IP-адреса одержувача пакета, в залежності від цієї адреси вибирається алгоритм захисту даного пакету(VPN-агенти можуть, підтримувати одночасно кілька алгоритмів шифрування і контролю цілісності). Пакет може і зовсім бути відкинутий, якщо в настройках VPN-агента такий одержувач не значиться;

обчислюється і додається в пакет його імітоприставка (криптографічний контрольна сума, розрахована з використанням ключа шифрування), що забезпечує контроль цілісності переданих даних;

пакет шифрується (цілком, включаючи заголовок IP-пакета, що містить службову інформацію);

формується новий заголовок пакета, де замість адреси одержувача вказується адреса його VPN-агента (ця процедура називається інкапсуляцією пакета).

В результаті цього, обмін даними між двома локальними мережами зовні представляється як обмін між двома комп'ютерами, на яких встановлені VPN-агенти. Будь-яка корисна для зовнішньої атаки інформація, наприклад, внутрішні IP-адреси мережі, в цьому випадку недоступна.

При отриманні IP-пакета виконуються зворотні дії:

заголовка пакета витягується інформація про VPN-агента відправника пакета, якщо такий відправник не входить в число дозволених, то пакет відкидається (те ж саме відбувається при прийомі пакету з навмисно або випадково пошкодженим заголовком);

згідно з налаштуваннями вибираються криптографічні алгоритми і ключі, після чого пакет розшифровується і перевіряється його цілісність (пакети з порушеною цілісністю також відкидаються);

після всіх зворотних перетворень пакет в його початковому вигляді відправляється справжньому адресату по локальній мережі.

Таким чином, інформаційний потік по громадській мережі передається по захищеному каналу. Для створення захищеного каналу кошти VPN використовують процедури шифрування, аутентифікації і авторизації.

Методів шифрування досить багато, тому важливо, щоб на кінцях тунелю використовувався один і той же алгоритм шифрування. Крім того, для успішного дешифрування даних джерела і одержувача даних необхідно обмінятися ключами шифрування. Слід зазначити, що шифрування повідомлень необхідно не завжди. Часто воно виявляється досить дорогою процедурою, що вимагає додаткових приставок для маршрутизаторів, без яких вони не можуть одночасно з шифруванням забезпечувати прийнятний рівень швидкодії.

Під аутентифікацією розуміється визначення користувача або кінцевого пристрою. Аутентифікація дозволяє встановлювати з'єднання лише між легальними користувачами і, відповідно, запобігає доступ до ресурсів мережі

несанкціонованих користувачів.

У процедурі беруть участь дві сторони: одна доводить свою автентичність, а інша її перевіряє і приймає рішення.

Найчастіше для аутентифікації використовується пароль, але можуть застосовуватися і інші докази. Недоліками застосування паролів є їх розкриття, що частково компенсується їх простотою.

Аутентифікація даних свідчить про їх цілісності, а також про те, що вони надійшли від одного з учасників (при цьому використовується електронний підпис).

Авторизація передбачає надання абонентам різних видів послуг. Кожному користувачеві надаються певні адміністратором права доступу.

Ця процедура виконується після процедури аутентифікації і дозволяє контролювати доступ санкціонованих користувачів до ресурсів мережі.

Взагалі, процедури аутентифікації і авторизації виконують одну задачу і до них пред'являються однакові вимоги.

Цілісність переданих даних дозволяє забезпечити застосування електронного підпису.

1.5 Застосування тунелів для VPN

Для передачі даних VPN-агенти створюють віртуальні канали між захищеними локальними мережами або комп'ютерами (такий канал називається «тунелем», а технологія його створення називається «тунелювованням»).

На рис.1.4. представлений принцип роботи в режимі тунелю, в якому комп'ютери інтрамереж можуть взаємодіяти з комп'ютерами інший інтрамережі шляхом маршрутизації пакетів через тунель IPsec між двома шлюзами.

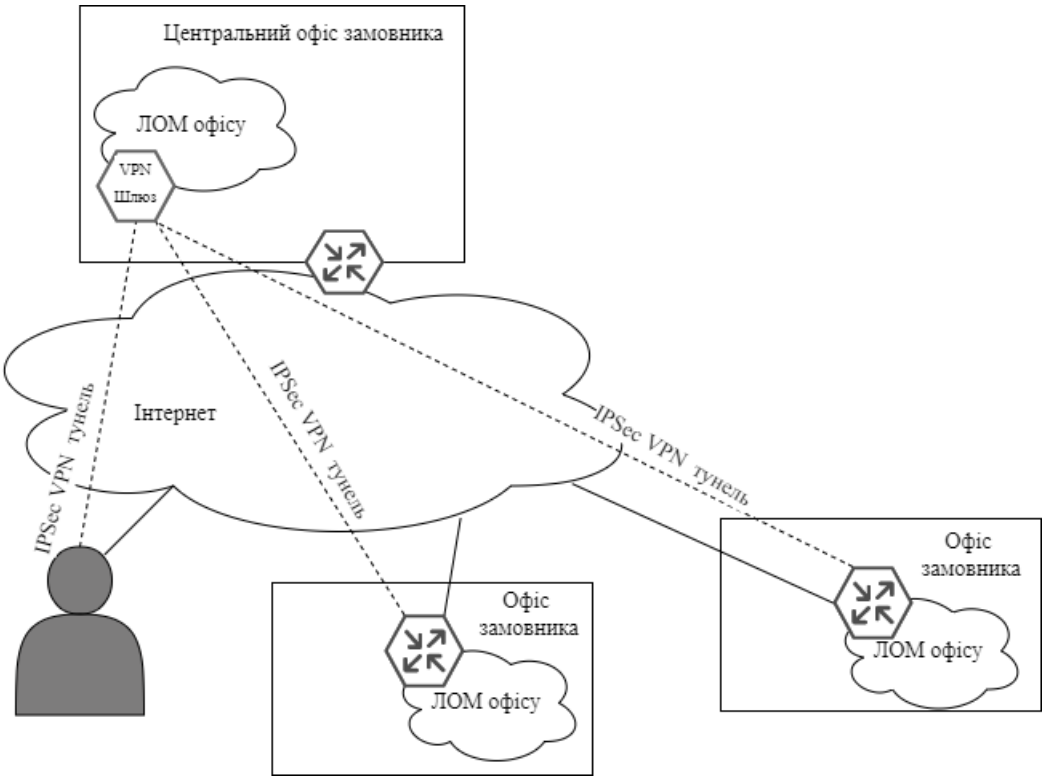

Рис. 1.4. Принцип роботи в режимі тунелю

Протоколи захищеного каналу, як правило, використовують в своїй роботі механізм тунелювання. За допомогою даної методики пакети даних транслюються через загальнодоступну мережу як по звичайному двоточковому з’єднанню. Між кожною парою «відправник–отримувач даних» встановлюється своєрідний тунель – безпечне логічне з’єднання, яке дозволяє інкапсулювати дані одного протоколу в пакети іншого.

Сукупність правил створення тунелів, яка називається «політикою безпеки», записується в налаштуваннях VPN-агентів. IP-пакети направляються в той чи інший тунель або відкидаються після того, як будуть перевірені:

IP-адреса джерела (для вихідного пакету - адреса конкретного комп'ютера мережі, що захищається);

IP-адреса призначення;

протокол більш високого рівня, якому належить даний пакет (наприклад, TCP або UDP);

номер порту, з якого або на який відправлена інформація.

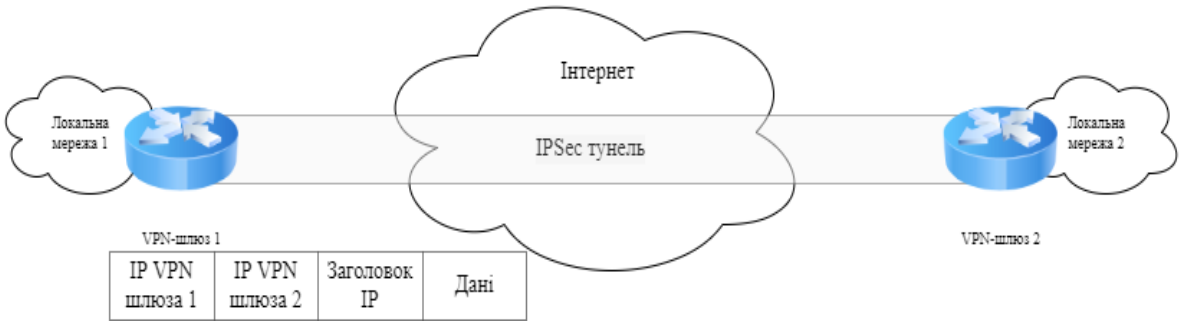

Рис.1.5. Приклад тунелю

Процес тунелювання (або інкапсуляції) зводиться до того, що пакет протоколу нижчого рівня поміщається в поле даних пакета протоколу такого ж або більш високого рівня.

Цей механізм використовується для безпеки передачі даних через публічні мережі шляхом упаковки пакетів під зовнішню оболонку. Тунель створюється між двома граничними пристроями, які розміщуються в точках входу в мережу.

Технологія тунелювання дозволяє зашифрувати вихідний пакет повністю, разом із заголовком, а не тільки його поле даних. Такий зашифрований пакет поміщається в інший пакет з відкритим заголовком. Цей заголовок використовують для транспортування даних на ділянці загальної мережі. У граничній точці захищеного каналу витягується зашифрований заголовок, який буде використовуватися для подальшої передачі пакета.

Як правило, тунель створюється тільки на ділянці мережі загального користування, де існує загроза порушення конфіденційності і цілісності даних.

Крім захисту переданої інформації механізм тунелювання використовують для забезпечення цілісності та автентичності. При цьому, захист потоку реалізується більш повно. Тунелювання застосовується також і для узгодження різних транспортних технологій, якщо дані одного протоколу транспортного рівня необхідно передати через транзитну мережу з іншим транспортним протоколом.

Для формування тунелів VPN використовують протоколи PPTP, L2TP, IPsec, IP-IP. Протокол PPTP дозволяє інкапсулювати IP-, IPX-і NetBEUI-трафік в

заголовки IP для передачі по IP-мережі, наприклад, Internet.

Протокол L2TP дозволяє шифрувати и передавати IP-трафік з використання будь-яких протоколів, що підтримують режим "точка-точка" доставки дейтаграм. Наприклад, до них відносяться протокол IP, ретрансляція кадрів и асинхронний режим передачі (ATM). Протокол IPsec - дозволяє шифрувати та інкапсулювати корисну інформацію протоколу IP в заголовки IP для передачі по IP-мережі.

Таблиця 1.1

Протоколи захищеного каналу в моделі OSI

Модель OSI	Протоколи захищеного каналу
Прикладний	MIME
Представлення	SSL, TLS
Сеансовий	
Транспортний	
Мережевий	IPSec
Канальний	PPTP, L2TP
Фізичний	

1.6 Топологія побудування VPN

Топологія мережі визначає її функціональні можливості, такі як пропускна здатність, швидкість передачі, дальність відправки сигналу. Щоб створити бездротову мережу, здатну вирішувати широкий спектр завдань як у побуті, так і в комерційній та виробничій діяльності, слід визначити, які саме параметри системи є найбільш значущими. Для побудови мереж існує кілька основних принципів, які відрізняються як у сфері застосування, так і за ключовими характеристиками.

1.6.1 Мережа «point-to-point»

Один із типів логічної топології побудови мереж – точка-точка, від

англійської «point-to-point». Цей вид архітектури мережі зумовлює пряме з'єднання між двома вузлами. Пакети безпосередньо передаються від одного вузла до іншого, однак фізично даний відрізок мережі може складатися з великої кількості проміжного обладнання.

Правила передачі – протокол доступу – визначаються логічною структурою і прив'язані до устаткування. Цей тип організації обміну інформацією дозволяє спростити маршрутизацію, ідентифікацію пакетів у структурах з безліччю фізичних комунікаційних вузлів.

Мережа точка-точка може застосовуватися як у випадках кабельного з'єднання різних типів обладнання, так і для бездротового з'єднання двох ПК, двох роутерів, двох локальних мереж. Також застосовується для облаштування віртуальних каналів на основі різноманітних комплексів мережевого обладнання, що забезпечує і бездротовий, і кабельний зв'язок практично на будь-яких відстанях.

Пряме фізичне з'єднання двох пристроїв є на дальності до 100 м, при використанні стандартного споживчого обладнання. Професійні маршрутизатори дають змогу збільшити відстань передачі сигналу до 500 метрів. Також можуть бути використані спеціальні антени:

Секторні;

Параболічні;

Панельні.

Їх застосування збільшує дальність обміну даними між точками зв'язку до 20 кілометрів. Основні види обладнання, що застосовуються для організації мереж топології:

Точки доступу;

Адаптери;

Роутери;

Модеми.

Сукупність пристроїв, складених у вигляді ланцюжка передавального та приймаючого обладнання, здатна забезпечити передачу інформації на великі

відстані. Незалежність каналу точка-точка від видів та кількості обладнання, з якого фізично складена мережа, дозволяє не тільки суттєво збільшувати дальність, а й забезпечувати високу швидкість передачі даних.

Щоб організувати канал зв'язку, точка-точка не потрібна висока кваліфікація або глибокі знання. Налаштувати дане підключення можна просто ґрунтовно вивчивши відповідне керівництво, зокрема деякі виробники мережевого обладнання прикладають покрокові інструкції до пристроїв, що продаються. Також нескладно здійснювати процес підтримки працездатності створеного підключення. Простота налаштування вважає однією з основних переваг мережі точка-точка.

Інший плюс – якість зв'язку. Як самостійно створені, так і придбані у постачальника послуг зв'язку віртуальні канали з реалізацією принципу передачі сигналу точка-точка мають високу пропускну здатність, що мінімізує затримки при обміні інформацією, а також знижує вплив перешкод. Канали надійні, безперервно готові до передачі даних, завжди мають вільні ресурси для негайної реакції на запит про термінове відправлення сигналу.

До недоліків цього виду зв'язку варто віднести відносно високу вартість, якщо даним способом необхідно з'єднати кілька різних вузлів зв'язку. Вартість обладнання також відіграє важливу роль. Для створення стійкого перешкодозахищеного з'єднання потрібна закупівля та розміщення якісного, потужного обладнання, здатного забезпечити безперебійну роботу системи за будь-яких зовнішніх умов.

Низька гнучкість мережі: пропускна здатність суворо лімітована і, при необхідності, будь-які зміни параметрів каналу вимагатимуть заміни всього обладнання.

Не повне використання ресурсів системи. Пропускна здатність є фіксованою, розрахованою на максимальний можливий обсяг передачі даних, а також на вирішення позаштатних ситуацій. У повсякденному використанні всі можливості системи використовуються рідко.

1.6.2 Мережа «point-to-multipoint»

При організації мереж передачі часто необхідно вирішувати завдання з'єднання декількох абонентів з однією точкою доступу, через яку може забезпечуватися обмін інформацією з будь-якої локальної мережею, підключення до корпоративних мереж, зв'язок з оператором зв'язку, доступ до глобальної загальносвітової мережі Інтернет. У разі затребуване застосування типу зв'язку точка-мультиточка – також використовується варіант назви «point-to-multipoint».

Даний вид організації передачі широко застосовується для забезпечення бездротового зв'язку. З'єднання точка-мультиточка необхідно там, де утруднене або нерентабельне прокладання кабельних трас, приймальні пристрої розсіяні по великій площі, також зручно використовувати дану систему для обміну інформацією з мобільними абонентами.

Топологія з єдиним вузлом зв'язку, якого підключаються окремі абоненти носить найменування точка-мультиточка. У кабельних мережах аналогічна топологія називається "зірка". Широко використовуються при побудові розподілених систем зв'язку.

На базі систем точка-багатоточка можна побудувати приватні захищені приватні мережі, з обмеженням доступу. Це особливо важливо для великих виробничих та проектних установ, де відбувається обмін конфіденційною інформацією між різними підрозділами. І тут безпека зв'язку є ключовим елементом локальної мережі організації.

Для створення бездротової мережі точка-багатоточка використовують базові станції оснащені декількома антенами. Дане обладнання встановлює поза будівлями та спорудами, тому застосовують всі погодні виконання пристроїв. Зовнішні антени посилюють сигнал, збільшуючи як відстань передачі, і підвищуючи якість і швидкість обміну даними.

Приймальні пристрої є точки доступу, для подальшого з'єднання з мережею користувальницького обладнання, так і схожі з ними по функціоналу роутери, маршрутизатори, інші пристрої здатні приймати і передавати Wi-Fi сигнал.

Як правило обладнання розраховане на роботу у двох діапазонах: 2,4 ГГц та 5 ГГц. Частота 2,4 ГГц часто буває перенасичена сигналами, які створюють перешкоди та вносять спотворення сигнал, що впливає як на швидкість роботи, так і на дальність передачі. У деяких випадках набагато ефективніше використовувати частоту 5 ГГц. Але далеко не всі користувальницькі пристрої, такі як смартфони, ноутбуки, здатні працювати на даній частоті.

Тому рекомендується приймати рішення про перехід на іншу, попередньо ознайомившись з параметрами обладнання. Багато виробників випускають пристрої, здатні функціонувати на інших частотах. При необхідності з'єднання кількох об'єктів, що знаходяться на значному віддаленні один від одного, в єдину мережу найбільш ефективним способом є створення мережі з топологією крапка. Дана система відрізняється гнучкістю: можна додавати і видаляти абонентів, а також змінювати ширину каналу доступу, що виділяється кожному з них.

Простота зміни конфігурації мережі – для цього достатньо додати абонента, встановити обмеження на підключення нових абонентів, заборонити певним пристроям доступ до ресурсів мережі. Усі перелічені настройки вносяться програмно, під час роботи з вбудованим ПЗ устаткування. Іншою перевагою є наявність великої кількості опцій, що налаштовуються, що дозволяє максимально ефективно задіяти ресурси обладнання. Великою перевагою є широкий асортимент пристроїв, що пропонуються виробниками. Практично під кожне завдання можна підібрати обладнання, яке найкраще впорається з його вирішенням.

Правильний підбір передавальних пристроїв дозволяє забезпечувати потрібну зону покриття. Професійні моделі дозволяють суттєво збільшити дальність передачі сигналу, а якщо не потрібні особливі параметри мережі, то можна суттєво заощадити, купуючи менш дороге обладнання.

До недоліків системи можна віднести складнощі, пов'язані з використанням деяких алгоритмів здійснення зв'язку базової станції з абонентами. Але у більшості сучасних станцій використовуються сучасні версії ПЗ, в яких враховано та виправлено зазначені недоліки.

1.6.3 Відмінності мереж «point-to-point» та «point-to-multipoint»

Застосування мереж точка-точка і точка-мультиточка доцільно під час вирішення різних завдань. Наприклад, при невеликій кількості кінцевих абонентів може бути дешевше і ефективніше організувати кожному з них з'єднання за принципом точка-точка, якщо абоненти вимогливі до якості зв'язку. Або можна здешевити систему, встановивши одну базову станцію, здатну працювати з усіма абонентами. Очевидно, що при зростанні числа приймаючих пристроїв, а також при зниженні вимог до параметрів зв'язку, краще застосовувати систему точка-багатоточка. Це дозволить знизити витрати і збільшить можливості налаштування мережі, дозволить додати нових абонентів, що важливо, якщо прогнозується зростання їхньої кількості, а заздалегідь точну кількість невідомо.

Однак економічна ефективність системи точка-багатоточка падає зі зростанням вимоги до дальності передачі сигналу. Якщо потрібно організувати обмін даними з віддаленими об'єктами, то мережа точка-точка здатна покривати великі відстані, і навіть перетинати континенти. Тут може використовуватись велика кількість проміжного обладнання. При цьому завдяки можливості реалізації віртуального каналу в мережі точка-точка прокладання маршруту відбуватиметься тільки між певними вузлами.

Пакети даних, що передаються, «бачитимуть» тільки початковий пункт відправки і кінцевий, приймаючий, комунікаційний вузол. За необхідності організації бездротового зв'язку, основним етапом підготовки, що зумовлює успішне виконання роботи, є правильний вибір обладнання. Коли проаналізовано всі фактори і визначено буде використовуватися система з безліччю абонентів та одним передавальним та приймаючим пристроєм на всіх, або для кожного буде створено свій спеціалізований канал, необхідно встановити такі характеристики, як пропускна здатність, бажана дальність передачі, перешкодостійкість, кількість абонентів, що підтримується. Щоб отримати результат, який стане прийнятним рішенням поставленого завдання, слід ретельно вивчити ринок обладнання.

1.7 Аналіз методів та засобів побудови VPN на базі протоколу WireGuard

WireGuard - це захищений та шифрований мережевий тунель що працює на 3 мережевому рівні і реалізований як інтерфейс віртуальної мережі ядра Linux. WireGuard запланований як заміна IPsec для більшості сценаріїв використання, так і як заміна рішень на базі TLS таких як OpenVPN. Водночас VPN WireGuard є більш безпечним, ефективнішим та простим у використанні.

Робота віртуального інтерфейсу тунелю заснована на принципі безпечного механізму асоціації між відкритими ключами однорангової мережі та IP-адреси джерела тунелю. WireGuard використовує єдиний обмін ключами в обидва кінці що заснований на NoiseIK, і прозоро обробляє всі сеанси створення для користувача, використовуючи новий таймер стану механізму машини.

Попередні короткі спільні статичні ключі криптопримітиву Curve25519 використовуються для аутентифікації у стилі OpenSSH. Протокол забезпечує високий рівень таємності пересилання інформації в додаток до високого рівня захищеності від ідентифікації особи що пересилає данні. Транспортна швидкість досягається за допомогою аутентифікованого шифрування ChaCha20Poly1305 для інкапсуляції пакетів у UDP. Покращений підхід до файлів cookie з прив'язкою до IP використовується для пом'якшення атак типу «відмова в обслуговуванні», значно покращуючи механізми cookie IKEv2 и DTLS для додавання шифрування та аутентифікації.

Загальна реалізація протоколу дозволяє не розподіляти ресурси у відповідь на отримані пакети, і з точки зору у системі Linux є багато цікавих методів для реалізації черг та паралелізму. Програмний код WireGuard складається приблизно з 4000 тисяч рядків коду який просто інспектувати на безпеку і відсутність вразливостей програмного продукту.

У операційній системі Linux стандартним рішенням для зашифрованих тунелів є IPsec, який використовує рівень перетворення Linux ("xfrm"). Користувачі заповнюють структуру ядра визначаючи який шифр і ключ, або інші

перетворення, такі яке стиснення використовувати для селектора пакетів що проходять через підсистему. Як правило демон процесу оновлення користувацького простору відповідає за оновлення цих інформаційних структур на основі результату обміну ключами зазвичай виконаного за допомогою IKEv2 , самого складного і гнучкого протоколу. Тому цей протокол і є величезним за об'ємом коду, і дуже складним у своїй нинішній реалізації.

Мережеві інженери та системні адміністратори мають окремий набір семантики для фаєрволу та безпечного маркування пакетів IPsec. Коли відокремлюючи рівень обміну ключами від транспортного шифрування або рівня трансформації то використовують розумне та правильне відокремлення з семантичної точки зору, і подібно до цього розділення рівня перетворення з рівнем інтерфейсу є правильним з точки зору мережі, але цей підхід до суворого і правильного розподілу шарів збільшує складність і робить неможливим правильне розгортання і реалізацію.

WireGuard позбавляється цих розділювальних шарів. Замість складності IPsec та шарів xfrm, WireGuard просто надає віртуальний інтерфейс наприклад wg0 яким потім можна керувати за допомогою стандарту ip (8) та ifconfig (8) утиліти. Після налаштування інтерфейсу за допомогою закритого ключа (і, за бажанням, загальнодоступного симетричного ключа) та різних відкритих ключів кінцевих клієнтів з якими він спілкується, тунель просто надійно працює. Обмін ключами, підключення, відключення, повторне підключення, виявлення далі відбуваються прозоро і надійно, тому адміністратору не потрібно турбуватися про ці деталі.

Правила брандмауера можна налаштувати використовуючи звичайну інфраструктуру для інтерфейсів брандмауера, з гарантією того що пакети даних що надходять з інтерфейсу WireGuard будуть автентифіковані та зашифровані. WireGuard набагато менше схильний до катастрофічних збоїв та неправильної конфігурації, ніж IPsec. Однак важливо підкреслити що відокремленні рівні IPsec є правильним і надійним рішенням для досягнення академічної досконалості протоколу. Але, як це часто буває с правильними рівнем абстракції, важко

досягнути достатнього глибокого рівня комфортності використання і аудиту безпеки реалізації протоколу.

У свою чергу WireGuard заснований на порушенні деяких правил розділення на рівнях реалізації, вирішує проблеми цього розділення для використання практичних інженерних рішень і криптографічних методів, для вирішення реальних проблем.

Для порівняння можна розглянути рішення на базі OpenVPN засноване на інтерфейсі користувача TUN/TAP яке використовує TLS. TUN/TAP інтерфейс не є частиною ядра системи, тому він має дуже низьку продуктивність - оскільки пакети потрібно копіювати кілька разів між рівнем ядра та рівнем користувача а також постійно тримати у роботі демон інтерфейсу. Хоча інтерфейси TUN/TAP (наприклад tun0) мають подібні до wg0 інтерфейсу переваги, OpenVPN також надзвичайно складний і підтримує цілий ряд функціональних можливостей TLS, що дає можливість для використання потенційних вразливостей.

OpenVPN можна правильно застосовувати на користувацькому рівні, але оскільки ASN.1 та x509 парсери в ядрі історично були досить проблематичними (CVE-2008-1673, CVE-2016-2053) то додавання ще і стеку TLS лише погіршить цю проблему. TLS також приносить із собою величезну кінцевий автомат, а також менш чистий між вихідними IP-адресами та відкритими ключами. Для організації процесу доставки ключів, WireGuard бере натхнення з OpenSSH, використовується дуже простий метод і підхід до керування ключами. Через різноманітний набір статичних механізмів два однорангових вузла обмінюються своїми статичними відкритими ключами. Іноді це просто, як електронна пошта з підписом PGP, а інколи – складний механізм розподілу ключів із використанням LDAP та органів сертифікації.

Важливо те, що здебільшого розподіл ключів в OpenSSH повністю незалежне, тому і WireGuard застосовує подібні приклади такого механізму. Два вузли WireGuard обмінюються своїми відкритими ключами через якийсь невизначений механізм, і згодом вони вже можуть передавати данні.

Ставлення WireGuard до розподілу ключів полягає в тому що такий

розподіл ключів не правильним для вирішення цієї конкретної проблеми, і тому інтерфейс спрощується для використання будь-якого рішення для розподілу ключів. Додатковою перевагою є те що відкриті ключі мають лише 32 байти і можуть бути легко представлені в кодуванні Base64 розміром в 44 символи, тобто ця особливість буде корисною передачі ключів через різноманітні носії.

Нарешті, WireGuard має криптографічну надійність. Йому навмисно не вистачає гнучкості шифрування і протоколу. Якщо будуть знайдені вразливості у криптографічному примітиві то буде потрібно оновити всі кінцеві точки мережі. Як показують триваючі знаходження вразливостей SSL/TLS гнучкість шифру у протоколі значно збільшує складність реалізації.

WireGuard використовує варіант шуму Trevor Perrin's, який отримав чимало відгуків від розробників WireGuard на етапі розробки, для обміну ключами 1-RTT із Curve25519 для ECDH, HKDF для розширення результатів ECDH, RFC7539 щоб побудувати ChaCha20 та Poly1305 для автентифікованого шифрування та BLAKE2 для хешування. Протокол WireGuard має вбудований захист від атаки типу «відмова в обслуговувані» використовуючи новий механізм крипто-файлів cookie для атрибуції IP-адреси. Так само

WireGuard працює на 3 рівні, бо цей підхід є самим чистим для забезпечення цілісності та атрибутивності пакетів. Автори вважають що для об'єднання кількох IP-мереж має відбуватися на 3 рівні і накладення на цей рівень протоколу WireGuard дозволяє багато чого спростити. В результаті отримуємо більш чистий і простий в реалізації протокол. Він підтримує 3 рівень як для IPv4, так і для IPv6, та також може інкапсулювати v4-в-v6 або v6-в-v4.

WireGuard поєднує всі ці принципи, зосереджуючись на простоті та перевіреній кодовій базі, залишаючись при цьому надзвичайно швидкісним і придатним для використання у різних рішеннях.

Поєднуючи обмін ключами та шифрування на 3 рівні з використанням механізму віртуального інтерфейсу, а не рівнем перетворення, справді порушує традиційні принципи багаторівневого підходу до написання протоколу, тим самим прагнучи до надійного інженерного рішення яке буде одночасно

практичним та безпечним. Попутно WireGuard використовує кілька нових криптографічних та системних рішень для досягнення своїх цілей.

2 ОСНОВНІ МЕХАНІЗМИ РОБОТИ ПРОТОКОЛУ VPN WIREGUARD НА БАЗІ LINUX

2.1 Таблиці маршрутизації криптоключів

Основоположним принципом безпечного VPN є взаємозв'язок між одноранговими підмережами та IP-адресами кожна з яких дозволена для використовувати як вихідні IP-адреси. У WireGuard однорангові вузли ідентифікуються строго за їх відкритим ключем, 32-байтним Curve25519. Це означає, що існує просте співставлення між відкритими ключами та списком дозволенних IP-адрес.

Таблиця 2.1

Маршрутизація криптоключа

Публічний ключ інтерфейсу	Закритий ключ інтерфейсу	Прослуховування порту UDP
HIgo...8ykw	yAnz...fBmk	41414
Публічний ключ однорангового інтерфейсу	Список дозволених підмереж	
xTIB...p8Dg TrMv...WXX0 gN65...z6EA	10.192.122.3/32, 10.192.124.0/24 10.192.122.4/32, 192.168.0.0/16 10.10.10.230/32	

Сам інтерфейс має закритий ключ та UDP порт який він прослуховує і за яким йде список вузлів. Кожен одноранговий вузол ідентифікується своїм відкритим ключем. У кожного із них є список вихідних дозволених IP-адрес. Коли вихідний пакет передається через інтерфейс WireGuard (wg0), ця таблиця використовується для визначення відкритого ключа бо буде використаний для

шифрування. Наприклад, пакет із мережею призначення 10.192.122.4 буде зашифровано за допомогою захищеного сеансу, отриманого із відкритого ключа TrMv...WXX0. Так і навпаки, коли wg0 отримує шифрований пакет то після його розшифровки і аутентифікації він буде прийнятий у безпечний режим автентифікації тільки в тому випадку якщо його вихідна IP-адреса буде дозволена і співставленна у таблиці.

Наприклад пакет буде дозволеним якщо він розшифрований із xTIB...qp8D та був надісланий з IP-адреси 10.192.122.3 або в діапазоні з 10.192.124.0-10.192.124.255, в інакшому випадку подібний пакет буде видалений. Використовуючи цей простий принцип адміністратори можуть покластися на прості правила брандмауера. Наприклад вхідний пакет на інтерфейсі wg0 з IP-джерелом 10.10.10.230 може розглядатися як автентичний якщо він був розшифрований від вузла з публічним ключем gN65...z6EA. Загалом будь-які пакети надіслані на інтерфейс WireGuard будуть мати надійний та справжню IP-адресу джерела (звісно за умови гарантованої секретності ключа шифрування).

Доцільно звернути увагу що це можливо лише тому що суворо працює на 3 рівні. У відмінності від декотрих розповсюджених протоколів VPN наприклад таких як L2TP/IPsec забезпечує ідентифікацію однорангових вузлів на 3 рівні що забезпечує більш просту логіку мережі. У випадку однорангового вузлу який буде мати змогу пересилати весь трафік на інший вузол WireGuard, таблиця маршрутизації криптоключів може бути налаштована наступним чином:

Таблиця 2.2

Публічний ключ інтерфейсу	Закритий ключ інтерфейсу	Прослуховування порту UDP
gN65...z6EA	gI6E...fWGE	21841
Публічний ключ однорангового інтерфейсу	Список дозволених підмереж	
HIgo...8ykw	0.0.0.0/0	

У цьому випадку вузли авторизовані за допомогою ключа HIgo...8ykw можуть надсилати пакети з будь-якої. Всі пакети що надійшли на інтерфейс wg0 будуть зашифровані та розшифровані і надіслані на кінцевий вузол призначення

2.2 Кінцеві хости та роумінг

Звичайно важливо щоб однорангові вузли могли надсилати зашифровані пакети WireGuard UDP один одному на кінцеві вузли мережі Інтернет. За бажаннями можна попередньо вказати в таблиці маршрутизації криптоключів відому надсилаючого вузла.

Причина необов'язковості полягає в тому що якщо в таблиці не буде вказано кінцеву IP-адресу та UDP-порт, то WireGuard буде отримувати коректно аутентифікований пакет від кінцевого вузла і використовувати зовнішню IP-адресу для визначення кінцевого вузла.

Оскільки відкритий ключ однозначно ідентифікує кінцевий вузол з його IP та UDP портом, це дозволяє одноранговим вузлам вільно переміщуватися між підмережами та зовнішніми IP-адресами як наприклад Mesh.

Таблиця маршрутизації криптоключів може бути доповнена наступними аргументами для визначення кінцевого вузла:

Таблиця 2.3

Публічний ключ інтерфейсу	Закритий ключ інтерфейсу	Прослуховування порту UDP
gN65...z6EA	gI6E...fWGE	21841
Публічний ключ однорангового інтерфейсу	Список дозволених підмереж	Кінцевий IP та UDP порт
HIgo...8ykw	0.0.0.0/0	192.95.5.69:41414

Хост gN65...z6EA надсилає зашифрований пакет до HIgo...f8yk на 192.95.5.69:41414. Після HIgo...8ykw отримує пакет і він оновлює свою таблицю

маршрутизації ключа щоб дізнатися, що кінцевою точкою для відправки пакетів відповідей є наприклад 192.95.5.64:21841

Таблиця 2.4

Публічний ключ інтерфейсу	Закритий ключ інтерфейсу	Прослуховування порту UDP
HIgo...8ykw	yAnz...fBmk	41414
Публічний ключ однорангового інтерфейсу	Список дозволених підмереж	Кінцевий IP та UDP порт
xTIB...p8Dg TrMv...WXX0 gN65...z6EA	10.192.122.3/32, 10.192.124.0/24 10.192.122.4/32, 192.168.0.0/16 10.10.10.230/32	192.95.5.64:21841

Легкості використання та простоти у протокол додає те що порт прослуховування вузлів та вихідний порт надісланих пакетів завжди однаковий, ця особливість також забезпечує просте проходження пакетів за NAT.

Оскільки ця властивість маршрутизації гарантує що вузли матимуть найновіший IP та UDP порт зовнішнього джерел то для NAT не потрібно тримати постійне збереження з’єднання та сеансу (для випадків використання, у яких надзвичайно важливо нескінченно тримати відкритим сеанс NAT або брандмауер із встановленим статусом, інтерфейс можна додатково налаштувати для періодичної надсилання постійних аутентифікованих пакетів keepalive).

2.3 Потік, відправлення та отримання даних

Структура роумінгу об’єднана з маршрутизацією криптоключів в складає наступні потоки даних при отримані та передаванні пакету на інтерфейс за допомогою «Конфігурації 1» зверху. Пакет генерується локально (або пересилається) і готовий до передачі на вихідному інтерфейсі wg0:

1. Пакет нешифрованих даних досягає інтерфейсу WireGuard wg0;

2. Перевіряється IP-адреса призначення пакету 192.168.87.21, яка відповідає ключу вузла TrMv...WXX0. (Якщо пакет не відповідає жодному ключу , його відкидають і відправник повідомляється стандартним ICMP повідомленням «немає маршруту до хосту»);

3. Симетричний ключ шифрування та генератор випадкового числа асоціюють безпечну сесію з вузлом TrMv...WXX0 та шифрують пакет за допомогою ChaCha20Poly1305;

4. Заголовок, що містить додаткові поля описані в розділі 5.4, додається до шифрованого пакету;

5. Цей заголовок та зашифрований пакет разом надсилаються у вигляді пакету UDP до кінцевої точки Інтернету асоційованого з вузлом TrMv ...WXX0, в результаті чого утворюється зовнішній пакет що містить у собі заголовок та корисну шифровану інформацію. Якщо кінцева точка не була попередньо сконфігурована або не отримала у полі заголовку коректної IP з останнього отриманого аутентифікованого пакета то вхідні данні будуть відброшені та видалені.

IP/UDP Пакет досягає UDP-порту 41414 хоста, який є слухаючим портом UDP інтерфейсу wg0:

1. Пакет IP/UDP що містить певний заголовок та зашифроване корисне навантаження, приймається на правильний порт(у цьому конкретному випадку порт 41414);

2. Заголовок WireGuard визначає, що вона пов’язана з безпечним сеансом вузла TrMv...WXX0, перевіряє дійсність лічильника повідомлень та намагається аутентифікуватись і розшифрувати його використовуючи симетричний ключ отриманий під час захищеного сеансу. Якщо він не може визначити вузол або якщо автентифікація не вдається, пакет скидається і видаляється;

3. Коли пакет аутентифікувався правильно, вихідний IP зовнішнього пакета UDP використовується для оновлення IP кінцевої точка для вузла TrMv... WXX0;

4. Як тільки розшифровується корисне навантаження пакета, інтерфейс має відкритий текст пакету. Якщо це не IP-пакет, його буде видалено.

В іншому випадку WireGuard перевіряє, чи відповідає вихідна IP-адреса внутрішнього пакету маршрутизації криптоключів. Наприклад, якщо вихідний IP розшифрованого пакету - 192.168.31.28, то такий пакет даних приймається. Але якщо IP джерела - 10.192.122.3 то пакет буде відкинутим і видаленим. Якщо пакет даних не було скинуто, він вставляється в чергу прийому інтерфейсу wg0. Можна було б розділити список дозволених IP-адрес на два списки - один для перевірки адреси пакетів, та один для вибору мережі на основі адреси призначення. Але, зберігаючи їх як частину того ж списку, це дозволяє реалізувати щось подібне до фільтрації із зворотним шляхом.

Під час надсилання пакетів порівнюються IP-адреси призначення у таблиці маршрутизації, а при отриманні пакету у таблиці визначається чи дозволено отримання пакетів з цієї IP-адреси. Однак замість того щоб хост порівнював чи IP джерело отриманого пакету співпадає зі списком дозволених IP-адрес, замість цього хост може задати більш глобальне запитання - до якого хоста буде обрано таблицю маршрутизації для цього джерела IP. Це забезпечує індивідуальний зв'язок для надсилання та отримання IP-адрес, якщо пакет отриманий від певного сервера, то відповідь гарантовано буде надана тому самому хосту.

2.4 Загальна конфігурація та використання протоколу

Перш ніж заглиблюватися в деталі криптографії та реалізації протоколу, буде корисно переглянути простий командно-лінійний інтерфейс використання WireGuard, для надання конкретики представленим на сьогодні концепціям. Розглянемо середовище Linux з єдиним фізичним мережевим інтерфейсом eth0, що підключає його до Інтернету з загальнодоступною IP-адресою 192.95.5.69. Інтерфейс WireGuard, , може бути доданий і налаштований на тунель з IP-адресою 10.192.122.3 у підмережі з маскою / 24 за допомогою стандартних утиліт ip (8), показаних нижче.

Таблиця 2.5

Додавання нового інтерфейсу

$ ip link add dev wg0 type wireguard
$ ip address add dev wg0 10.192.122.3/24
$ ip route add 10.0.0.0/8 dev wg0
$ ip address show
1: lo: <LOOPBACK> mtu 65536
inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST> mtu 1500
inet 192.95.5.69/24 scope global eth0
3: wg0: <POINTOPOINT,NOARP> mtu 1420
inet 10.192.122.3/24 scope global wg0

Конфігурування таблиці маршрутизації криптоключу wg0

$ wg setconf wg0 configuration-1.conf
$ wg show wg0
interface: wg0
public key: HIgo...8ykw
private key: yAnz...fBmk
listening port: 41414
peer: xTIB...p8Dg
allowed ips: 10.192.124.0/24, 10.192.122.3/32
peer: TrMv...WXX0
allowed ips: 192.168.0.0/16, 10.192.122.4/32
peer: gN65...z6EA
allowed ips: 10.10.10.230/32
endpoint: 192.95.5.70:54421
$ ip link set wg0 up

Продовження таблиці 2.5

Конфігурування таблиці маршрутизації криптоключу wg0
$ ping 10.10.10.230 PING 10.10.10.230 56(84) bytes of data. 64 bytes: icmp_seq=1 ttl=49 time=0.01 ms

У цьому прикладі пакет надісланий до 10.10.10.230 буде направлений через інтерфейс wg0, який зашифрує пакет за допомогою захищеного сеансу, пов'язаного з відкритим ключем gN65 ... z6EA, і надішле зашифрований та інкапсульований пакет до 192.95.5.70:54421. При отриманні пакета від 10.10.10.230 на wg0 інтерфейс адміністратор може бути впевнений що цей пакет автентифіковано з gN65...z6EA.

2.5 Протокол та криптографія

Як вже згадувалося раніше, для того щоб розпочати відправлення зашифрованих інкапсульованих пакетів, спершу має відбутися обмін ключами (рукостискання) 1-RTT. Ініціатор надсилає повідомлення відповідачеві, а той відповідає ініціатору.

Як вже згадувалося раніше, для того щоб розпочати відправлення зашифрованих інкапсульованих пакетів, спершу має відбутися обмін ключами (рукостискання) 1-RTT. Ініціатор надсилає повідомлення відповідачеві, а той відповідає ініціатору.

Після цього рукостискання ініціатор може надсилати зашифровані повідомлення, використовуючи спільну пару симетричних ключів, один ключ для надсилання та один ключ для отримання відповіде. Після проходження першого успішного шифрованого повідомлення від ініціатора до відповідача, відповідач може почати надсилати зашифровані повідомлення ініціатору.

Цей процес обміну ключами вимагає підтвердження описаних в KEA + C, що також дозволяє повідомленням рукостискання оброблятися асинхронно для

транспортування даних цього типу. Ці повідомлення використовують шаблон “IK” від Noise яке пом’якшує атаки типу. Кінцевим результатом протоколу є дуже надійна система безпеки, яка відповідає вимогам безпеки автентифікованого обміну ключами (AKE) , попереджує компрометацію ключів, забезпечує ідентичність приховування статичних відкритих ключів, подібних до SIGMA і чинить опір атакам типу «відмова в обслуговуванні».

2.5.1 Ігнорування протоколом неавтентифікованих пакетів

Як вже згадувалося раніше, для того щоб розпочати відправлення зашифрованих інкапсульованих пакетів, спершу має відбутися обмін ключами (рукостискання) 1-RTT. Ініціатор надсилає повідомлення відповідачеві, а той відповідає ініціатору. пакетів тоді і не генерується відповідь, WireGuard невидимий для чужих вузлів та мережевих сканерів. Декілька класів атак можна уникнути не дозволяючи неавтентифікованим пакетам впливати на будь-який стан. І в цілому це можливо реалізувати у WireGuard таким чином щоб взагалі не використовувати динамічний розподіл пам'яті навіть для автентифікованих пакетів.

Однак для цієї властивості потрібно перше повідомлення, отримане приймачем для автентифікація ініціатора. Наявність автентифікації в першому пакеті, подібному до цього, потенційно відкриває сервер відповідач до повторної атаки. Зловмисник може відтворити початкові повідомлення про рукостискання, щоб вимусити сервер на відновлення його ефемерного ключа, тим самим анулюючи сесію законного ініціатора (хоча це не впливає на таємність або автентифікованість будь-яких повідомлень).

Щоб запобігти цьому у 12-байтовому TAI64N першого повідомлення включена зашифрована та автентифікована мітка часу. Сервер відслідковує найбільшу мітку часу отриману на одного партнера та відкидає пакети що містять мітки часу менші або рівні їй (насправді це навіть не повинно бути точною відміткою часу, це просто має бути монотонно зростаюче 96-розрядне число для

однорангової мережі). Якщо сервер перезапуститься і втратить цей стан, то це не є проблемою, навіть незважаючи на те що початковий пакет попереднього періоду може бути відтворений, він не може порушити будь-які поточні безпечні сеанси, оскільки сервер щойно перезапустився, і не має неактивних безпечних сеансів для зриву.

Після того як ініціатор відновить безпечний сеанс з сервером після його перезапуску, ініціатор буде використовувати більшу мітку часу, анулюючи попередню. Ця мітка часу виключає можливість зловмиснику порушити поточний сеанс між ініціатором та респондентом за допомогою повторної атаки). З точки зору реалізації, TAI64N дуже зручним, оскільки він є тупокінцевим (big-endian), що дозволяє порівнювати дві 12-байтових позначок часу, які слід робити за допомогою стандартного memcmp ().

Оскільки WireGuard не використовує підписи, то перше надсилання повідомлення покладається лише на відповідь протоколу Діффі-Хеллмана при обміні статичними ключами для автентифікації. Це означає що якщо будь-який із їхніх статичних ключів буде скомпроментований, то зловмисник зможе підробити повідомлення про ініціацію. Така дія не зможе завершити повне рукостискання, бо воно містить максимум значення мітки часу, тим самим запобігаючи успіху всіх майбутніх з'єднань.

Це може здатися схожим на традиційні вразливості при видаванні себе за іншу особу до яких WireGuard не є вразливим, але насправді це зовсім інше. Бо якщо компроміс з ключами дозволяє зловмисникові перешкодити вузлам коли-небудь використовувати їх компрометовані ключі ще раз, то таким чином зловмисник фактично допоміг належній реакції на такий компроміс. Точність TAI64N позначки часу створює непридатні умови для витоку інформації.

2.5.2 Додатковий режим симетричного ключа з попереднім загальним доступом

WireGuard покладається на те що вузли апріорі обмінюються між собою

статичними відкритими ключами як своїми статичними ідентифікаторами. Конфіденційність усіх надісланих даних залежить від безпеки функції Curve25519 ECDH. З метою пом'якшення будь-яких майбутніх досягнень у квантових обчисленнях, WireGuard також підтримує режим в якому будь-яка пара вузлів може додатково попередньо обмінятися одним 256-розрядним симетричним ключем шифрування між собою, щоб додати додатковий шар симетричного шифрування.

Модель атаки тут полягає в тому, що зловмисники можуть тривалий час реєструвати і записувати зашифрований трафік в надії колись вдасться зламати і розшифрувати минулий трафік. Попередній обмін симетричними ключами шифрування як правило дуже клопіткий з точки зору управляння ключами, тому ключі можуть більш вірогідно скомпрометовані. Ідея полягає в тому що до того часу коли квантові обчислення скомпрометують Curve25519, цей попередньо розділений симетричний ключ буде виведений з користування.

Що більш важливо, у коротший термін, якщо симетричний ключ був скомпрометований то Curve25519 буде забезпечувати більш ніж достатній захист. Замість використання повністю пост-квантової криптосистеми, яка на момент написання не є практичною для використання тут, цей необов'язковий гібридний підхід до обміну попереднім спільним симетричним ключем для доповнення криптографії еліптичної кривої забезпечує надійність і прийнятний компроміс для параноїдального захисту. Крім того, це дозволяє будувати поверх WireGuard складні схеми ротації ключів, щоб досягти різних типів посткомпромісної безпеки.

2.5.3 Відмова в обслуговувані та файли cookie

Обчислення точок множення Curve25519 вимагає значних ресурсів процесору, навіть якщо Curve25519 є надзвичайно швидкою кривою для більшості процесорів. Для того щоб визначити справжність повідомлення рукостискання треба обчислити криву Curve25519, що означає існування

потенційного шляху для атаки типу «відмова в обслуговуванні».

Для того щоб запобігти такому навантаженню на процесор сервер може прийняти рішення не обробляти повідомлення рукостискання (або ініціативне, або відповідне повідомлення про рукостискання), але замість цього відповісти повідомленням що містить файл cookie. Потім ініціатор використовує цей файл cookie для повторного надсилання повідомлення і це повідомлення має бути прийнято сервером наступного разу. Сервер підтримує секретне випадкове значення, яке змінюється кожні дві хвилини.

Файл cookie - це результат обчислення MAC вихідної IP-адреси ініціатора з використанням MAC адреси як секретного ключа. Ініціатор повторно надсилаючи своє повідомлення, надсилає MAC свого повідомлення використовуючи цей файл cookie в якості ключа MAC. Коли відповідач отримує повідомлення у час знаходження під навантаженням, він може приймати рішення про надання та обробку відповіді спираючись на те що чи є правильним MAC, який використовує файл cookie як ключ.

Цей механізм пов’язує повідомлення надіслані від ініціатора на власність йому IP-адреси, що дозволяє обмежити швидкість використання класичних алгоритмів обмеження швидкості IP протоколу. Ця схема більш-менш використовується в DTLS та IKEv2. Однак цей метод страждає від трьох основних вад. По-перше, ми вважаємо за краще мовчати, не надсилаючи жодної відповіді на неавтентифіковані повідомлення: без обробки надсилання повідомлення-відповіді на файли cookie при навантаженні не сервер порушує цю властивість.

По-друге, cookie не слід надсилати в чистому тексті, оскільки людина посеред може використовувати це, щоб потім надсилати шахрайські повідомлення які будуть обробляються. І по-третє, на самого ініціатора може бути спрямована атака «відмова в обслуговуванні», якщо йому направлять шахрайскі файли cookie, які він потім буде використовувати без успіху при обчисленні MAC свого повідомлення Механізм cookie WireGuard, який використовує два MAC (msg.mac1 та msg.mac2), вирішує ці проблеми обчислення.

Для першої проблеми, щоб респондент мовчав, навіть перебуваючи під

навантаженням, усі повідомлення мають перший MAC (msg.mac1), який використовує відкритий ключ відповідача. Це означає, що принаймні вузол який надсилає повідомлення повинен знати з ким він спілкується (завдяки відомості його відкритого ключа), щоб викликати будь-який тип відповіді. Під навантаженням чи не під навантаженням, цей перший MAC (msg.mac1) завжди повинен бути присутнім і дійсним.

Хоча відкритий ключ самого сервера не є секретним, він є достатньо конфіденційним у цій моделі атаки, метою якої є забезпечення безпечності послуг, і тому знання відкритого ключа відповідача є достатнім доказом про його існування (варто зазначити, що цей перший MAC дозволяє пасивно слухаючому зловмиснику робити припущення про те, для якого відкритого ключа призначений пакет, все ж трохи послаблюючи властивості приховування ідентичності, хоча правильність здогадки не буде криптографічним доказом, оскільки при створенні файлу не використовувався приватний MAC).

Так само, щоб вирішити другу проблему - проблему надсилання MAC-адрес у чистому тексті - ми застосовуємо AEAD з розширеним рандомізованим файлом cookie що передається з використанням як симетричний ключа відповідача як відкритого ключа шифрування. Знову ж таки, публічних цінностей тут достатньо для наших цілей в рамках захисту від моделі атаки типу «відмова в обслуговуванні»

Нарешті, для вирішення третьої проблеми ми використовуємо поле «додаткові дані» AEAD для шифрування файлу cookie в транзиті для додаткової автентифікації першого MAC (msg.mac1) початкового повідомлення, яке спровокувало файли cookie відповідь. Це гарантує, що зловмисник, який не знаходиться посередині, не може надсилати потік недійсних файлів cookie-відповіді ініціаторам з’єднання, щоб запобігти їх автентифікації за допомогою правильного файлу cookie (зловмисник що знаходиться посередині в будь-якому випадку може просто скинути повідомлення-відповіді на файли cookie, щоб запобігти з'єднанню, хоча зловмисник із просто пасивною позицією "посередині" справді міг би підробити пакети, що суттєво не відрізняється від атаки відмови в

обслуговуванні проти TCP.) Іншими словами ми використовуємо поле AD для прив’язки відповідей cookie до повідомлень про ініціацію.

Після вирішення цих проблем ми можемо додати вищезгаданий другий MAC (msg.mac2), використовуючи надійно переданий файл cookie як ключ MAC. Коли респондент знаходиться під навантаженням, він приймає лише повідомлення які додатково мають цей другий MAC.

Підсумовуючи, відповідач, після обчислення цих MAC, а також порівняння їх з отриманими в повідомленнях, завжди повинен відхиляти повідомлення з недійсним msg.mac1, а при завантаженні може відхиляти повідомлення з недійсним msg.mac2. Якщо сервер отримує повідомлення з дійсним msg.mac1, але з недійсним msg.mac2, і перебуває під навантаженням, він може відповісти повідомленням-відповіддю на файли. Це значно покращує схему файлів cookie, що використовується DTLS та IKEv2.

На відміну від HIPv2, який вирішує цю проблему за допомогою обміну ключами 2-RTT та складних головоломок, WireGuard уникає конструкцій, що вирішують головоломки, оскільки перша вимагає збереження стану, тоді як друга робить відносини між ініціатором та відповідачем асиметричними. У WireGuard будь-хто з них може в будь-який розпочати рукостискання.

2.6 Типи повідомлень у WireGuard

Існує чотири типи повідомлень, кожне з яких має однобайтовий префікс ідентифікатор типу повідомлення, позначений як msg.type:

повідомлення про ініціювання рукостискання, яке починає процес рукостискання для встановлення захищеної сесії;

відповідь рукостискання на повідомлення ініціації, яке завершує рукостискання, після чого-може бути встановлений безпечний сеанс;

відповідь або на повідомлення про ініціювання рукостискання, або на відповідь на рукостискання яке передає зашифроване значення файлу cookie для повторної відправки відхиленого повідомлення про ініціювання рукостискання

або повідомлення про відповідь на рукостискання;

інкапсульований та зашифрований IP-пакет, який використовує захищений сеанс, узгоджений рукостисканням.

Ініціатор рукостискання позначається індексом i, а відповідач рукостискання позначається індексом r, і будь-який з них позначається як індекс ∗. Для повідомлень які створив ініціатор сесії нехай повідомлення будуть позначатися як (m, m) = (i, r), а повідомлення що були створені відповідачем будуть позначатися як (m, m) = (r, i). Два хости мають декілька змінних, які вони підтримують локально:

I_* - 32 розрядний індекс, який локально представляє інший хост, аналогічно “SPI” IPsec.

S_*^{priv}, S_*^{pub} - статичні значення приватного та відкритого ключів

E_*^{priv}, E_*^{pub} - ефемерні значення приватного та публічного ключів.

Q - необов’язкове значення загальнодоступного симетричного ключа Коли режим попереднього спільного використання ключа не використовується, для цього встановлено значення 0

H_*, C - Значення хеш-результату та значення ключа зв’язування

T_*^{send}, T_*^{recv} - транспортні дані симетричні ключові значення для надсилання та отримання.

N_*^{send}, N_*^{recv} - транспортні дані nonce лічильників повідомлень для відправки та прийому.

У наступних конструкціях використовується кілька символів, функцій та операторів. Бінарний оператор представляє конкатенацію своїх операндів, а двійковий оператор := представляє призначення правого операнда до його лівого операнда. Анотація $\hat{n}$ повертає значення (n+16), яка є аутентифікацією Poly1305 мітки доданої до довжини n . є являє собою порожній бітовий ланцюжок нульової довжини, 0_n являє собою ланцюг бітів повністю (0x0) довжини n байт, а ρ_n являє собою випадковий бітовий рядок довжиною n байт. Нехай τ вважається тимчасовою змінною і нехай к вважається тимчасовим ключем шифрування. Усі

цілочисельні присвоєння є прямим порядком байтів, якщо не зазначено інше.

Використовуються такі функції та константи:

DH (private key, public key) - крива Curve25519 для множення точок приватного ключа та відкритого ключа, повторно повертає 32 байти виводу;

DH-Generate() - створює випадковий закритий ключ Curve25519 і отримує відповідний відкритий ключ, повертаючи пару з 32 байтових ключів;

Aead(ключ, лічильник, звичайний текст, автентифікований текст) - ChaCha20Poly1305 AEAD, як зазначено у RFC7539,при цьому його випадковий ключ (nonce) складається з 32 бітів нулів, за яким слідує 64-розрядне значення лічильника;

Xaead(ключ, лічильник, звичайний текст, автентифікований текст) - XChaCha20Poly1305 AEAD, з 24-байтним випадковим ключем (nonce), створеним за допомогою HChaCha20 та ChaCha20Poly1305;

Hash(input) - Blake2s (з вхідними 32 байтами), повертає 32 байти результату;

Mac(key, input) - Keyed-Blake2s (ключ, вхід, 16), ключовий MAC-варіант хешу BLAKE2s функція, повертаючи 16 байт виводу;

Hmac(key, input) - Hmac-Blake2s (32 бітний ключ введення), звичайна хеш-функція BLAKE2s, яка використовується в конструкції HMAC, повертаючи 32 байти виводу.;

Kdf_n(key, input) - встановлює τ_0 := Hmac(key, input), τ_1 := Hmac(τ_0, 0x1), τ_i := Hmac(τ_0, $\tau_{i-1} \| i$), та повертає n-кратних 32 байтових значень. (τ_1 , ..., τ_n). Це HKDF функція;

Timestamp() - повертає позначку часу TAI64N поточного часу, тобто 12 байт виводу, перші 8 байт - це ціле число великого кінця числа секунд з 1970 TAI, а останні 4 байти велике ціле число числа наносекунд з початку цієї секунди;

Construction - строковий літерал UTF-8 “Noise_IKpsk2_25519_ChaChaPoly_BLAKE2s”, 37 байт виводу;

Identifier - строковий літерал UTF-8 «WireGuard v1», 34 байти виводу;

Label-Mac1 - Строковий літерал UTF-8 “mac1 ----”, 8 байт виводу;

Label-Cookie - Строковий літерал UTF-8 "cookie--", 8 байт виводу.

2.6.1 Загальний огляд механізмів протоколу WireGuard

У більшості випадків рукостискання завершується в 1-RTT, після чого надходять транспортні дані. Якщо один з хостів завантажений, тоді відповідь повідомлення cookie додається до рукостискання, щоб запобігти атаці типу «відмова в обслуговувані»

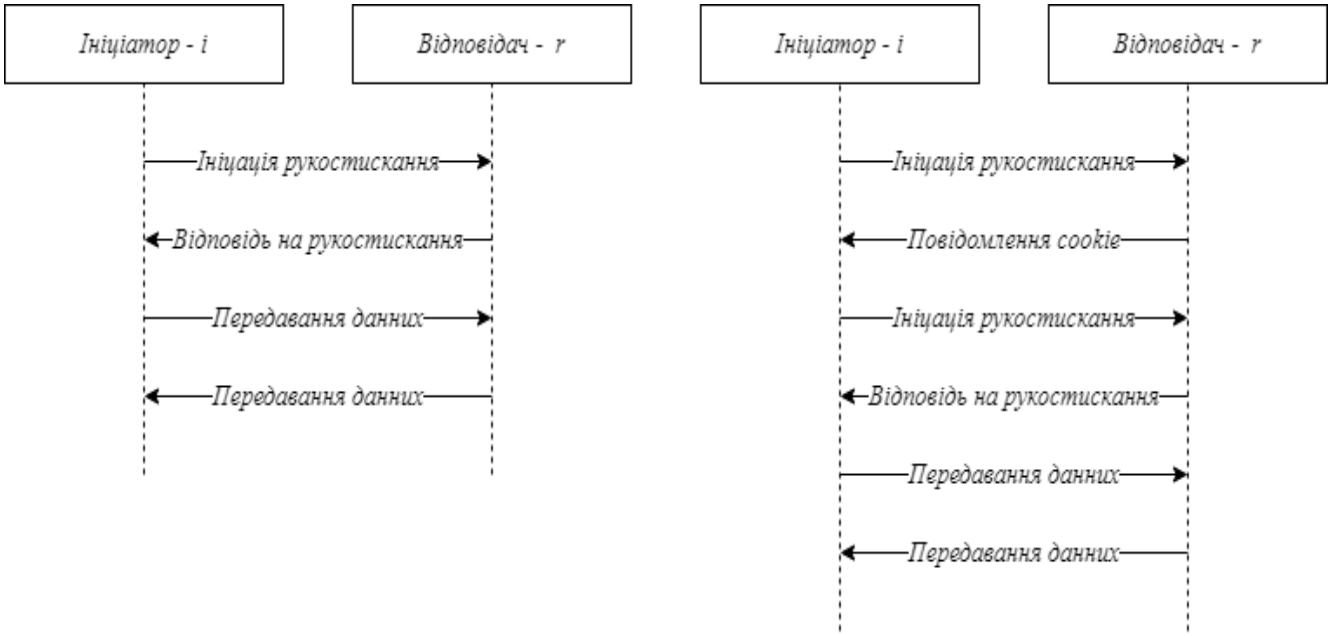

Рис. 2.1. Запобігання атаці типу «відмова в обслуговувані»

2.6.2 Перше повідомлення від ініціатора до відповідача

Таблиця 2.6

type := 0x1 (1 byte)	reserved := 0^3(3 bytes)
sender := I_i (4 bytes)	
ephemeral (32 bytes)	
static ($\widehat{32}$ bytes)	
timestamp ($\widehat{12}$ bytes)	
mac1 (16 bytes)	mac2 (16 bytes)

Ініціатор надсилає повідомлення, msg. Коли відправляється повідомлення I_i воно генерується випадковим чином (ρ^4), і використовується для прив'язки наступних відповідей до сеансу розпочатого у відповідь на це повідомлення. Вирахувані вище поля обчислюються наступним чином(2.1):

$$
\begin{aligned}
&C_i = Hash(Construction)\\
&H_i = Hash(C_i \,\mathrm{P}\, Identifer)\\
&H_i = Hash(H_i \,\mathrm{P}\, S_r^{pub})\\
&(E_i^{priv}, E_i^{pub}) := DH - Generate()\\
&C_i = K_{DF_1}(C_i, E_i^{pub})\\
&msg.ephemeral := E_i^{pub}\\
&H_i = Hash(H_i \,\mathrm{P}\, msg.ephemeral)\\
&(C_i, k) := K_{DF_2}(C_i, DH(E_i^{priv}, S_r^{pub}))\\
&msg.static := A_{EAD}(k, 0, S_i^{pub}, H_i)\\
&H_i = Hash(H_i \,\mathrm{P}\, msg.static)\\
&(C_i, k) := K_{DF_2}(C_i, DH(S_i^{priv}, S_r^{pub}))\\
&msg.timestamp := A_{EAD}(k, 0, Timestamp(), H_i)\\
&H_i := Hash(H_i \,\mathrm{P}\, msg.timestamp)
\end{aligned}
\qquad (2.1)
$$

Коли сервер отримує це повідомлення, він виконує ті самі дії, так що його кінцеві змінні стану є однаковими, замінюючи операнди функції DH для отримання еквівалентних значень.

2.6.3 Перше повідомлення від відповідача до ініціатора

Відповідач надсилає це повідомлення після обробки першого повідомлення від ініціатора та застосовує до повідомлення однакові операції для досягнення ідентичного стану. I_r генерується випадковим чином (ρ^4), коли це повідомлення відправляється, і використовується для прив’язки наступних відповідей до сесії розпочатої цим повідомленням.

Таблиця 2.7

Відповідач надсилає це повідомлення, msg

type := 0x2 (1 byte)	reserved := 0^3(3 bytes)
sender := I_r (4 bytes)	receiver := I_i (4 bytes)
ephemeral (32 bytes)	
empty ($\hat{0}$ bytes)	
mac1 (16 bytes)	mac2 (16 bytes)

Вирахувані вище поля обчислюються наступним чином(2.2):

$$
\begin{aligned}
&(E_r^{priv}, E_r^{pub}) := DH - Generate()\\
&C_r := K_{DF_1}(C_r, E_r^{pub})\\
&msg.ephemeral1 := E_r^{pub}\\
&H_r := Hash(H_r \text{ P} msg.ephemeral)\\
&C_r := K_{DF_1}(C_r, DH(E_r^{priv}, E_r^{pub}))\\
&C_r := K_{DF_1}(C_r, DH(E_r^{priv}, S_i^{pub}))\\
&(C_r, \tau, k) := K_{DF_3}(C_r, Q)\\
&H_r := Hash(H_r \text{ P}\tau)\\
&msg.empty := A_{EAD}(k, 0, \in, H_r)\\
&H_r := Hash(H_r \text{ P} msg.empty)
\end{aligned}
\qquad (2.2)
$$

Коли ініціатор отримує це повідомлення, він виконує ті самі дії, так що його кінцеві змінні стану ідентичні, тобто виконується заміна операндів функції DH для отримання еквівалентних значень. Зверніть увагу, що ця відповідь на рукостискання менше, ніж повідомлення про ініціювання рукостискання, що запобігає мережевим атакам.

2.6.4 MAC адреса файлів cookie

Два повідомлення рукостискання мають параметри msg.mac1 та msg.mac2. Для заданого повідомлення рукостискання, msg_α представляє всі байти msg до

msg.mac1, а msg_{β} - усі байти msg до msg.mac2. Останнє отримане cookie $\widetilde{L_*}$ секунд тому позначається L_*. Msg.mac1 та msg.mac2 поля заповнюються таким чином(2.3):

$$
\begin{aligned}
&msg.mac1 := Mac\ (Hash(Label - Mac1\,\mathrm{P}\,S_{m'}^{pub}), msg_{\alpha})\\
&if\ L_m = \in\ або\ \tilde{L}_m^{\%} \geq 120:\\
&\quad msg.mac2 := 0^{16}\\
&інкше:\\
&\quad msg.mac2 := Mac(L_m, msg_{\beta})
\end{aligned}
\tag{2.3}
$$

2.6.5 Отримання та обчислення ключу транспортних даних

Після обміну вищезазначеними двома повідомленнями обчислюються ключі ініціатором та відповідачем для надсилання та отримання повідомлень транспортних даних(2.4)

$$
\begin{aligned}
&(T_i^{send} = T_r^{recv}, T_i^{recv} = T_r^{send}) := K_{DF_2}(C_i = C_r, \in)\\
&N_i^{send} = N_r^{recv} = N_i^{recv} = N_r^{send} := 0\\
&E_i^{priv} = E_i^{pub} = E_r^{priv} = E_r^{pub} = C_i = C_r := \in
\end{aligned}
\tag{2.4}
$$

В останньому рядку більшість попередніх станів рукостискання видаляться з пам'яті, але значення $H_i = H_r$ не обов'язково дорівнює нулю, оскільки воно може бути корисним у майбутніх переробках шуму.

2.6.6 Наступні повідомлення з транспортними даними

Ініціатор і відповідач обмінюються повідомленнями транспортних даних для обміну зашифрованими інкапсульованими пакетами. Внутрішній пакет відкритого тексту який інкапсулюється представляється в вигляді P, довжини $\|P\|$. Обидва хости надсилають це повідомлення, msg:

Таблиця 2.8

type := 0x4 (1 byte)	reserved := 0^3(3 bytes)
receiver := I_{m}(4 bytes)	
counter (8 bytes)	
packet ($\|\hat{P}\|$ bytes)	

Решта полів заповнюються таким чином(2.5):

$$P := P \text{ P0}^{16\cdot\lceil \text{PPP}/16\rceil - \text{PPP}}$$
$$msg.counter := N_m^{send}$$
$$msg.packet := A_{EAD}(T_m^{send}, N_m^{send}, P, \in)$$
$$N_m^{send} := N_m^{send} + 1 \quad (2.5)$$

Одержувач цих повідомлень використовує T_{m}^{reciv} для читання повідомлення. Зверніть увагу, що в цьому не зберігається значення довжини заголовка, оскільки тег автентифікації служить для визначення законності повідомлення та внутрішня адреса IP-пакету вже міститься заголовку. Сам інкапсульований пакет є заповненим нулями (без змін у поле довжини пакету IP) перед шифруванням, щоб ускладнити аналіз трафіку, хоча це заповнення нулями не має перевищувати максимальний розмір пакету UDP.

Попередній msg.packet містить у собі 16 байт полів заголовка, цу означає що розшифровка може виконуватися на місці, це дозволяє спростити апаратну реалізацію та забезпечити значну продуктивність для багатьох поширених архітектур процесорів. Частково це результат 3 байтового зарезервованого нульового поля, що робить перші чотири байти читабельними як ціле число з прямим порядком байтів.

Значення msg.counter є одноразовим ідентифікатором для ChaCha20Poly1305 AEAD і відстежується отримувачем за допомогою використання N_{m}^{reciv} . Він також функціонує, щоб уникнути атак повтору. Оскільки WireGuard працює через UDP, повідомлення можуть іноді надходити не

по порядку. З цієї причини ми використовуємо вікно для відстеження лічильників отриманих повідомлень, в якому ми відстежуємо найбільший отриманий лічильник, а також вікно попередньо отриманих повідомлень перевірених лише після перевірки тегу автентифікації. Для перевірки використовують алгоритм, описаний у додатку С RFC2401 або від RFC6479.

2.6.7 Відповідь cookie від серверу під навантаженням

Коли надходить повідомлення з дійсним msg.mac1, але msg.mac2 недійсний бо його термін дії закінчився і сервер знаходиться під навантаженням, він може надіслати відповідь на повідомлення cookie. $I_{\acute{m}}$ визначається з поля повідомлення msg.sender, яке ініціювало відповідне повідомлення cookie, msg:

Таблиця 2.9

type := 0x3 (1 byte)	reserved := 03 (3 bytes)
receiver := $I_{\acute{m}}$ (4 bytes)	
nonce := ρ24 (24 bytes)	
cookie ($\widehat{16}$ bytes)	

Секретна змінна R_m змінюється кожні дві хвилини до випадкового значення, A_m являє собою конкатенацію значення зовнішньої адреси джерела IP та порт джерела UDP, а M представляє значення msg.mac1 значення повідомлення на яке це повідомлення відповідає. Останнє поле відповіді на файли cookie заповнюється таким чином(2.6):

$$\tau := Mac(R_m, A_{m'})$$
$$msg.cookie := X_{AEAD}(Hash(Label - Cookie \,\mathrm{P}\, S_m^{pub}), msg.nonce, \tau, M) \quad (2.6)$$

Значення Hash(Label-Cookie ‖ S_m^{pub}) можна попередньо обчислити. Використовуючи M як додаткову автентифікацію поля даних, ми прив'язуємо відповідь файлу cookie до відповідного повідомлення, щоб запобігти атак на хости, шляхом надсилання їм шахрайських відповідей на файли cookie.

Отримавши це дійсне повідомлення, одержувач цього повідомлення, має зберегти cookie разом із часом коли воно було отримано. Механізм буде використовуватися для повторної передачі повідомлень про рукостискання з цими отриманими файлами cookie; подібне повідомлення-відповідь cookie не повинно само по собі спричиняти ретрансляцію.

2.7 Таймери та UX без збереження стану

З точки зору користувача WireGuard не має стану. Налаштований закритий ключ інтерфейсу і відкритий ключ кожного з хостів, тоді користувач може просто нормально відправляти пакети.

Підтримка стану сеансу, ідеальна пряма секретність, і процес рукостискання повністю невидимі для користувача. Хоча подібні автоматичні механізми в минулому були погано пацючими та катастрофічними, WireGuard використовує надзвичайно простий автомат стану таймера в якому кожен перехід до іншого стану чітко визначений, що призводить до надзвичайно повної надійності.

Немає аномальних станів або послідовностей станів, все враховано. WireGuard був успішно протестований на 10 гігабітних інтрамережах, а також каналах з низькою пропускною здатністю і великою затримкою.

Простота автомата стану таймера обумовлена тим фактом, що потрібно лише рукостискання 1-RTT щоб ініціатор та відповідач могли прозоро обмінюватися даними.

2.7.1 Попередні відомості про значення що використовуються

Таблиця 2.10

Наступні константи використовуються для системи стану таймера

Символ	Значення
Rekey-After-Messages	260 повідомлень
Reject-After-Messages	264-213-1 повідомлень
Rekey-After-Time	120 секунд
Reject-After-Time	180 секунд
Rekey-Attempt-Time	90 секунд
Rekey-Timeout	5 секунд
Keepalive-Timeout	10 секунд

За жодних обставин WireGuard не надсилатиме повідомлення про ініціацію більше одного разу для кожного часу зміни ключа. Безпечний сеанс створюється після успішного отримання повідомлення відповіді на рукостискання і термін безпечного сеансу вимірюється з моменту обробки цього повідомлення та безпосередньо наступного виведення ключів транспортних даних. Кожного разу, коли в якості результату надсилається повідомлення про ініціювання рукостискання таймера, що завершується, до закінчення терміну додається додатковий джитер, щоб запобігти великій кількості рукостискань для двох хостів.

2.7.2 Обмеження для транспортних повідомлень

Після першого встановлення захищеного сеансу WireGuard спробує створити новий сеанс, надіславши файл повідомлення про ініціювання рукостискання, після того як він надішле повідомлення передачі даних *Rekey-After-Messages*

Подібним чином, якщо хост є ініціатором поточного захищеного сеансу, WireGuard надішле ініціацію рукостискання повідомлення, щоб розпочати новий захищений сеанс. Якщо після передачі повідомлення транспортних даних поточний захищений сеанс триває *Rekey-After-Time* секунд, або якщо після отримання повідомлення транспортних даних поточний захищений сеанс триває (*Reject-After-Time - Keepalive-Timeout - Rekey-Timeout*) секунд і він ще не діяв після цієї події. Це о повторне використання, що базується на часі, обмежене ініціатором поточної сесії, з метою запобігання проблемі в якій обидва хости можуть спробувати створити нову сесію в один і той же час. Завдяки функції пасивного збереження стану описаній у ініціації старого безпечного сеансу після передачі повідомлення транспортних даних, як правило, повинно бути достатньо для забезпечення нових сеансів створюється кожні *Rekey-After-Time* секунд. Однак для випадку, коли хост отримав дані, але отриманні данні не потребують відповіді на них, і наближається другий термін *Rekey-After-Time* швидше, ніж *Keepalive-Timeout* секунд, тоді ініціація старого захищеного сеансу відбувається під час прийому транспортних даних.

Після передачі повідомлень даних про *Reject-After-Messages* або після закінчення поточного захищеного сеансу *Reject-After-Time* секунд, залежно від того що настане раніше, WireGuard відмовиться надсилати або отримувати будь-які інші транспортні дані, доки новий захищений сеанс не буде створений за допомогою рукостискання.

2.7.3 Обмін ключами шифрування

Нові захищені сеанси створюються приблизно кожні *Rekey-After-Time* секунд (що набагато частіше трапляються до того, як були надіслані повідомлення з транспортними даними *Rekey-After-Messages*). Це означає, що захищений сеанс постійно обнулюється, створюючи новий ефемерний симетричний ключ сеансу для ідеальної прямої секретності. Але майте на увазі, що після ініціювання отримання повідомлення відповіді на рукостискання,

відповідач не може надіслати повідомлення транспортних даних, поки не отримає перше повідомлення з транспортними даними від ініціатора.

Далі транспортні повідомлення, зашифровані за допомогою попереднього захищеного сеансу, можуть передаватися після створення нового захищеного сеансу. З цих причин WireGuard зберігає в пам'яті поточний захищений сеанс, попередні захищений сесії для наступного безпечного сеансу. Якщо після (*Reject-After-Time* × 3) секунд не створюється жоден новий захищений сеанс то поточний захищений сеанс, попередній захищений сеанс, і наступний захищений сеанси відкидаються та обнуляються.

2.7.4 Повторна ініціація рукостискання

Перший раз, коли користувач відправляє пакет через інтерфейс WireGuard, пакет не може бути негайно відправлений, оскільки поточного сеансу не існує. Отже, після черги пакету WireGuard надсилає повідомлення про ініціювання рукостискання.

Після надсилання повідомлення про ініціювання рукостискання, через стан першого пакету або через обмеження умови, якщо повідомлення про відповідь на рукостискання не отримується згодом після *Rekey-Timeout* секунд, будується нове повідомлення про ініціювання рукостискання (з новим випадковим ефемерним ключем) та надсилається. Ця повторна ініціація робиться за Rekey-Attempt-Time секунд до відмови, хоча цей лічильник скидається, коли одноранговий явно намагається надіслати нове повідомлення транспортних даних. Критично важливо майбутньої роботи є коригування значення *Rekey-Timeout* для використання експоненціального відступу замість поточного фіксованого значення.

2.7.5 Пасивне збереження стану

WireGuard реалізує пасивний механізм постійного збереження, щоб це забезпечити активність сеансів і дозволяючи обом хостам пасивно визначати чи

не зникло з’єднання або було роз’єднано. Якщо хост отримав повідомлення про транспортні дані що автентифіковані, але не має жодних пакетів щоб відправити назад протягом часу Keepalive-Timeout, він надсилає повідомлення про підтримку сеансу. Повідомлення про підтримку сеансу транспортних даних із зашифрованим внутрішнім пакетом нульової довжини, є зашифрованими.

Оскільки всі інші транспортних даних містять IP - пакети, які мають мінімальну довжину заголовка (‖IPv4 заголовок‖, ‖заголовок IPv6‖), повідомлення keepalive можна легко розрізнити за наявностю інкапсульованого пакета нульової довжини. Це пасивне збереження надсилається лише тоді, коли хосту нічого надсилати, і надсилається лише за обставин коли інший хост надсилає йому автентифіковані повідомлення транспортних даних. Це означає, що коли немає жодного обміну повідомленнями транспортних даних, мережеве з’єднання буде неактивним.

Оскільки кожне надіслане повідомлення про транспортні дані вимагає певної відповіді, ми можемо визначити, чи було порушено захищений сеанс або відключено. Якщо повідомлення про транспортні дані не було отримано протягом (*Keepalive-Timeout+Rekey-Timeout*) секунд, в цьому випадку повідомлення про ініціювання рукостискання надсилається хосту що не відповідає раз на кожен визначений період повторного тайм-ауту.

2.7.6 Взаємодія з системою відповідей на файли cookie

Коли сервер знаходиться під навантаженням, повідомлення про ініціювання рукостискання або повідомлення-відповідь на рукостискання може бути відхилено, а повідомлення-відповідь cookie надіслано. Отримавши відповідь на повідомлення cookie, що дозволяє хосту надіслати нове повідомлення про ініціацію чи відповідь із дійсним повідомленням msg.mac2, якого не буде відкинуто, хост що відповідає не повинен негайно повторно надіслати дійсне повідомлення.

Натомість він повинен просто зберігати розшифроване значення файлу

cookie із повідомленням відповіді файлу cookie, та дочекатися закінчення терміну очікування повторного введення таймеру для повторної спроби повідомлення про ініціювання рукостискання. Це запобігає потенційному зловживанню пропускною здатністю та допомагає полегшити навантаження на хост.

2.8 Інтеграція WireGuarg у ядро Linux

Впровадження WireGuard всередині ядра Linux має на меті кілька цілей. По-перше, це впровадження повинно бути коротким і простим, тому перевірка та перегляд коду на наявність уразливих місць не тільки легкий, але й приємний. WireGuard реалізовано менш ніж у 4000 рядків коду (без урахування криптографічних примітивів).

По-друге, технологія повинна бути надзвичайно швидкою, щоб вона була конкурентоспроможною з IPsec. По-третє, вона повинна уникати розподілу ресурсів у відповідь на вхідні пакети. По-четверте, WireGuard повинен спокійно інтегруватися до існуючої інфраструктури ядра, інструментів та API. І по-п’яте, WireGuard повинен бути у вигляді зовнішнього модулю ядра і не вимагати жодних змін в основному ядрі Linux. WireGuard є не просто академічним проектом з ніколи не випущеним лабораторним кодом, а радше практичним проектом спрямований на готові до реалізації ідеї.

2.8.1 Система черг пакетів

Драйвер WireGuard має мітки, що вказують ядру на те, що він підтримує загальне розвантаження сегментації (GSO), введення-виведення розбросу та розвантаження контрольної суми обладнання, що в сумі означаєте що ядро передасть «Суперпакети» які значно перевищують розмір MTU до WireGuard, і ці пакети буть надіслані у чергу в верхні рівні стеку TCP/IP або UDP/IP.

Це дозволяє WireGuard працювати в пакетному режимі групи вихідних пакетів. Розбивши пакети на сегменти з меншим MTU, WireGuard намагається

зашифрувати, інкапсулювати та надсилати по UDP всі кадри відразу, кешування інформації про маршрутизацію для виділеного кластеру пакетів має обчислюватись лише один раз. Це має дуже важливий ефект для зменшення пропусків кешу.

Очікування моменту коли поки всі окремі пакети «суперпакета» будуть зашифровані та інкапсульовані для їх передачі до мережевого рівня створює відчутне навантаження на процесор, тому зберігання проміжних інструкцій та передбачення розділення гілок у кеші процесора в багатьох випадках збільшує продуктивність відправки на 35%. Також, як зазначалося, іноді вихідні пакети повинні бути в черзі до завершення успішного рукостискання.

Коли пакети нарешті можуть бути відправлені, вся черга існуючих пакетів розглядається як один «суперпакет», щоб отримати вигоду від оптимізацій згаданих вище. Щоб запобігти непотрібним розподілам, усі перетворення пакетів виконуються на місці , уникаючи необхідності копіювання. Це стосується не лише шифрування та дешифрування даних які відбуваються на місці, але також до певних користувацьких даних і файлів, надісланих за допомогою sendfile (2), бо вони теж обробляються за допомогою цієї черги обслуговування пакетів.

3 ТЕХНОЛОГІЯ РОЗГОРТАННЯ VPN WIREGUARD НА БАЗІ LINUX ВИКОРИСТАННЯМ РІЗНИХ ТОПОЛОГІЙ МЕРЕЖ

3.1 VPN на базі протоколу WireGuard

Технологія WireGuard є швидкою та безпечною альтернативою програмному забезпеченню для побудови VPN такому як OpenVPN або IPsec. Головною превагою WireGuard є багатопоточність та інтегрованість у ядро Linux, що дозволяє досягнути симетричної швидкості передачі інформації до 1 Гбіт/с на пристроях з низькою обчислювальною здатністю. Сам сервер є достатньо легким і зрозумілим для його швидкого розгортання. WireGuard клієнт вже підтримується на наступних операційних системах: Windows, macOS, Ubuntu, Android, iOS, Debian, Fedora, Mageia, Arch, OpenSUSE, Slackware, Alpine, Gentoo, NixOS, OpenWRT, Oracle Linux 8, Red Hat Enterprise Linux 8, CentOS 8, FreeBSD, OpenBSD, EdgeOS, RouterOS 7.1.

WireGuard дає можливість запускати такі служби як SSH, FTP та RDP не на зовнішньому IP серверу, а у внутрішній мережі VPN. Наприклад до власного серверу ви можете залишити собі доступ по SSH через зовнішню мережу адресу, а всім іншим користувачам давати доступ по SSH тільки через IP-VPN. Така практика розгортання мереж прибирає більший процент загроз для вразливих сервісів і забезпечує доступом до них лише авторизованих користувачів.

За допомогою WireGuard ви можете тунелювати весь, або частину свого інтернет трафіку через транспортну мережу в якості VPN на швидкості 1 Гбіт/с. Якщо ви знаходитесь за NAT сірим IP, то використання WireGuard на недорогому VPS, що дозволить вам зробити доступними сервери працюючі у вашій локальній мережі з будь-якої точки Інтернету.

Загально процес розгортання VPN WireGuard можна розділити на декілька наступних етапів:

1) розгортання серверу на базі Linux;
2) оновлення пакетів на сервері;

3) встановлення пакетів WireGuard та iperf3;
4) генерування відкритого і закритого ключів;
5) налаштування wg інтерфейсу;
6) запуск служб WireGuard;
7) перевірка проходження ICMP та тест швидкості.

Виконання цих загальних етапів дозволяє швидко розгорнути працюючу VPN мережу уникаючи такої складності налаштування як у OpenVPN або IPsec.

В наступних розділах буде детально розглянуто технологію побудови VPN мереж з використанням наступних топологій:

1) топологія точка – точка;
2) топлогія зірка;
3) повнозв’язна (mesh) топологія.

3.1 Побудова VPN за топологією «точка-точка» або «point to point»

Топологія «точка-точка» з’єднує два незалежні вузли мережі між собою. Кінцеві вузли що зв’язуються у мережі з топологією «точка-точка» не обов’язково мають зв’язуватися без проміжних точок, а навпаки як раз пристрої що працюють з подібною топологією з’єднуються через безліч проміжних вузлів. Така кількість проміжних вузлів не впливає на схему, а формує так званий «віртуальний канал». Віртуальний канал є логічним з’єднанням, створюваним всередині мережі між двома пристроями.

Для реалізації поставленого завдання нами буде орендовано 2 VPS машини у центрі обробки даних Hetzner. Звичайно, робочі машини можна було підняти на платформі VirtualBOX, але рішення з VPS будуть найбільш близькими до реалізації справжніх робочих рішень на базі WireGuard.

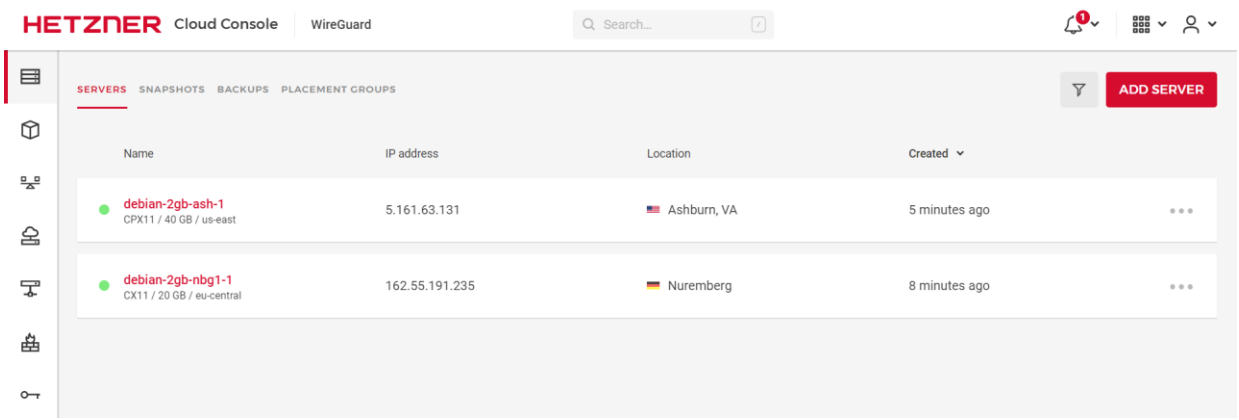

Рис. 3.1. Два VPS орендовані в ЦОД Hetzner для побудови VPN

На рисунку 3.1. ви можете побачити дві VPS ноди з IP-адресами 162.55.191.235 та 5.161.63.131. Ці VPS знаходяться у 3 датаценрі ЦОД Hetzner в Нюрбергу та в Ашберні США. Для правильного і об'єктивного тесту швидкості через iperf3 доцільно орендувати виділені сервери в різних ЦОД, що і буде зроблено в наступних розділах. Перейдемо до підготовки серверів для розгортання сервісу VPN WireGuard.

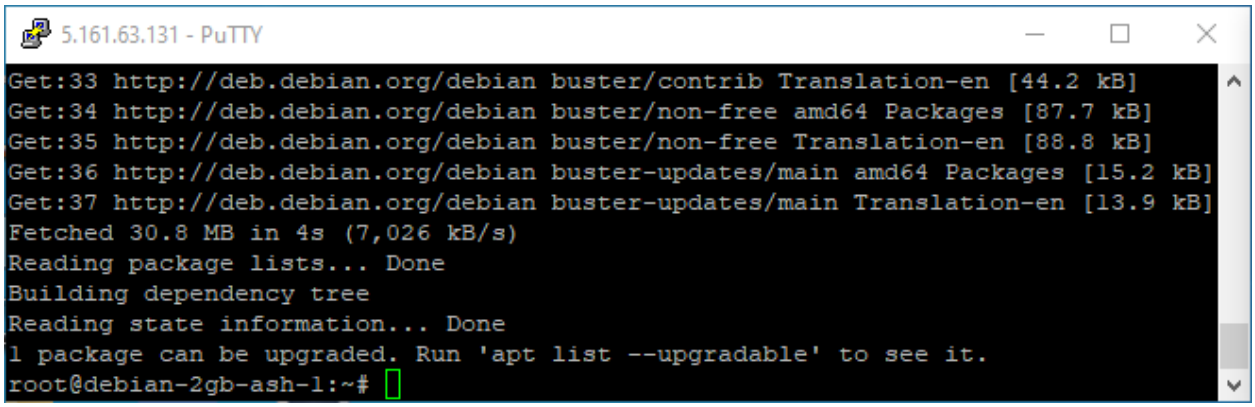

Рис 3.2. Процес оновлення пакетів Debian 10

За допомогою консольного інструменту apt-get, що зображений на рисунку 3.2., ми оновлюємо локальні індекси доступних пакетів у репозиторії (sudo apt-get update) і оновлюємо пакети (sudo apt-get upgrade) до останніх актуальних версій.

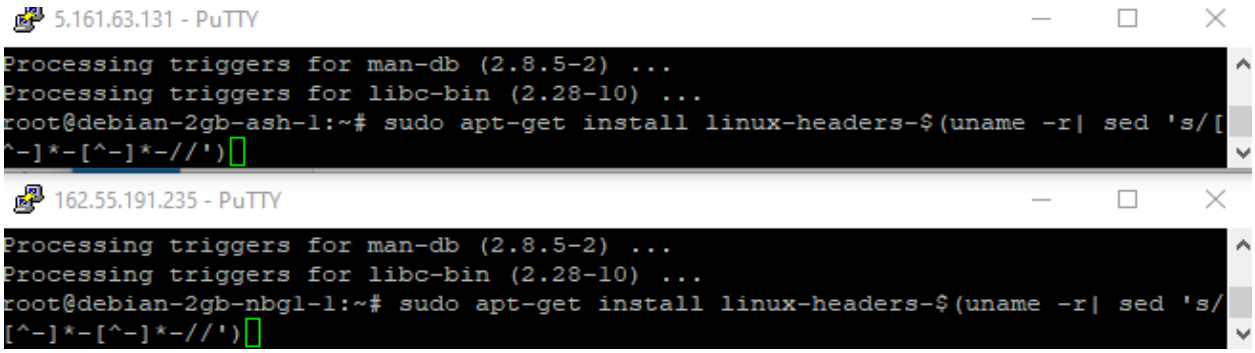

Рис. 3.3. Оновлення заголовків ядра Debian 10

Перед початком розгортання VPN на сервері бажано оновити заголовки ядра операційної системи, як зображено на рисунку 3.3., для того щоб актуалізувати витримки вихідного коду ядра що описують вихідні інтерфейси операційної системи для бінарної сумісності.

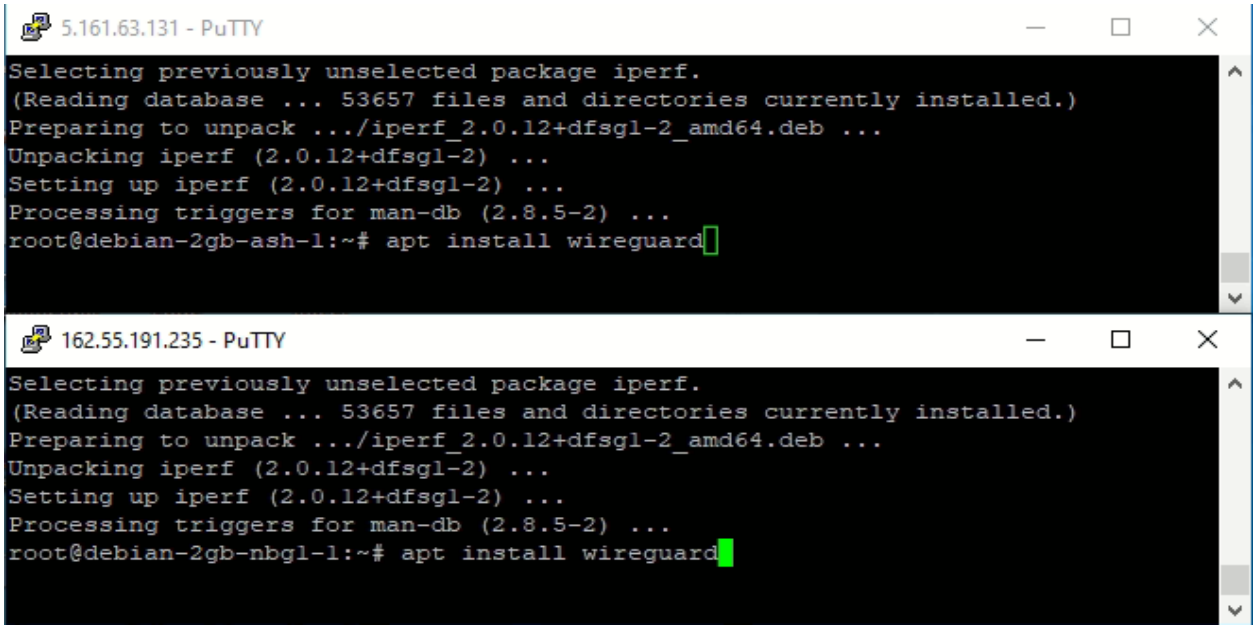

Рис. 3.4. Встановлення пакету VPN WireGuard та iperf3

Рис. 3.5. Топологія «точка-точка» для розгортання VPN WireGuard на серверах з ОС Debian 10

Після підготовки серверів до розгортання та встановлення необхідних пакетів розроблюється топологічна схема що висвітлює структуру мережі з плануємою адресацією. Як показано на рисунку 3.5. ми об’єднуємо 2 вузли шляхом з’єднання двох wg інтерфейсів.

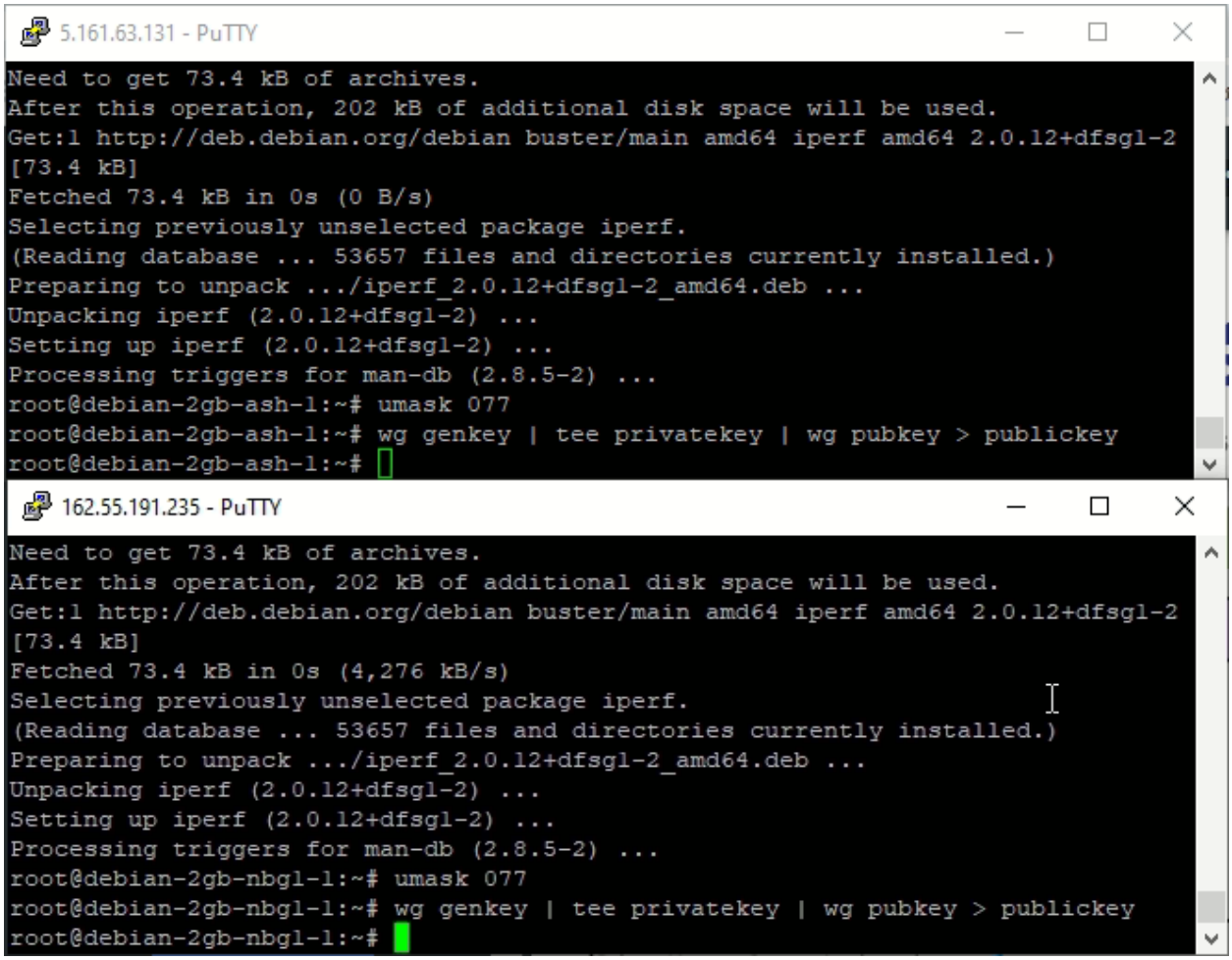

Рис. 3.6. Генерування відкритого і закритого ключа шифрування

Як ми можемо побачити на рисунку 3.6. за допомогою команди cd перешли до директорії /etc/wireguard/ де і будемо герувати пару відкритий-закритий ключ. Командою umask (user mask) визначаємо кінцеві права доступу до файлів у директорії, з міркувань безпеки доступ до ключів шифрування мають бути тільки у адміністратора. За допомогою команди wg genkey формуємо пару ключів доступу:

wg genkey > privatekey.key, ми отримуємо закритий ключ з виводом у файл /etc/wireguard/privatekey.key;

wg pubkey < privatekey.key > publickey.key, на основі закритого ключа формується відкритий ключ серверу і виводиться у файл /etc/wireguard/ publickey.key.

У нашому випадку для полегшення роботи процедура генерування пари ключів створена за допомогою конвеєра перенаправлення виводу, і має вид:

wg genkey | tee privatekey | wg pubkey > publickey.

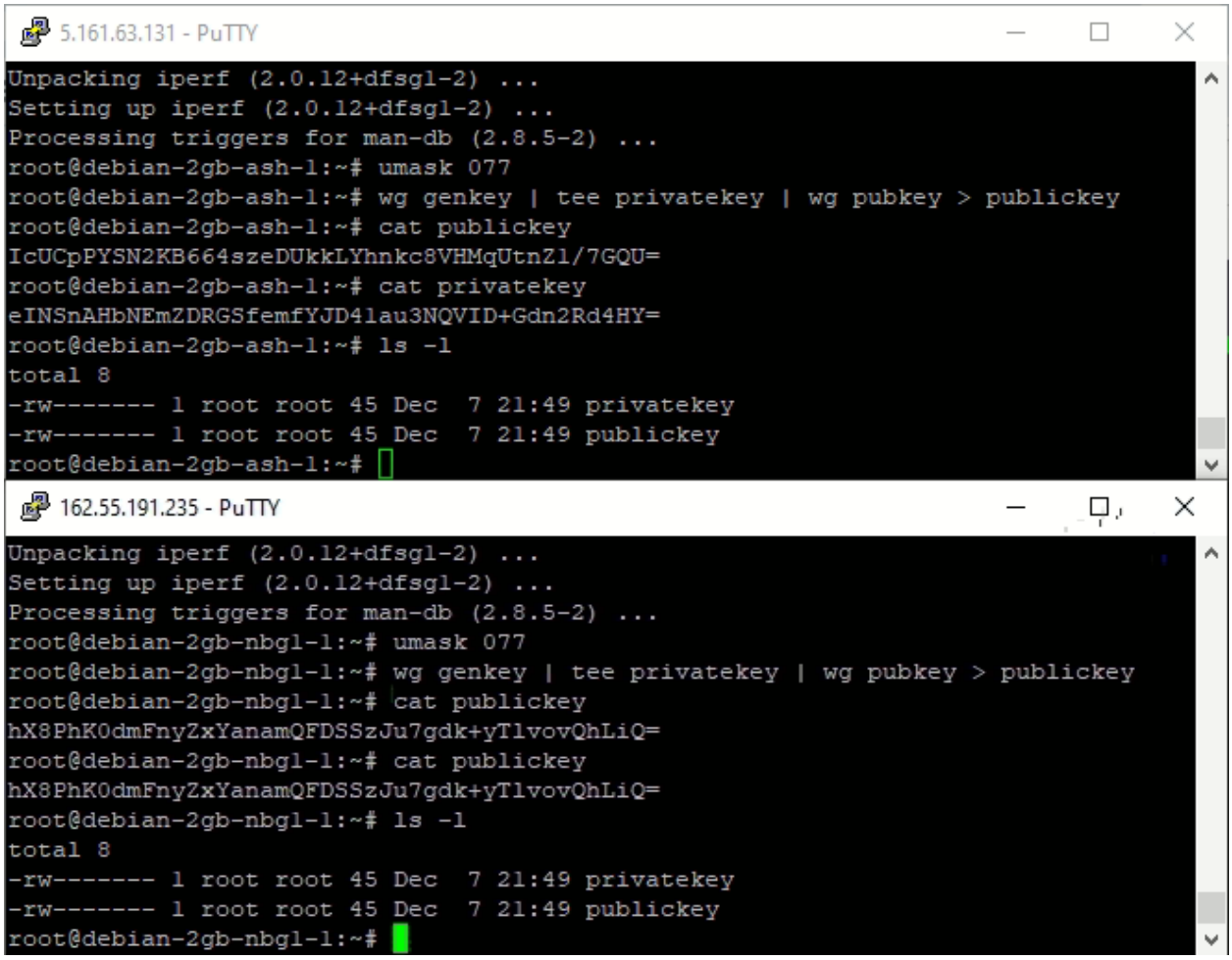

Рис. 3.7. Згенеровані приватний і публічний ключі

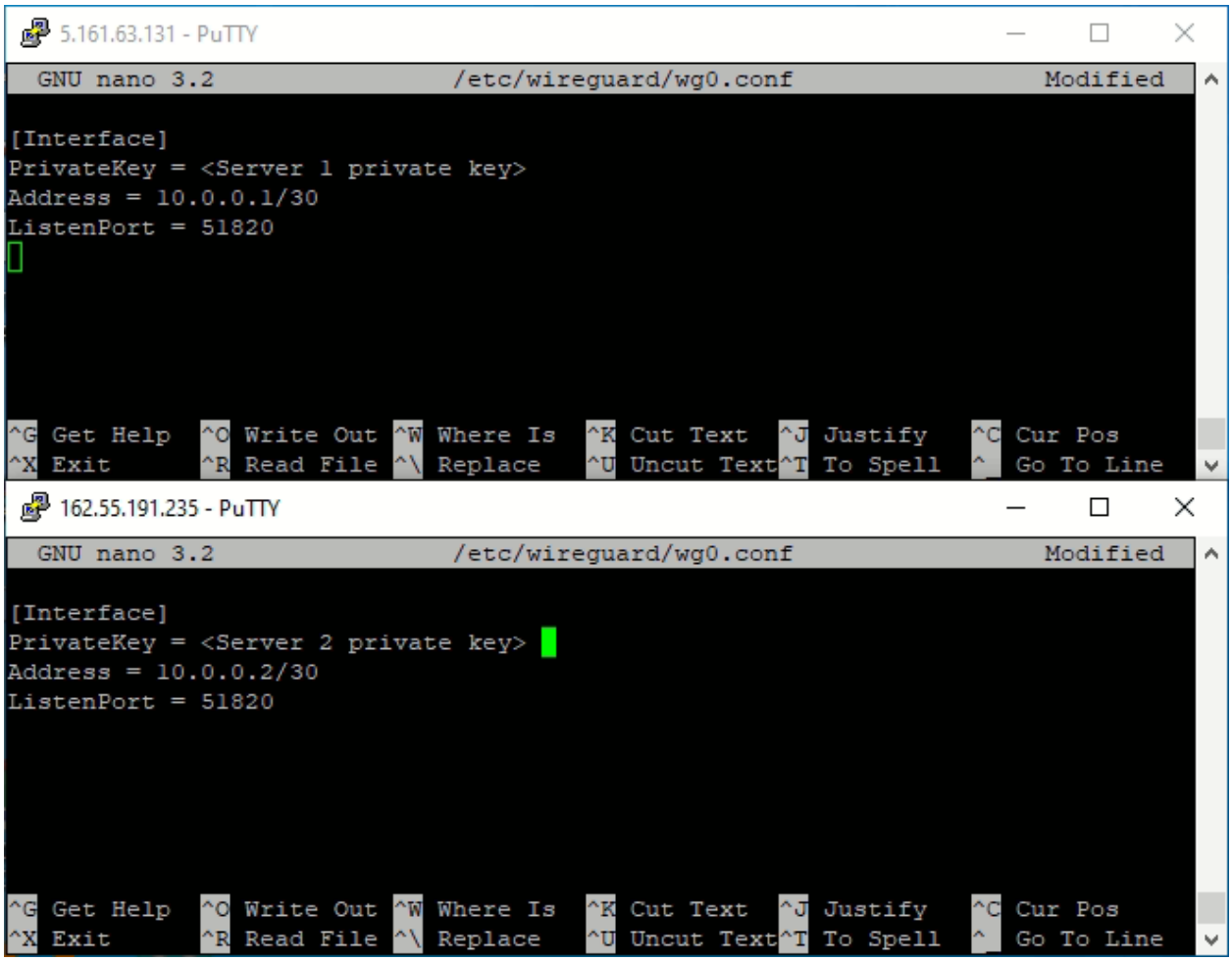

Рис. 3.8. Порожня конфігурація інтерфейсу wg0 для мережі з топологією «точка-точка»

Таким чином як зображено на рисунку 3.8. має виглядати конфігурація інтерфейсу wg0 для конфігурації інтерфейсу wg0 у топології «точка-точка».

Розглянемо детальніше структуру конфігурації інтерфейсу wg0:

[Interface]

PrivateKey = <Server private key> - згенерований приватний ключ серверу

Address = 0.0.0.0/0 - ip-адреса серверу всередині приватної мережі

ListenPort = 51820 - порт на який очікувати підключення

[Peer]

PublicKey = <Server Public key> - згенерований публічний ключ серверу сусіда

Endpoint = <Server Public IP>:51820 -IP/UDP зовнішня ip-адреса серверу

AllowedIPs = 0.0.0.0/0 - ip-адреса з яких дозволено приймати та передавати данні

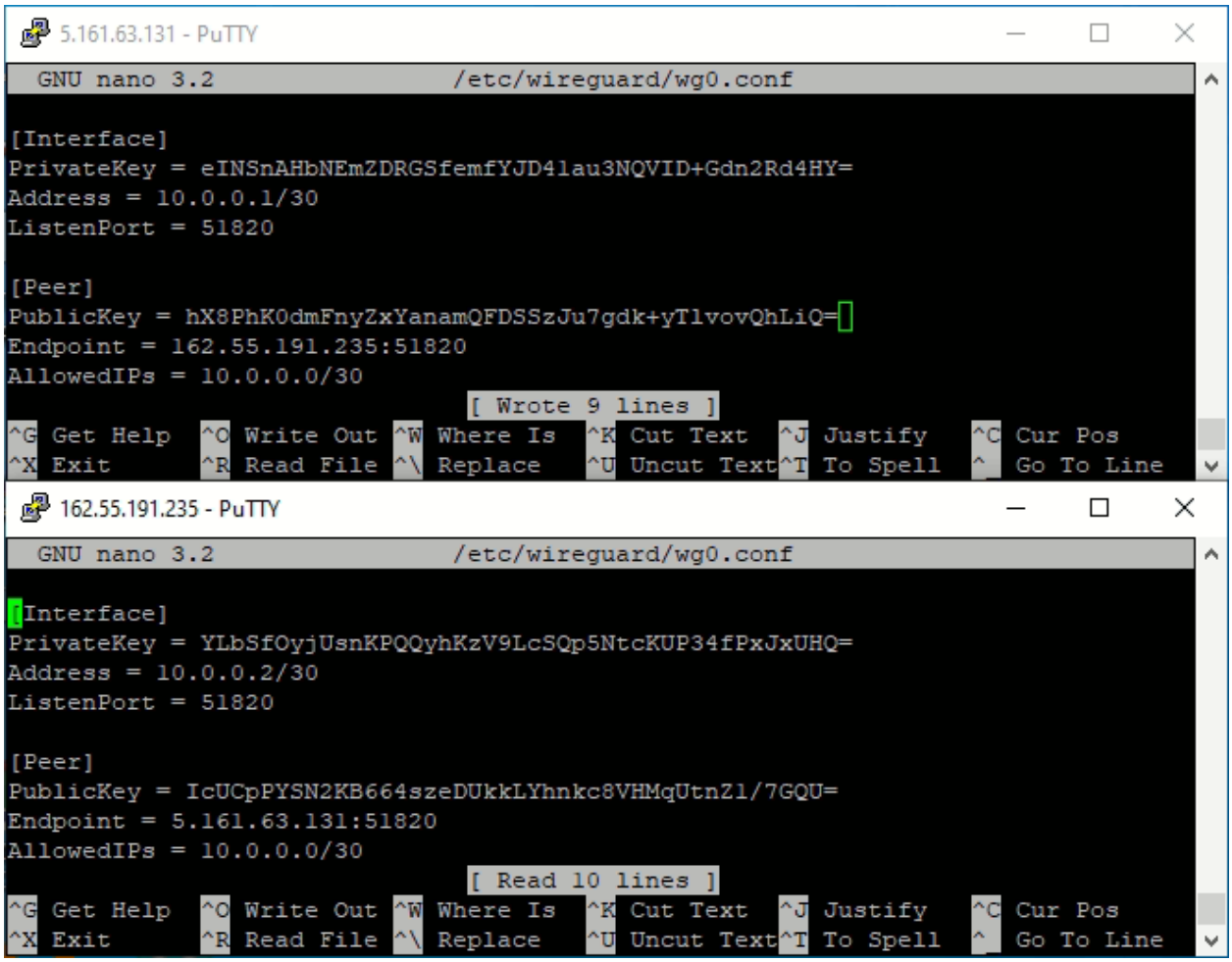

Рис. 3.9. Працююча конфігурація інтерфейсів wg0 на Server1 та Server2

Згідно рисунку 3.9. можемо прослідкувати за логікою налаштування. Сам інтерфейс серверу wg0 використовує закритий ключ згенерований на сервері, постійно прослуховує порт 51820 та має внутрішню адресу VPN мережі 10.0.0.1/30. Клієнт WireGuard має у своєму використанні публічний ключ що згенерований на сервері до якого треба буде підключатись, кінцевий ip та порт до якого буде здійснюватися підключення разом зі списком дозволених підмереж.

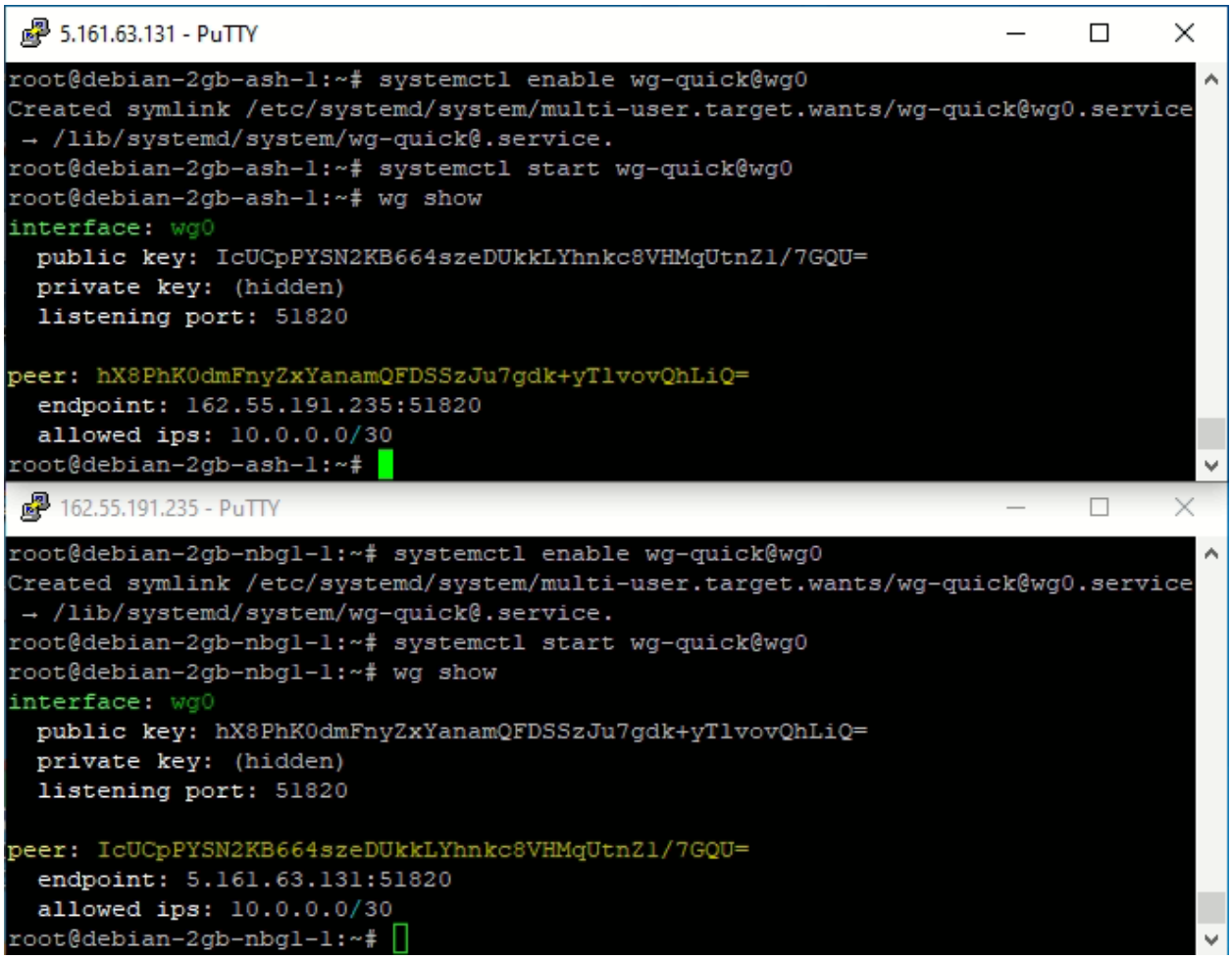

Рис. 3.10. Запуск сервісу WireGuard та перевірка стану інтерфейсів

Після налаштування інтерфейсів серверу та інтерфейсів клієнту, треба виконати налаштування автозапуску служби WireGuard (systemctl enable wg-quick@wg0) та запуску служби WireGuard (systemctl start wg-quick@wg0). Якщо запуск служб пройшов в шатному режимі що зображено на рисунку 3.10 то після виконання команди wg show ми маємо отримати вивід активного інтерфейсу серверу та клієнта.

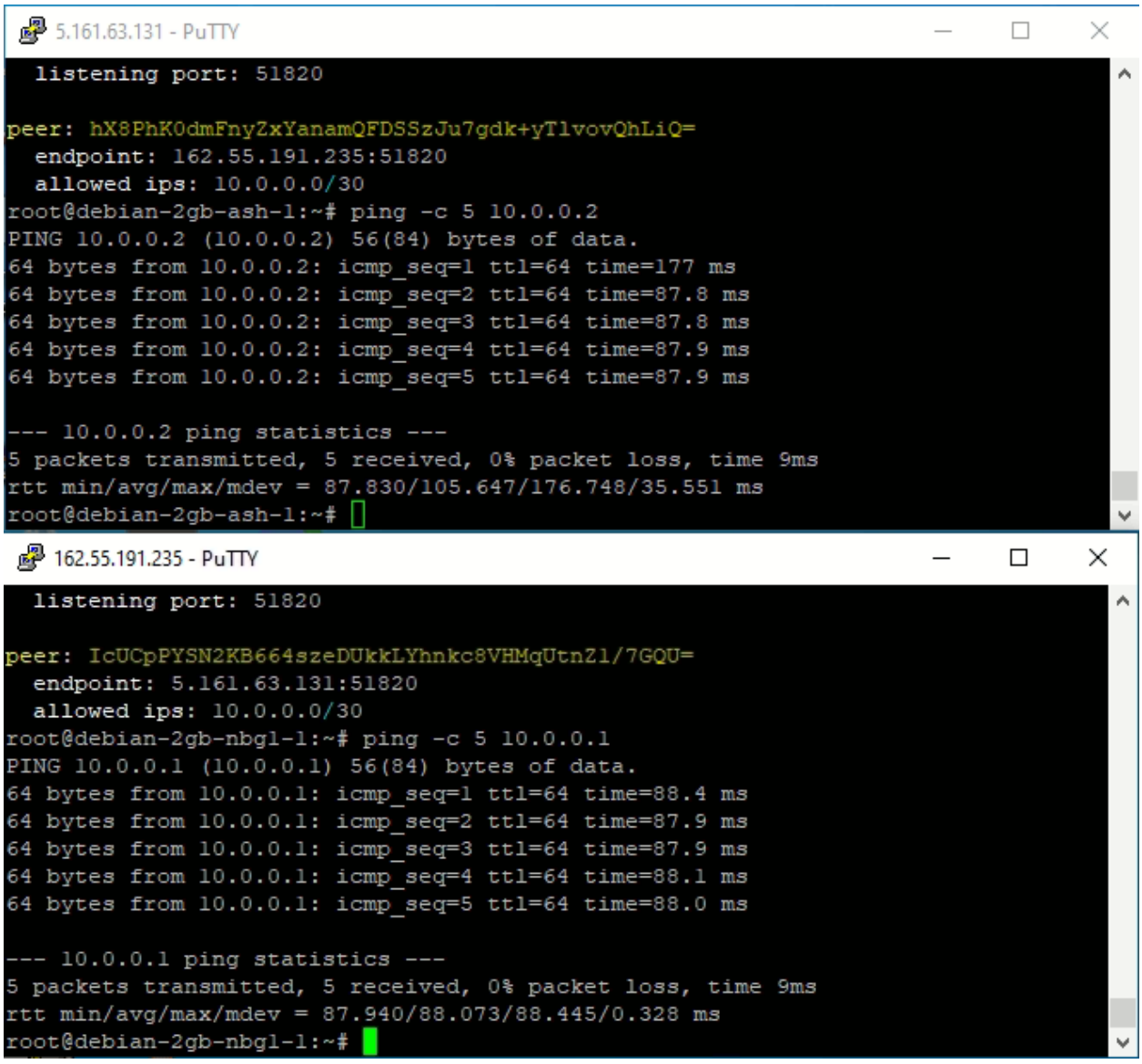

Рис. 3.11. Перевірка роботи тунелю шляхом ICMP запиту

Робота тунелю VPN перевіряється шляхом надсилання ICMP-ping пакетів до кінцевих IP-адрес внутрішньої VPN мережі. Як можемо побачити на рисунку 3.11. ping виконується успішно до обох сторін VPN тунелю. Якщо виконати команду wg show то на клієнтському інтерфейсі буде повідомлення про останнє успішний хендшейк. Таку конфгурацію можна вважати коректно налаштованою і цілком робочою.

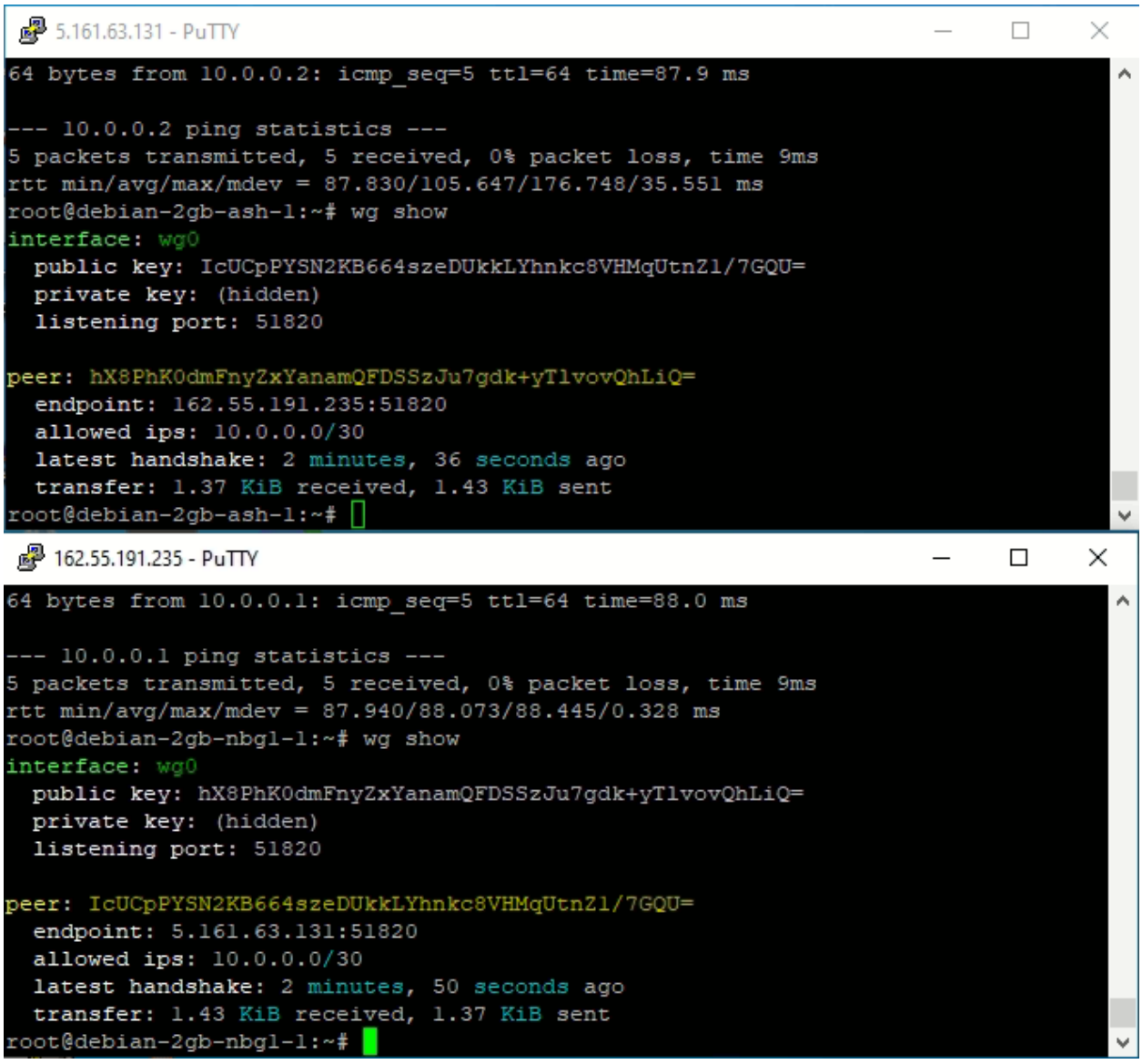

Рис. 3.12. Процес успішного рукостискання у тунелі і процес передачі даних

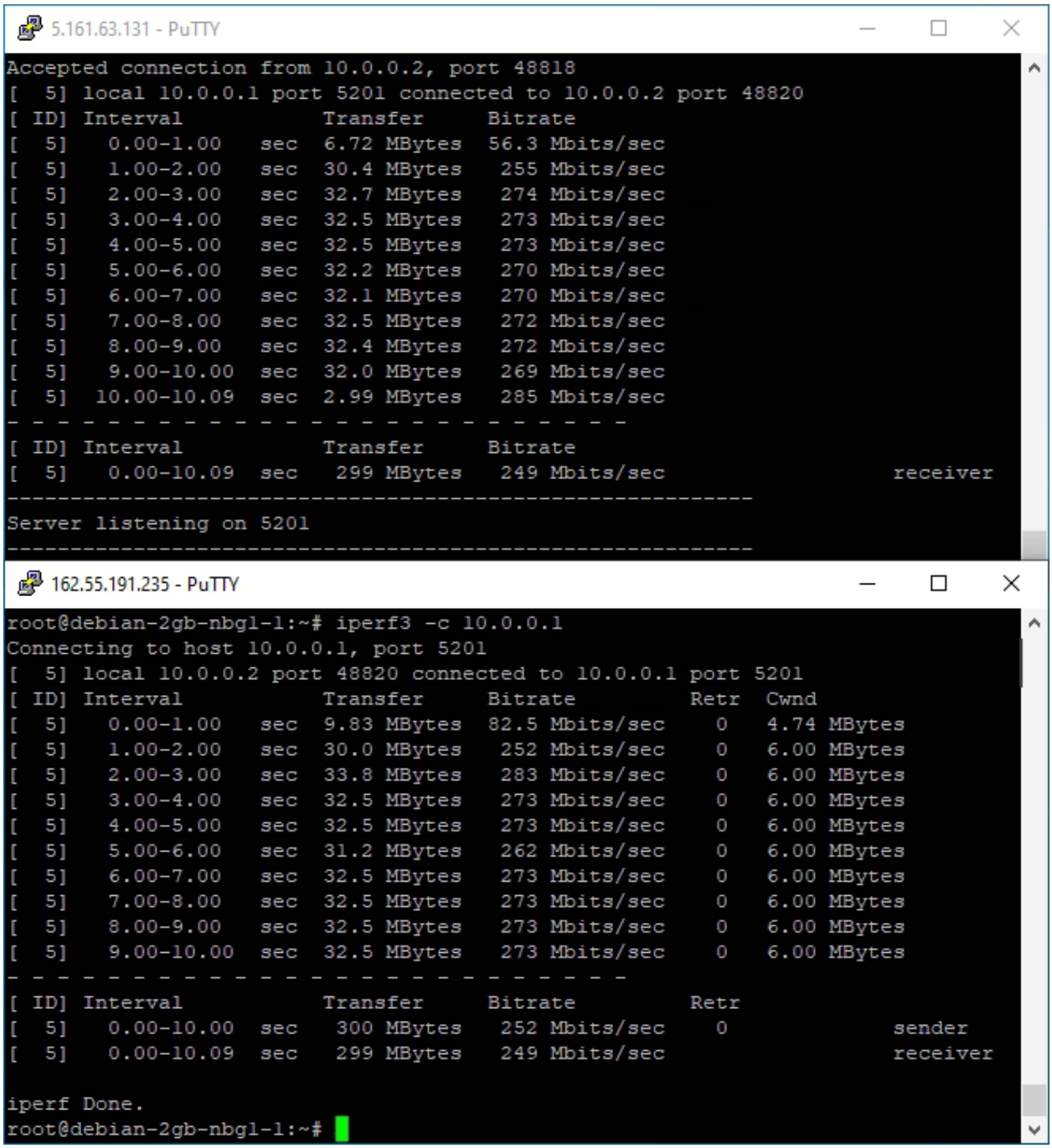

Рис. 3.13. Тест швидкості передачі даних по тунелю WireGuard за допомогою iperf3

Для виконання тесту переводимо ноду Sever1 у режим серверу, а нода Sever2 переводиться у режим клієнту iperf3. Шляхом пересилання інформації з серверу на сервер отримуємо швидкість на вивантаження 252 Мбіт/с та завантаження 249 Мбіт/с.

Тепер зрівняємо отримані результати швидкості передачі даних по шифрованому тунелю зі швидкістю передачі даних без шифрування.

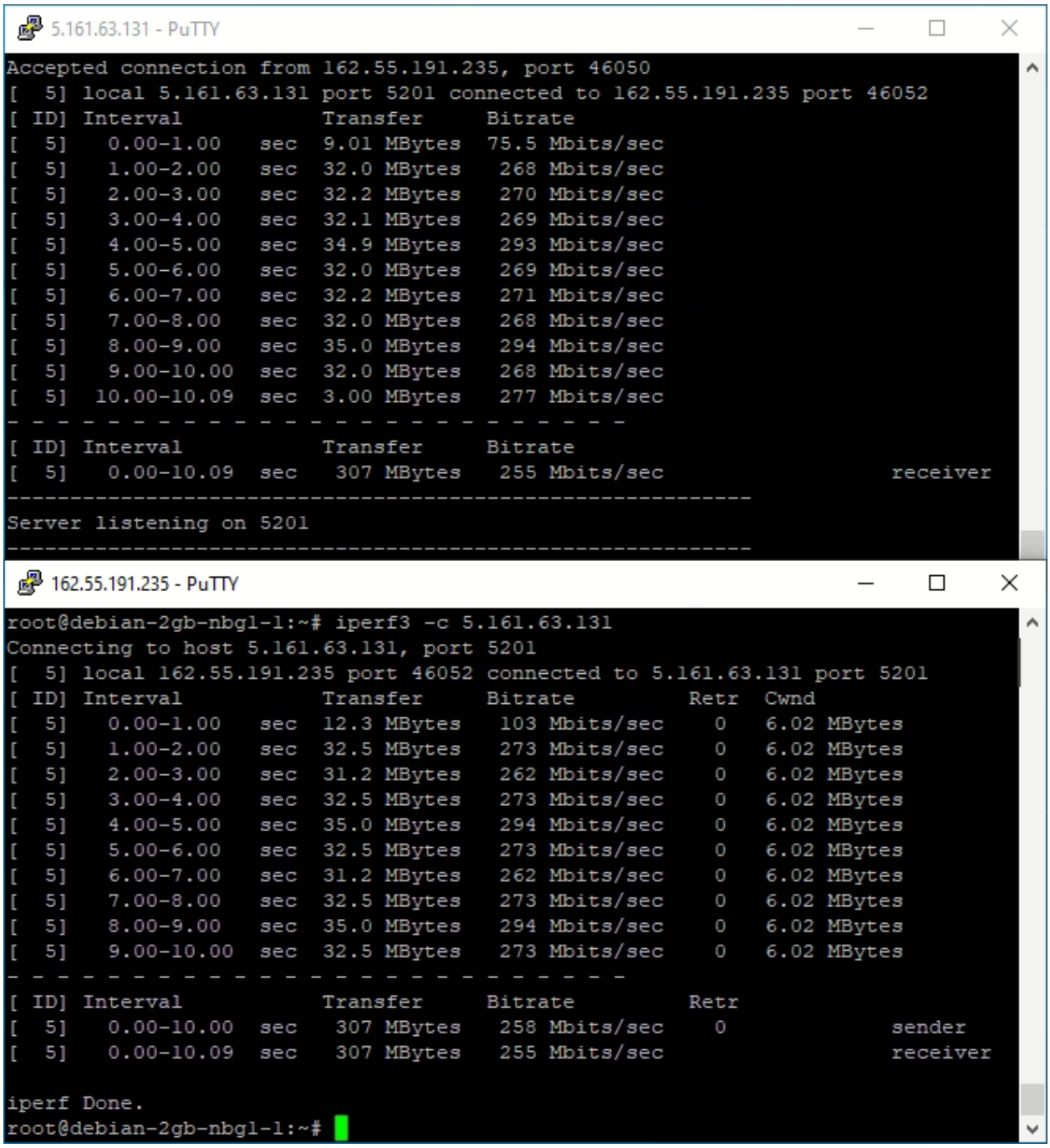

Рис. 3.14. Тест швидкості передачі не шифрованих даних за допомогою iperf3

Як можна побачити на рисунку 3.14. швидкість обміну даними по тунелю WireGuard не поступається швидкості обміну даними без шифрування.

Виходячи з результатів тесту на рисунку 3.13. і рисунку 3.14. можна

переконатись у достовірності показників швидкості зазначених розробниками.

3.2 Побудова VPN за топологією «зірка»

Зірка – це топологія локальної мережі, де кожна робоча станція приєднана до центрального пристрою (комутатор або маршрутизатор). Центральний пристрій керує рухом пакетів у мережі.

Топологія "зірка" на сьогоднішній день стала основною при побудові локальних мереж. Це сталося завдяки її численним перевагам:

вихід з ладу однієї робочої станції або пошкодження її кабелю не відбивається на роботі всієї мережі загалом;

відмінна масштабованість;

легкий пошук та усунення несправностей та обривів у мережі;

висока продуктивність;

в мережу легко вбудовується додаткове обладнання.

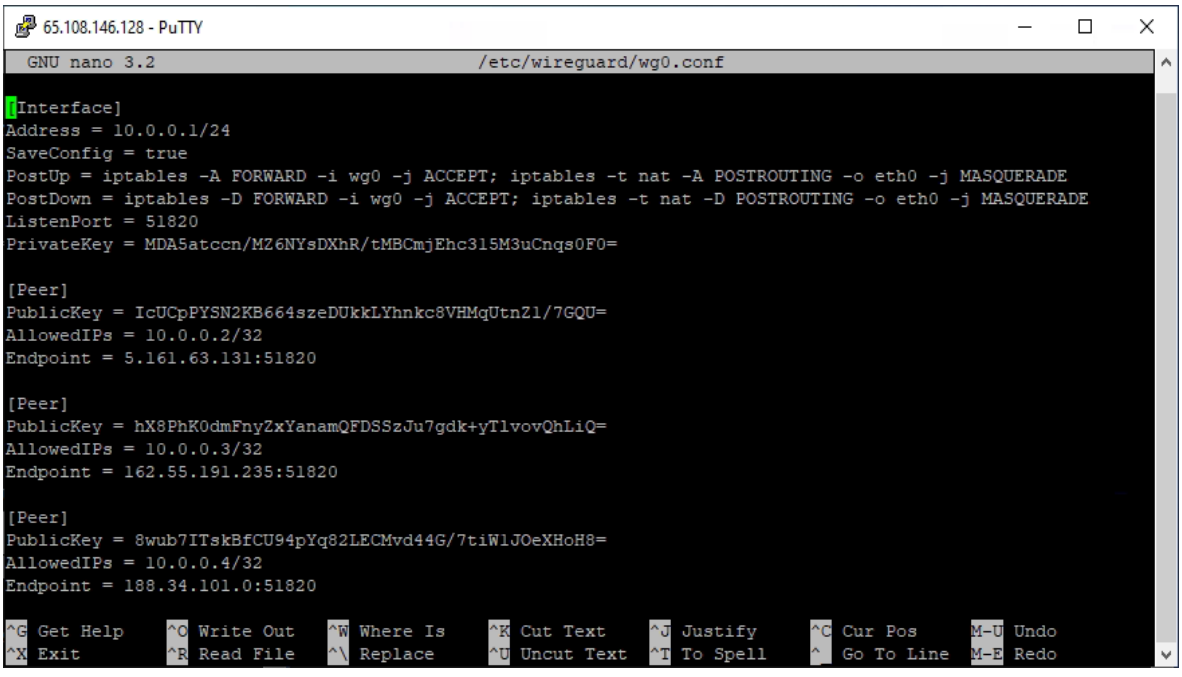

Рис. 3.15. Налаштування центрального вузла в топології «Зірка»

Таким чином має виглядати конфігурація інтерфейсу центрального вузла що зображено на рисунку 3.15. в топології «Зірка».

Розглянемо детальніше структуру конфігурації інтерфейсу:

[Interface]

PrivateKey = <Server 1 private key> - згенерований приватний ключ серверу

Address = 10.0.0.1/24 – ip-адреса в WireGuard VPN тунелі на стороні Server1

PostUp = iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -o ens3 -j MASQUERADE – набір команд або скрипт який буде запущено після старту сервера

PostDown = iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D POSTROUTING -o ens3 -j MASQUERADE - набір команд або скрипт який буде запущено після зупинки сервера

ListenPort = 51820 - порт на який прослуховує службаWireGuard

SaveConfig = true – Автоматичне збереження в конфігураційний файл параметрів EndPoint клієнтів що підключаються до сервера

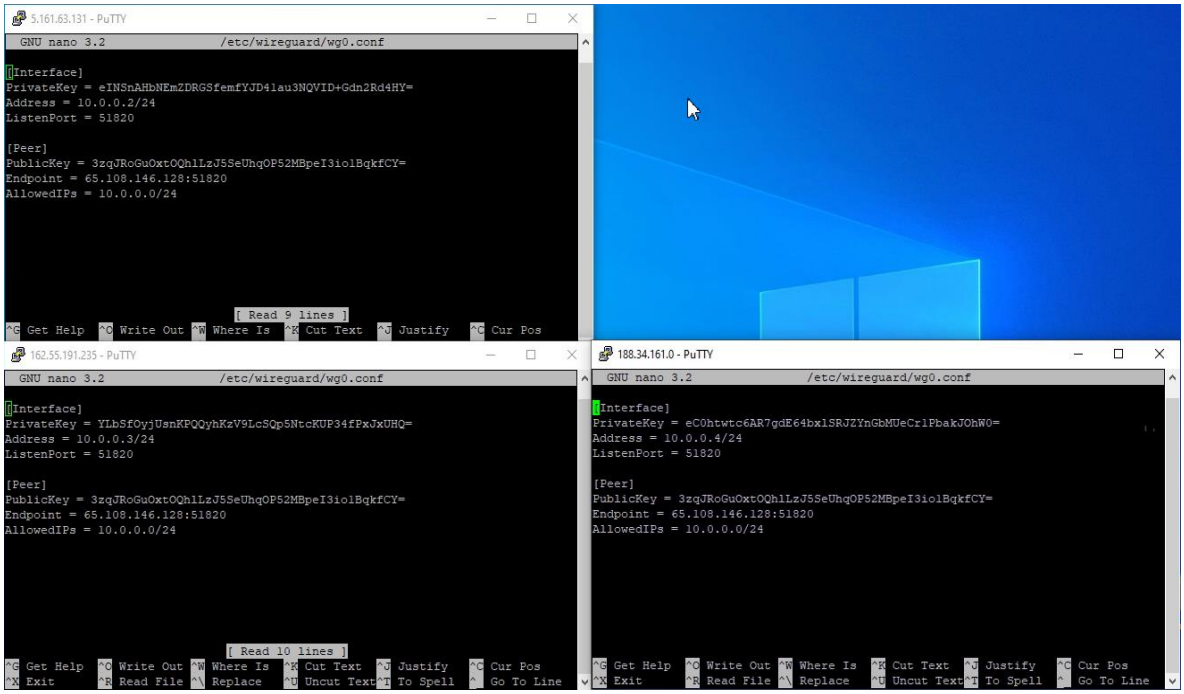

Рис. 3.16. Працююча конфігурація клієнтів в топології «Зірка»

Як можна побачити на рисунку 3.16. клієнтські інтерфейси використовують публічний ключ що був згенерований на сервері, порт і кінцевий ip до яких буде здійснюватися підключення.

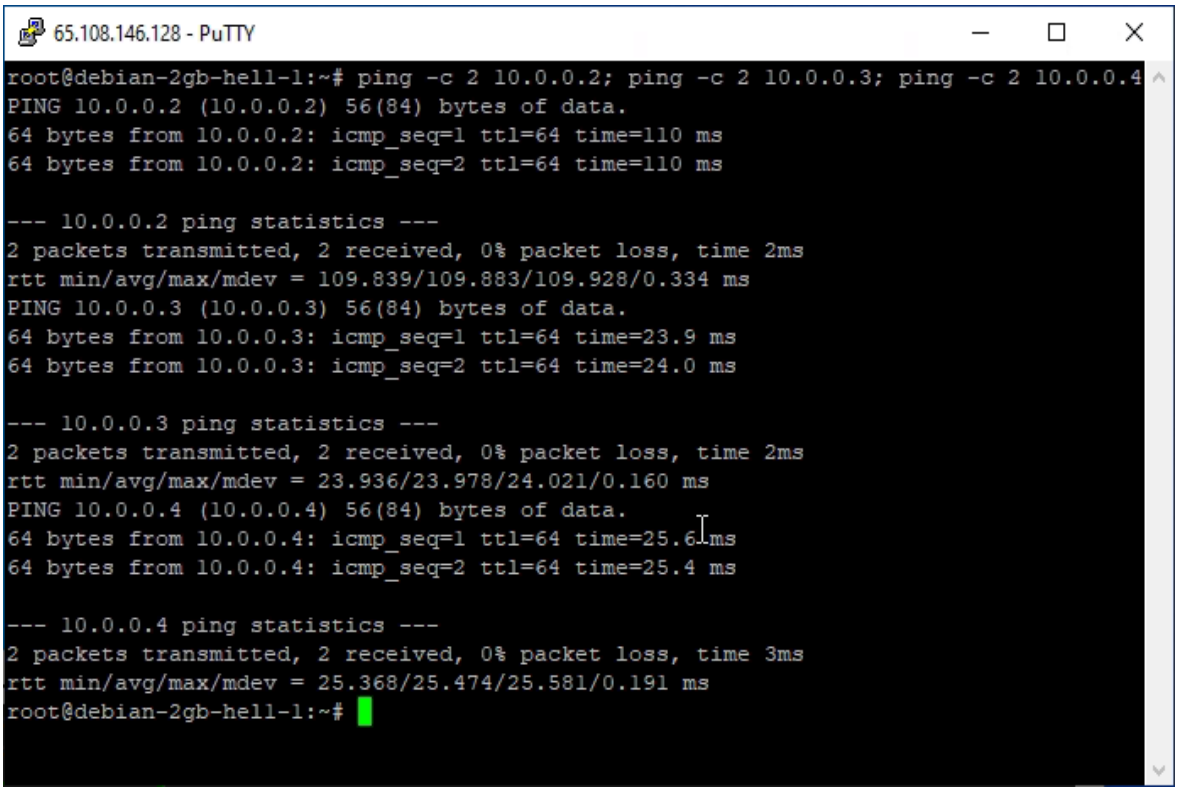

Рис. 3.17. Запуск сервісу WireGuard та перевірка стану інтерфейсу центрального вузла

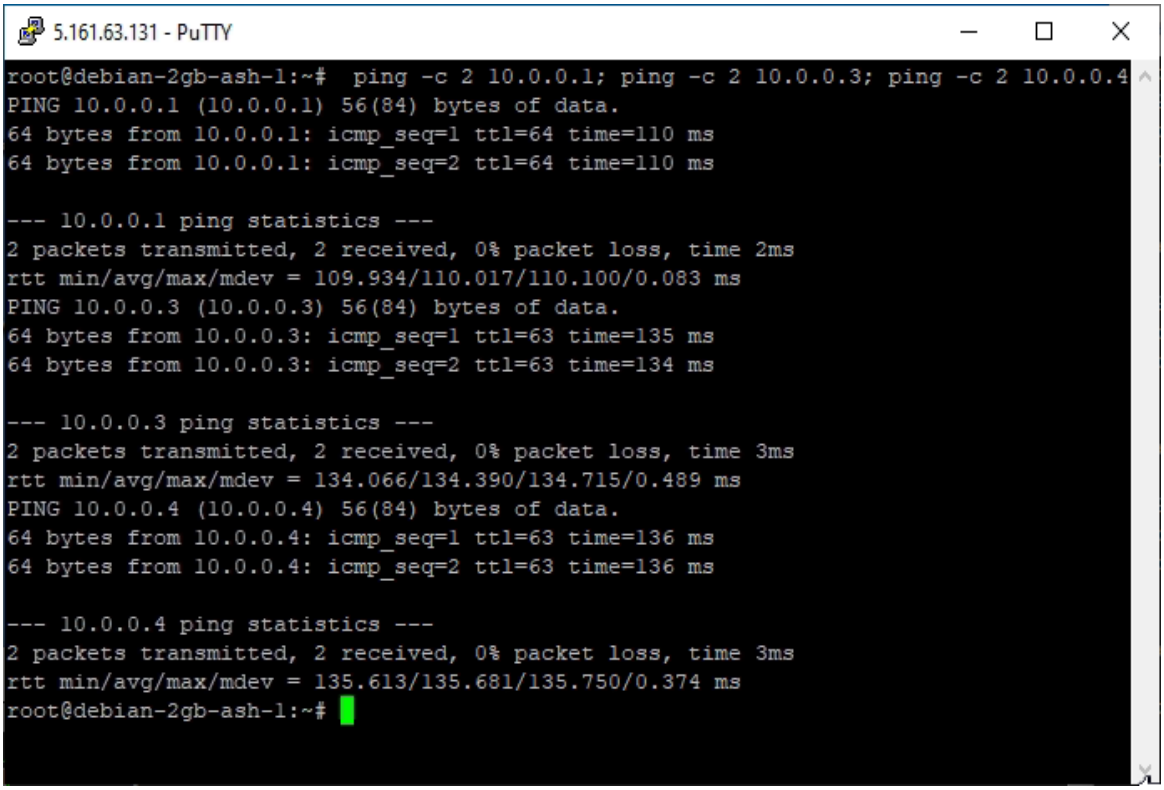

Рис. 3.18. Перевірка зв’язності та доступності вузлів

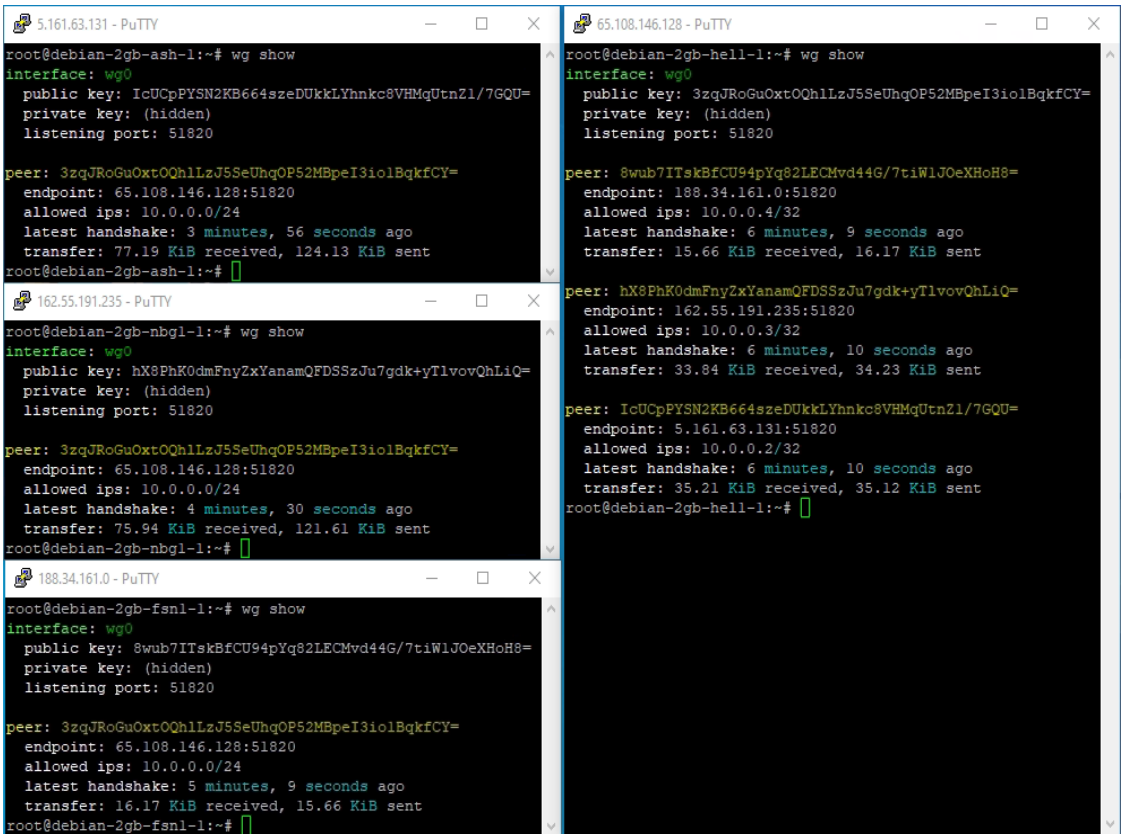

Рис. 3.19. Перевірка стану роботи інтерфейсу і успішного рукостискання

Сам інтерфейс серверу wg0 використовує закритий ключ згенерований на сервері, постійно прослуховує порт 51820 та має внутрішню адресу VPN мережі 10.0.0.1/.32, для забезпечення з’єднання центральній сервер використовує публічний ключ згенерований на клієнтському сервері кінцевий ip та порт до якого буде здійснюватися підключення разом зі списком дозволених підмереж.

3.3 Побудова VPN за топологією «Mesh»

Mesh Topology — це топологія комп'ютерної мережі, в якій кожна робоча станція мережі з'єднується з декількома іншими робочими станціями цієї мережі з можливим прийняттям на себе функцій комутатора для інших робочих станцій. Характеризується високою стійкістю до відмов, складністю налаштування і, для провідних мереж, надмірною витратою кабелю. Кожен комп'ютер має безліч можливих шляхів з'єднання з іншими комп'ютерами. Обрив кабелю не призведе до втрати з'єднання між двома комп'ютерами. Виходить з повнозв'язної шляхом

видалення деяких можливих зв'язків. Ця топологія припускає з'єднання великої кількості комп'ютерів і характерна, як правило, для великих мереж.

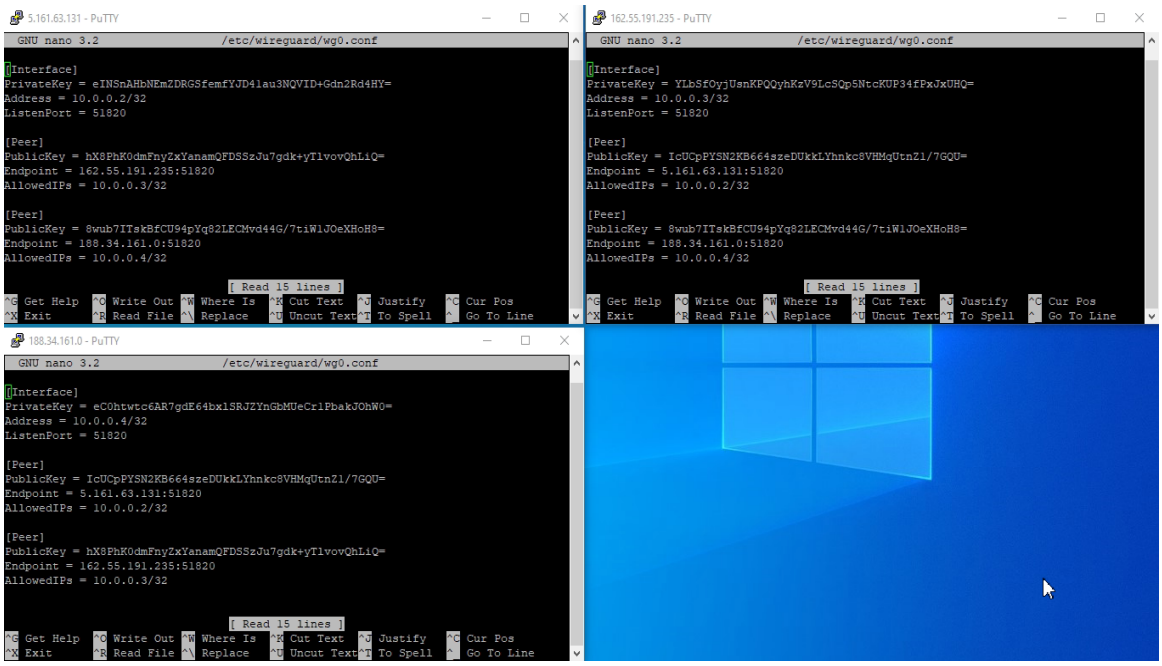

Рис. 3.20. Працююча конфігурація клієнтів в топології «Mesh»

Як ми бачимо на рисунку 3.20. кожен сервер у мережі має доступ до декількох вузлів одночасно. Кожен вузол використовує свій закритий ключ і прослуховує порт 51820, а також має кінцевий ір та порт до якого буде здійснюватися підключення разом зі списком дозволених підмереж і публічні ключі що згенеровані на серверах до яких він буде підключатись.

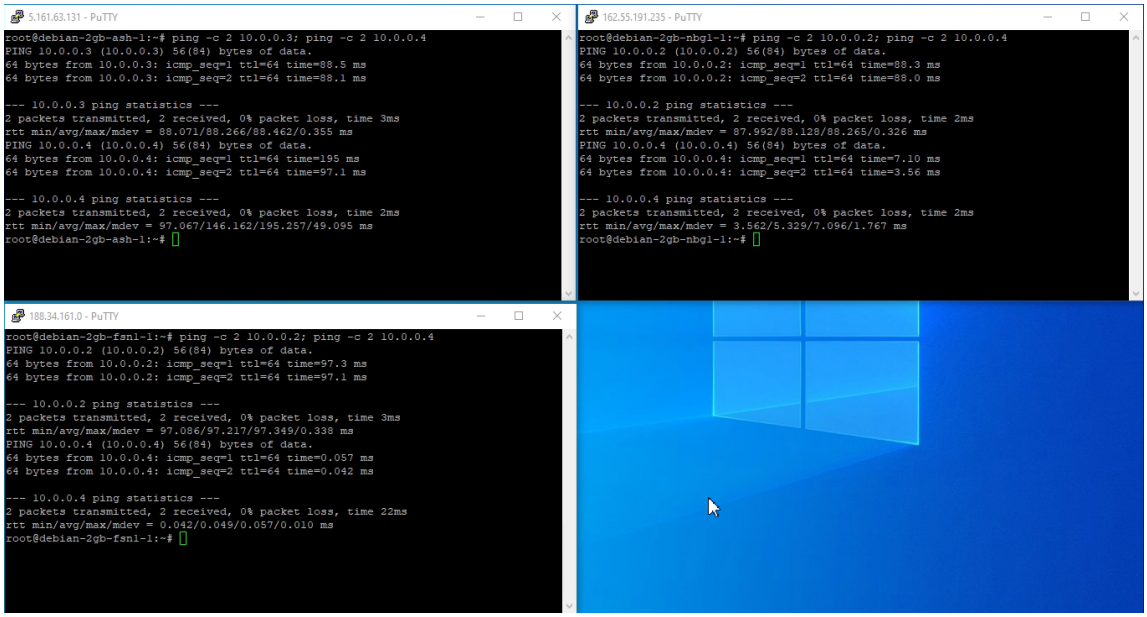

Рис. 3.21. Перевірка зв’язності та доступності вузлів

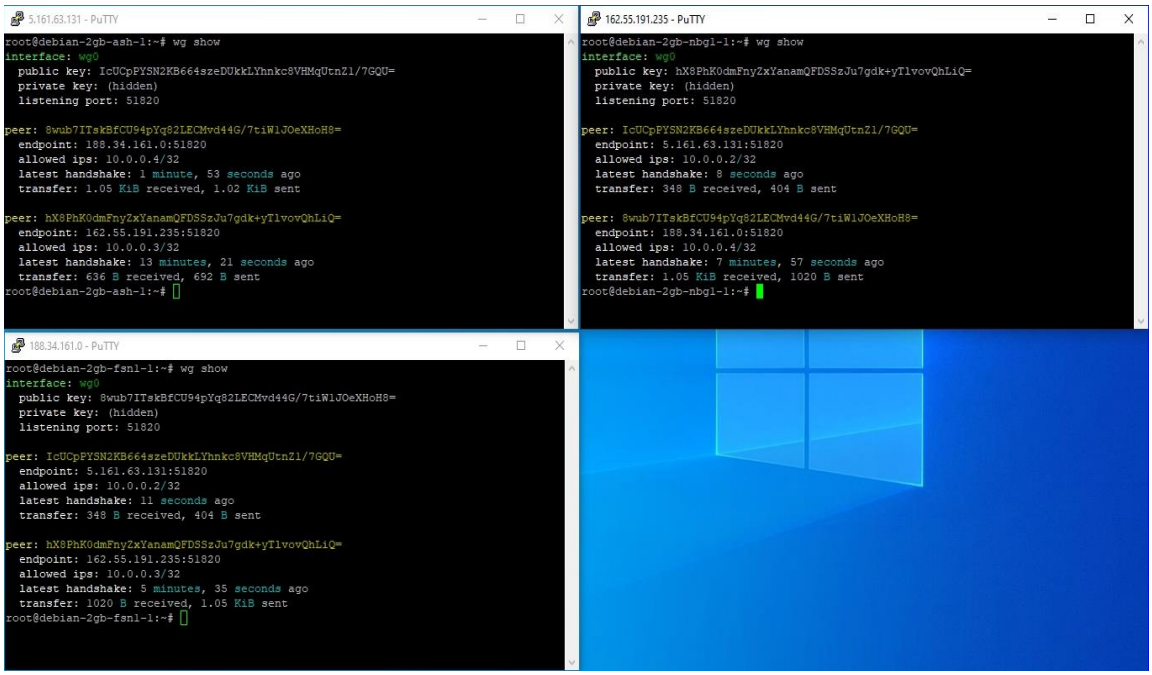

Рис. 3.22. Перевірка стану роботи інтерфейсу і успішного рукостискання

3.4 Перевірка захищеності обміну даними в тунелі WireGuard

За допомогою додатка Wireshark ми розпочали аналіз обміну мережевих пакетів в між орендованими серверами в результаті якого було перехоплено пакет даних с паролем від серверу у відкритому вигляді що зображено на рисунку 3.23. Для тесту використовувався незахищений протокол Telnet чим було продемонстровано його вразливість та доказано необхідність використання VPN WireGuard.

Перехоплений трафік ми відсортували за протоколом який використовувався для передачі даних і переглянувши інформацію шо передавалася між сервером і клієнтом отримали root-пароль серверу.

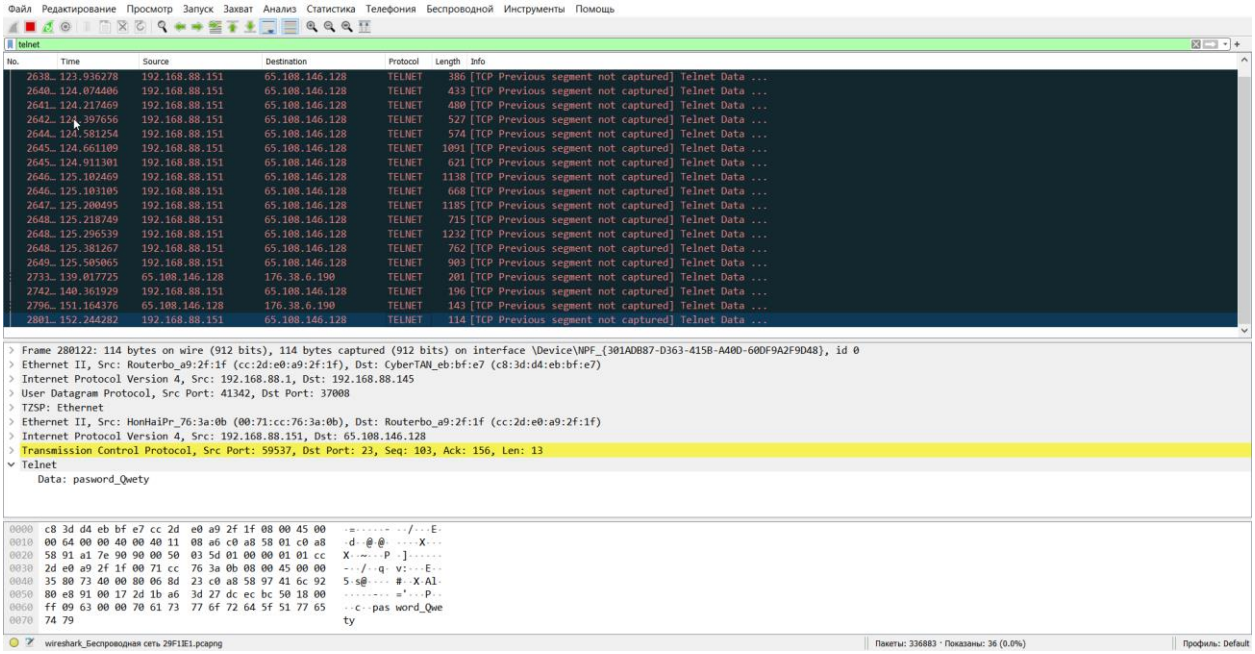

Рис. 3.23. Перехоплений пакет даних з паролем у відкритому вигляді

Для тесту захищеного обміну даними ми створили шифрований тунель на базі протоколу WireGuard, рисунок 3.24.

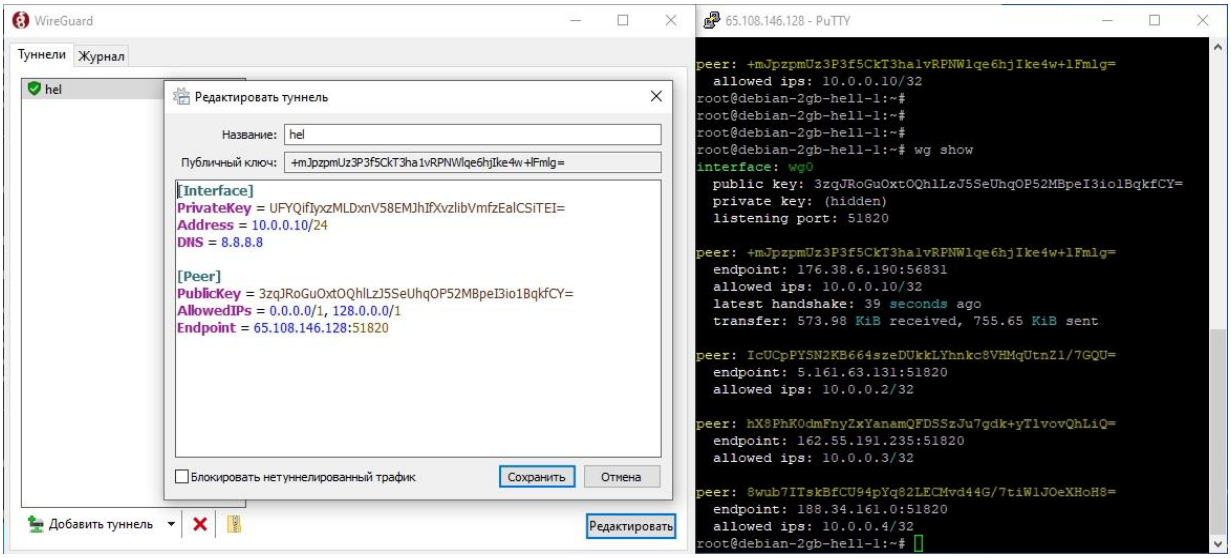

Рис. 3.24. Створення та перевірка роботи тунелю

Як можна бачити на рисунку 3.25. при повторній спробі перехопити пакет з даними якими міг би скористатися зловмисник, жодних даних у відкритому вигляді ми не отримали.

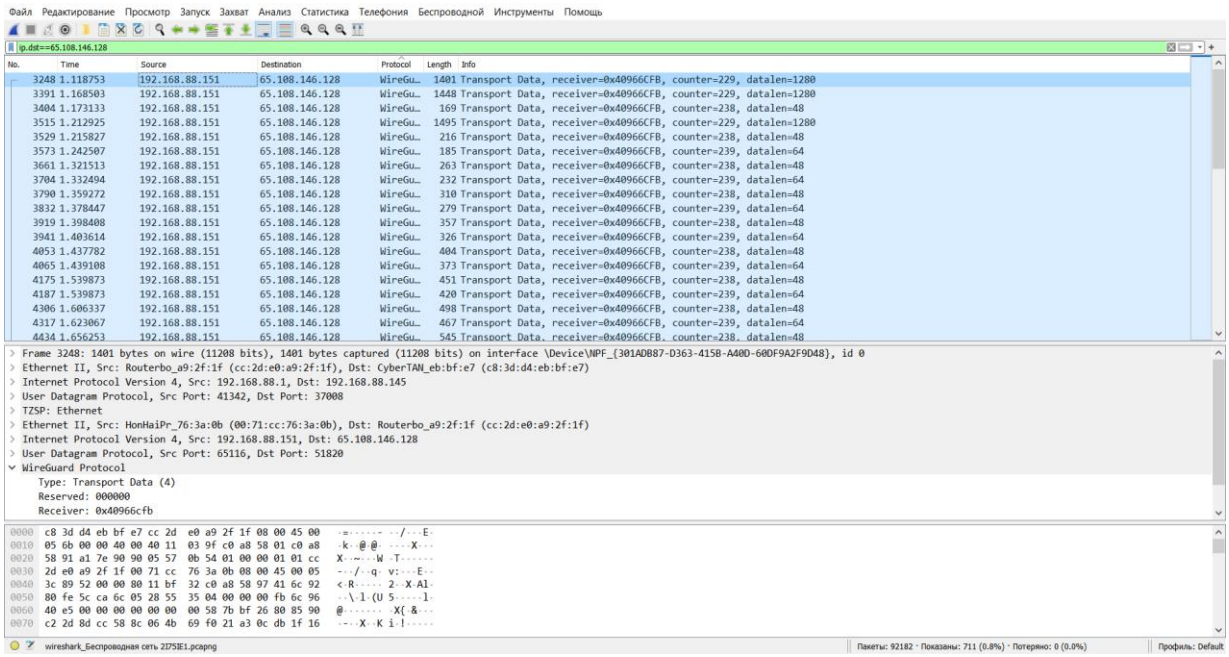

Рис. 3.25. Результат аналізу даних шо передаються через тунель VPN WireGuard

Спираючись на інформацію надану в даному та попередніх розділах, а також враховуючи той факт, що протокол WireGuard ще жодного разу не був скомпрометований можна вважати що рішення WireGuard VPN це простий, але швидкий і сучасний VPN, в якому використовується сучасна криптографія.

Він є кросплатформним (Linux, Windows, MacOS, BSD, iOS, Android) і може широко розгортатися. В даний час він знаходиться в стадії інтенсивної розробки, але вже може вважатися безпечним і простим у використанні рішенням у галузі VPN. Поєднання високошвидкісних криптографічних примітивів та того факту, що WireGuard знаходиться всередині ядра Linux, означає, що безпечна мережа може бути дуже швидкісною. Він підходить як для невеликих пристроїв, таких як смартфони, так і повністю завантажених магістральних маршрутизаторів.

ВИСНОВКИ

У магістерській роботі досліджено сутність проблеми забезпечення захищеного обміну даних в корпоративній мережі.

Встановлено сутність завдань захисту обміну даними в корпоративній мережі.

Проаналізовано існуючі технології захисту обміну даними в корпоративній мережі такі як OpenVPN, IPsec, WireGuard.

Проаналізовано методи та засоби забезпечення захищеного обміну даними в корпоративній мережі на базі VPN WireGuard. Проаналізовано різні види віртуальних мереж та розроблено рекомендації з їх розгортання в залежності від заданих технічних вимог до системи та з врахуванням необхідності підвищення ефективності систем. Досліджено протокол WireGuard та показано ефективність його застосування.

Проаналізовано основні функції та принципи реалізації захищеного обміну даними в корпоративній мережі, а також архітектуру VPN тунелів, та принципи мережного тунелювання. Охарактеризовано технології IPSec та OpenVPN.

На основі досліджень проведених в роботі запропоновано порядок застосування технології захисту обміну даними в корпоративній мережі із застосуванням рішення VPN WireGuard, а також розроблено рекомендації щодо застосування рішення WireGuard для забезпечення захищеного обміну даними в корпоративній мережі.

Досліджено принципи розгортання VPN на базі Linux, та підвищення безпеки приватних мереж за допомогою маршрутизації криптоключа.

Розроблено варіант моделі безпечного обміну даними в корпоративній мережі за допомогою рішення VPN WireGuard та рекомендації щодо прокладання VPN між вузлами мережі.

ПЕРЕЛІК ПОСИЛАНЬ

1. Віртуальні приватні мережі (VPN) [Електронний ресурс] – https://compbest.com.ua/ua/virtualnye-chastnye-seti-vpn/

2. BYOD Horizons-Global [Електронний ресурс] – https://www.cisco.com/c/dam/en_us/about/ac79/docs/re/BYOD_Horizons-Global.pdf

3. Олифер Виктор и Олифер Наталья. Виртуальные частные сети: администраторы требуют гарантий. Часть I [Електронний ресурс] – https://www.olifer.co.uk/articles/vpn/vpn.htm

4. Владимир Хазов. VPN – типы подключения и проверка безопасности [Електронний ресурс] – https://vasexperts.ru/blog/vpn-tipy-podklyucheniya-iproverka-bezopasnosti/

5. Хілах Мазіяр. Що таке VPN? І чому вона вам [ДІЙСНО] потрібна у 2020 році [Електронний ресурс] – https://uk.vpnmentor.com/blog

6. C. Kaufman et al. Internet Key Exchange Protocol Version 2. RFC 5996. RFC Editor, Sept. 2010 [Електронний ресурс] – http://www.rfc-editor.org/rfc/rfc5996.txt

7. Adam Langley and Yoav Nir. ChaCha20 and Poly1305 for IETF Protocols. RFC 7539. RFC Editor, May 2015 [Електронний ресурс] – http://www.rfc-editor.org/rfc/rfc7539.txt

8. Базовые понятия и классификация VPN [Електронний ресурс] – https://hidemy.name/ru/articles/bazovye-ponjatija-i-klassifikacija-vpn/

9. Безопасность сетей. Лекция 11. Виртуальные частные сети [Електронний ресурс] – https://intuit.ru/studies/courses/102/102/lecture/2991

10. IPsec made simple [Електронний ресурс] – https://briolidz.wordpress.com/2012/01/23/ipsec-made-simple/amp/

11. .IP security (IPSec) [Електронний ресурс] – https://www.geeksforgeeks.org/ipsecurity-ipsec/

12. Технологии Cisco VPN и выбор VPN технологии туннелирования [Електронний ресурс] – https://blog.telecom-sales.ru/tehnologii-cisco-vpn-i-vybor-vpn-tehnologii/

13. IPSec VPN клиент/сервер [Електронний ресурс] – https://help.keenetic.com/hc/ru/articles

14. Daniel J. Bernstein. “ChaCha, a variant of Salsa20”. In: SASC 2008. [Електронний ресурс] – https://cr.yp.to/chacha/chacha-20080128.pdf

15. Классификация сетей VPN [Електронний ресурс] – https://ozlib.com/825055/informatika/klassifikatsiya_setey

16. Немного о 2FA: Двухфакторная аутентификация [Електронний ресурс] – https://habr.com/ru/company/1cloud/blog/277901/

17. .Рябко Е.И. Калейдоскоп VPN-технологий [Електронний ресурс] – https://cyberleninka.ru/article/n/kaleydoskop-vpn-tehnologiy/viewer

18. R. Moskowitz et al. Host Identity Protocol Version 2. RFC 7401. RFC Editor, Apr. 2015. [Електронний ресурс] – http: //www.rfc-editor.org/rfc/rfc7401.txt

19. VPN-шифрование (всё что нужно знать) [Електронний ресурс] – https://www.cactusvpn.com/ru/beginners-guide-to-vpn/vpn-encryption/

20. Принципы организации VPN [Електронний ресурс] – http://ciscotips.ru/vpn

21. Paul E McKenny et al. Paul E. McKenny et al. “Read-Copy Update”. In: Ottawa Linux Symposium. June 2002, pp. 338–367. [Електронний ресурс] – http://www.rdrop.com/~paulmck/RCU/rcu.2002.07.08.pdf

22. Добреля Віталій Олексійович. Проблеми забезпечення захищеного обміну даними в корпоративній мережі. ВСЕУКРАЇНСЬКА НАУКОВА КОНФЕРЕНЦІЯ «АКТУАЛЬНІ ПРОБЛЕМИ КІБЕРБЕЗПЕКИ». Державний Університет Телекомунікацій. 27 жовтня 2021. Тези доповідей. С.71 –72. http://www.dut.edu.ua/uploads/p_2099_79407917.pdf