WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ CYBERNETYKI

Temat pracy: KONFIGURACJA BEZPIECZNEGO POŁĄCZENIA VPN Z WYKORZYSTANIEM OPENVPN

KRYPTOLOGIA I CYBERBEZPIECZEŃSTWO

.....

(kierunek studiów)

CYBEROBRONA

.....

(specjalność)

Autor:

Łukasz ROZBICKI

Warszawa 2025

Spis treści

Rozdział I. Podstawowe informacje o projekcie

I.1. Wstęp

W dobie dynamicznego rozwoju technologii informatycznych, coraz większą rolę odgrywa ochrona danych przesyłanych w sieciach publicznych i prywatnych. Jednym z najczęściej stosowanych rozwiązań jest wykorzystanie wirtualnych sieci prywatnych (VPN), które umożliwiają bezpieczne tunelowanie ruchu sieciowego oraz zapewnienie poufności i integralności danych. Projekt ten ma na celu skonfigurowanie środowiska VPN z wykorzystaniem OpenVPN na serwerze Ubuntu 20.04 oraz klienta Windows, co umożliwi bezpieczne przesyłanie informacji między zdalnymi lokalizacjami.

I.2. Założenia projektu

1. Platforma serwerowa i kliencka:
    - Serwer: Ubuntu Server 20.04 LTS skonfigurowany jako host wirtualnej maszyny.
    - Klient: system Windows z zainstalowanym oprogramowaniem OpenVPN.
2. Technologie i narzędzia:
    - OpenVPN jako główne oprogramowanie do zarządzania siecią VPN.
    - Easy-RSA do generowania kluczy kryptograficznych i certyfikatów.
    - Protokóły szyfrowania SSL/TLS do zabezpieczania połączeń.
3. Kroki implementacji:
    - Instalacja i konfiguracja serwera Ubuntu oraz OpenVPN.
    - Utworzenie infrastruktury kluczy publicznych (PKI) oraz generacja certyfikatów dla serwera i klienta.
    - Konfiguracja zapory sieciowej i przekierowanie ruchu VPN.
    - Weryfikacja poprawności konfiguracji i testowanie połączenia między serwerem a klientem.
4. Bezpieczeństwo:
    - Użycie szyfrowania AES-256-GCM oraz funkcji skrótu SHA-256 w celu zapewnienia integralności danych.
    - Ograniczenie dostępu do kluczy i certyfikatów poprzez odpowiednie uprawnienia plików.

I.3. Cel projektu

Głównym celem projektu jest praktyczne zastosowanie kryptografii w celu zapewnienia bezpiecznej komunikacji sieciowej. Wdrożenie rozwiązania VPN umożliwi:

- Zabezpieczenie przesyłanych danych przed nieautoryzowanym dostępem.
- Ochronę tożsamości użytkowników i ich aktywności w sieci.
- Demonstrację poprawnej konfiguracji systemu OpenVPN oraz możliwości jego zastosowania w środowiskach firmowych i domowych.

I.4. Zakres prac

Projekt obejmuje następujące etapy:

1. Instalacja serwera Ubuntu i niezbędnych narzędzi.
2. Konfiguracja infrastruktury kluczy publicznych (PKI) przy użyciu Easy-RSA.
3. Generacja certyfikatów dla serwera i klienta.
4. Konfiguracja plików konfiguracyjnych OpenVPN na serwerze i kliencie.
5. Testy połączeń oraz analiza dzienników logów w celu weryfikacji poprawności działania systemu.

I.5. Oczekiwane wyniki

Po zakończeniu projektu oczekuje się:

- Stabilnego połączenia VPN między serwerem a klientem z poprawnie uwierzytelnionymi certyfikatami.
- Przejrzystej dokumentacji zawierającej opisy kroków konfiguracji oraz wyniki testów.

Projekt ten stanowi podstawę do zrozumienia zaawansowanych aspektów bezpiecznych połączeń sieciowych oraz może być rozwinięty w przyszłości o dodatkowe funkcje, takie jak monitorowanie połączeń czy zarządzanie wieloma klientami w ramach jednego serwera VPN.

Rozdział II. Teoretyczne podstawy sieci VPN i kryptografii

II.1. Definicja i działanie sieci VPN

Wirtualna sieć prywatna (VPN) to technologia, która pozwala na bezpieczne przesyłanie danych przez publiczne sieci, tak jakby urządzenia były bezpośrednio podłączone do prywatnej sieci. Głównym celem VPN jest zapewnienie poufności, integralności i uwierzytelnienia przesyłanych danych. Użytkownicy, korzystając z tej technologii, mogą bezpiecznie łączyć się z sieciami prywatnymi, chronić swoje dane przed podsłuchiwaniem oraz ominąć ograniczenia geograficzne.

Mechanizmy tunelowania danych: Tunelowanie to technika polegająca na enkapsulacji pakietów danych w bezpiecznym tunelu tworzonym przez protokoł VPN. Pakiety te są szyfrowane przed przesłaniem, co uniemożliwia ich przechwycenie przez osoby nieupoważnione. Proces tunelowania odbywa się za pomocą protokołów, takich jak IPsec czy SSL/TLS.

Rodzaje sieci VPN:

- VPN Site-to-Site: Łączy całe sieci lokalne znajdujące się w różnych lokalizacjach. Jest stosowany głównie w firmach, gdzie biura w różnych miejscach mogą dzielić zasoby.
- VPN Remote Access: Umożliwia indywidualnym użytkownikom łączenie się z prywatną siecią za pomocą internetu, co jest przydatne dla pracowników zdalnych lub w sytuacjach, gdy konieczny jest dostęp do zasobów sieciowych spoza biura.

II.2. Protokoły i technologie szyfrowania stosowane w VPN

Protokoły VPN:

- SSL/TLS: Zapewnia bezpieczne połączenia poprzez szyfrowanie danych na poziomie aplikacji. Jest stosowany w połączeniach HTTPS oraz w niektórych konfiguracjach VPN.
- IPsec: Działa na poziomie warstwy sieciowej i oferuje kompleksowe zabezpieczenie poprzez szyfrowanie i uwierzytelnianie każdego pakietu danych. Jest często używany w sieciach VPN typu Site-to-Site.
- OpenVPN: Otwarta implementacja VPN wykorzystująca protokoły SSL/TLS do bezpiecznego tunelowania danych.

Typy szyfrowania:

- AES-256: Zaawansowany standard szyfrowania blokowego, zapewnia wysoki poziom bezpieczeństwa.
- HMAC: Mechanizm uwierzytelniania wiadomości, który gwarantuje integralność danych i chroni przed ich modyfikacją.

- SHA-256: Funkcja skrótu stosowana do zapewnienia integralności danych i uwierzytelniania przesyłanych informacji.

Rola certyfikatów i kluczy kryptograficznych: Certyfikaty służą do uwierzytelnienia stron komunikacji oraz do negocjacji kluczy szyfrowania, co umożliwia bezpieczne nawiązanie połączenia.

II.3. Bezpieczeństwo i zagrożenia w sieciach VPN

Typowe zagrożenia:

- Ataki typu man-in-the-middle (MITM): Polegają na przechwytywaniu danych przesyłanych pomiędzy dwoma stronami komunikacji.
- Podsłuchiwanie ruchu sieciowego: Możliwość przechwytywania danych przesyłanych w niezabezpieczonej sieci.
- Ataki DDoS: Skierowane na infrastrukturę sieciową w celu jej przeciążenia i uniemożliwienia działania VPN.

Metody przeciwdziałania:

- Używanie silnych mechanizmów szyfrowania, takich jak AES-256.
- Uwierzytelnianie dwuskładnikowe (2FA).
- Weryfikacja integralności danych za pomocą funkcji skrótu (np. SHA-256).

Rola integralności danych: Zapewnia, że dane przesyłane w tunelu VPN nie zostaną zmodyfikowane. Uwierzytelnienie końców komunikacji zapobiega podszywaniu się pod legalne urządzenia.

II.4. Infrastruktura klucza publicznego (PKI)

Czym jest PKI: PKI (Public Key Infrastructure) to system zarządzania kluczami publicznymi i certyfikatami cyfrowymi, który umożliwia bezpieczne przesyłanie informacji w sieciach. Klucz publiczny jest używany do szyfrowania danych, podczas gdy klucz prywatny służy do ich odszyfrowania.

Generowanie i zarządzanie certyfikatami:

- Certyfikat urzędu certyfikacji (CA) służy do podpisywania kluczy publicznych.
- Certyfikat serwera zapewnia uwierzytelnienie po stronie serwera.
- Certyfikaty klientów umożliwiają autoryzowanym użytkownikom nawiązywanie połączenia z siecią VPN.
- Proces generowania kluczy i certyfikatów obejmuje tworzenie infrastruktury PKI, podpisywanie kluczy oraz zarządzanie ich ważnością**.**

II.5. Zastosowania praktyczne VPN

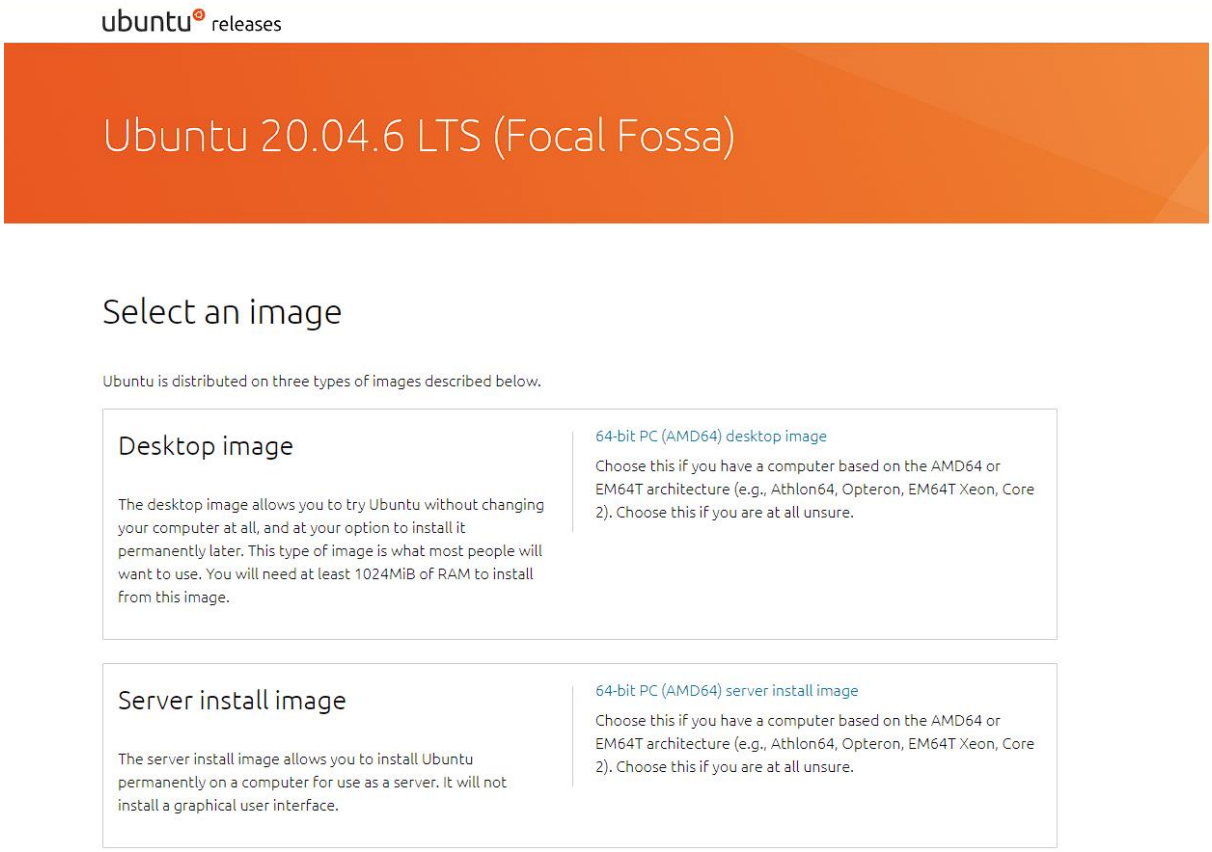
Bezpieczne połączenia firmowe: VPN jest szeroko stosowany w środowiskach biznesowych, gdzie zapewnia bezpieczny dostęp do zasobów sieciowych oraz umożliwia pracownikom bezpieczne połączenie z siecią firmową podczas pracy zdalnej.

Zdalny dostęp do zasobów sieciowych: Pracownicy mogą bezpiecznie łączyć się z bazami danych, plikami oraz innymi zasobami firmowymi niezależnie od lokalizacji.

Ochrona prywatności w Internecie: Użytkownicy indywidualni korzystają z VPN, aby ukryć swoje adresy IP, ominąć regionalne ograniczenia oraz zabezpieczyć połączenia w publicznych sieciach Wi-Fi.

Rozdział III. Instalacja i konfiguracja środowiska VPN

III.1. Przygotowanie środowiska serwera



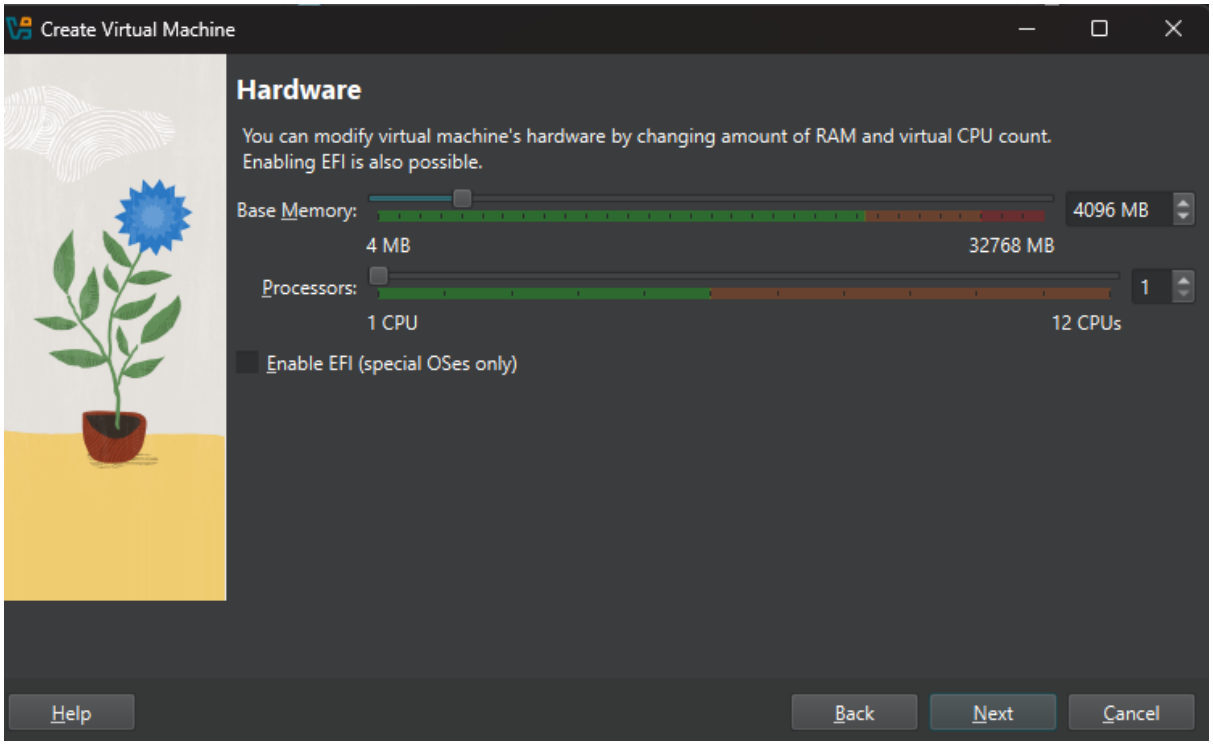
Obraz 1. Wybór odpowiedniego obrazu instalacyjnego Ubuntu 20.04.6 LTS

Obraz przedstawia stronę internetową z sekcją wyboru obrazu instalacyjnego dla systemu operacyjnego Ubuntu 20.04.6 LTS (Focal Fossa). Użytkownik ma do wyboru dwa główne typy obrazów instalacyjnych:

1. **Desktop image:**
    - Przeznaczony dla komputerów stacjonarnych i laptopów.
    - Pozwala na uruchomienie Ubuntu w trybie Live bez konieczności instalacji.
    - Może być później zainstalowany na stałe.
2. **Server install image:**
    - Przeznaczony do instalacji na komputerach, które będą działały jako serwery.
    - Nie posiada graficznego interfejsu użytkownika (CLI).
    - Zaprojektowany dla zaawansowanych użytkowników, takich jak administratorzy systemów.

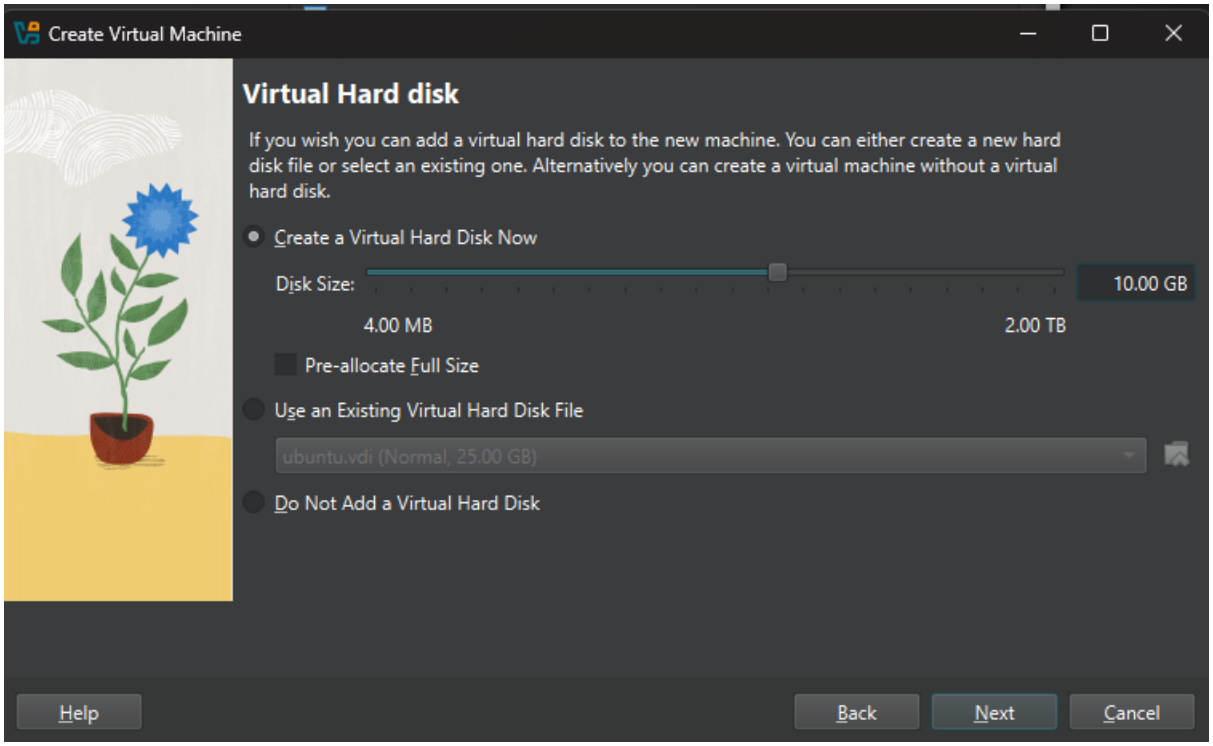
Obie opcje mają wersję dla architektury **64-bit PC (AMD64)**, co oznacza, że są przeznaczone dla komputerów opartych na nowoczesnych procesorach, takich jak Athlon64, Xeon, lub Core 2.

Obraz ten przedstawia krok wyboru odpowiedniego wariantu instalacyjnego w procesie pobierania Ubuntu, co jest istotnym elementem przygotowania środowiska serwera dla projektu.



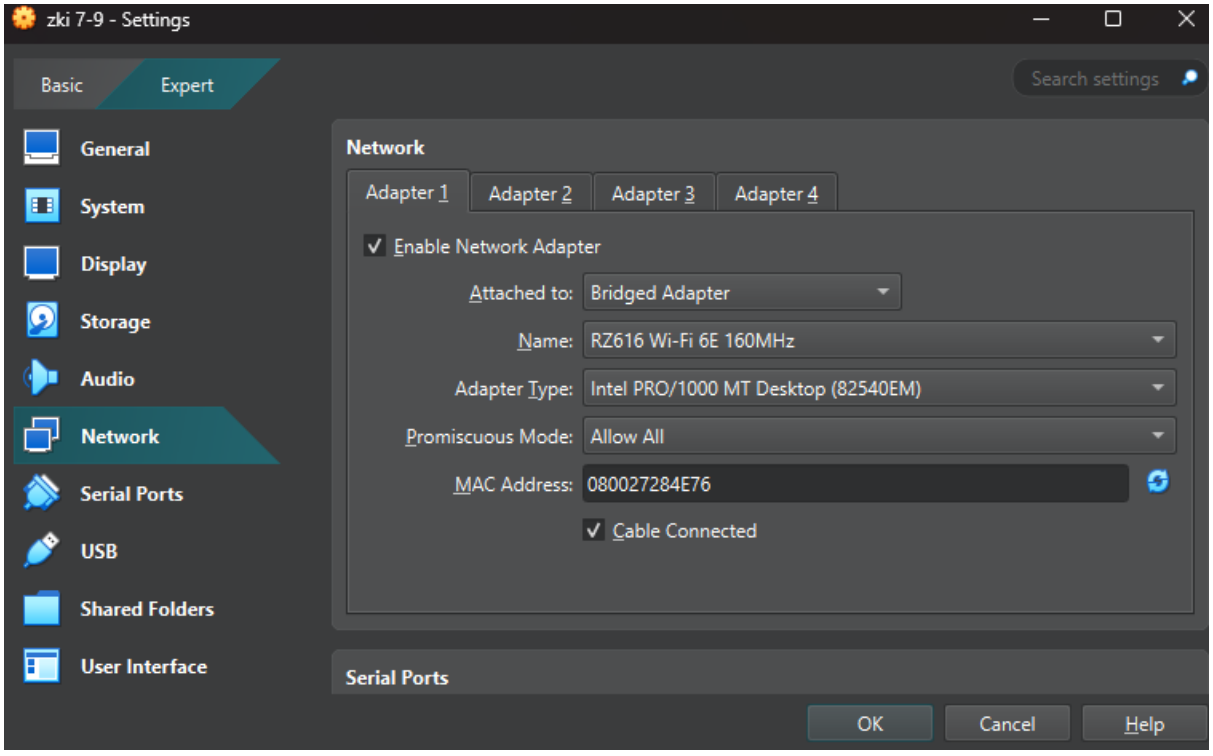
Obraz 2. Konfiguracja zasobów maszyny wirtualnej dla Ubuntu Server.

Ekran konfiguracji umożliwia dostosowanie ilości pamięci RAM (ustawionej na 4096 MB) oraz liczby wirtualnych procesorów (1 CPU) do wymagań systemu. Możliwość włączenia EFI (Extensible Firmware Interface) jest dostępna, ale zalecana tylko dla specjalnych systemów operacyjnych. Właściwe dobranie zasobów zapewnia optymalne działanie środowiska serwera podczas realizacji projektu.



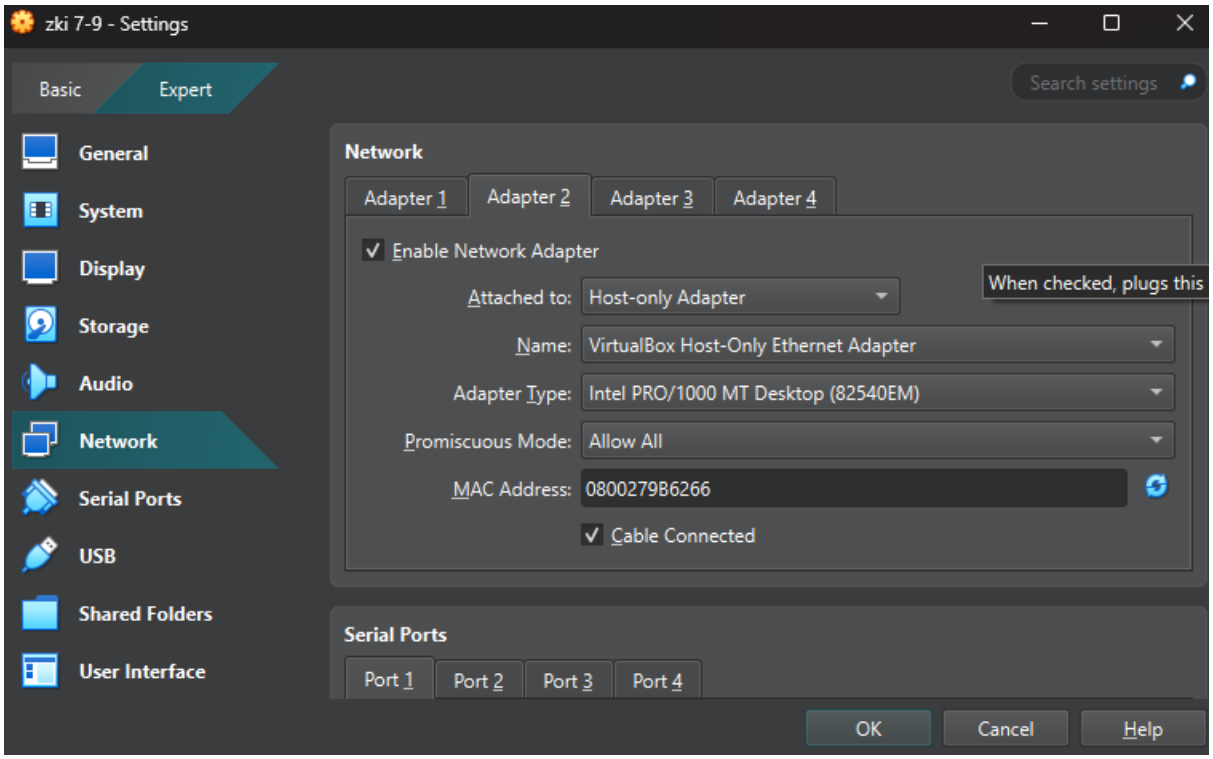
Obraz 3. Tworzenie wirtualnego dysku twardego dla maszyny Ubuntu.

Użytkownik wybiera opcję utworzenia nowego wirtualnego dysku twardego o rozmiarze 10 GB. Istnieje możliwość zmiany rozmiaru dysku w zakresie od 4 MB do 2 TB. Opcja „Pre-allocate Full Size" jest wyłączona, co oznacza dynamiczne przydzielanie miejsca na dysku w miarę potrzeby. Odpowiednia wielkość dysku zapewnia wystarczającą przestrzeń na system operacyjny i dodatkowe oprogramowanie potrzebne do konfiguracji serwera.



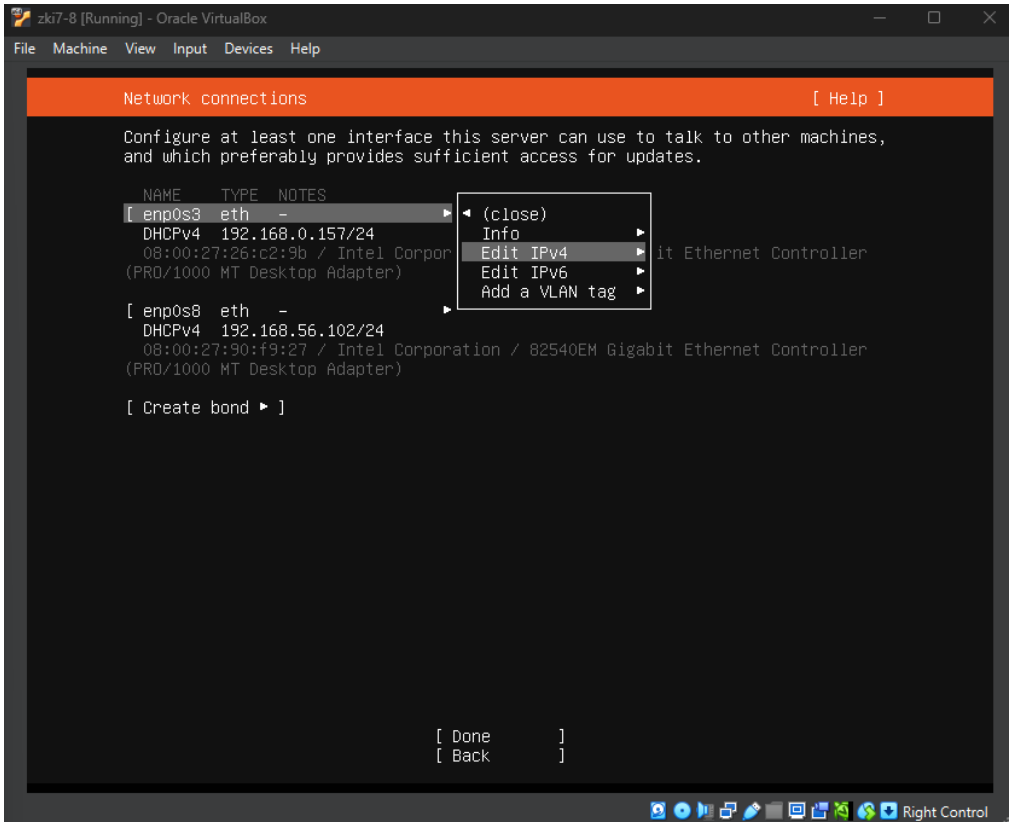
Obraz 4. Ustawienia adaptera sieciowego w trybie "Bridged Adapter".

Pierwszy adapter sieciowy jest skonfigurowany w trybie "Bridged Adapter", co oznacza, że wirtualna maszyna będzie działać jak fizyczne urządzenie w tej samej sieci co host. Interfejs sieciowy jest przypisany do fizycznego adaptera Wi-Fi (RZ616 Wi-Fi 6E). Ustawienie "Allow All" w trybie nasłuchiwania (Promiscuous Mode) umożliwia przechwytywanie całego ruchu sieciowego, co może być przydatne do monitorowania sieci i analizy bezpieczeństwa

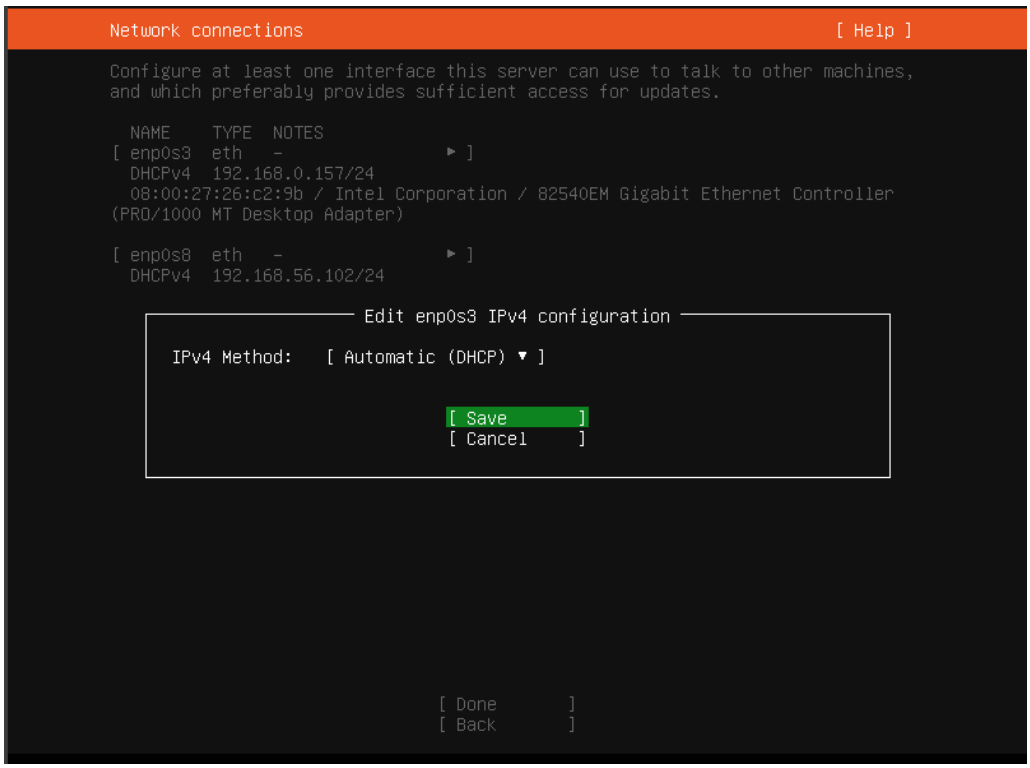


Obraz 5. Ustawienia adaptera sieciowego w trybie "Host-only Adapter".

Drugi adapter sieciowy jest skonfigurowany jako "Host-only Adapter", co pozwala na komunikację między maszyną wirtualną a hostem bez dostępu do sieci zewnętrznej. Wirtualny adapter sieciowy "VirtualBox Host-Only Ethernet Adapter" umożliwia bezpośrednie połączenie do testowania usług serwera, takich jak OpenVPN, w odizolowanym środowisku. Dzięki temu można bezpiecznie przeprowadzać testy bez ryzyka zakłócenia rzeczywistej sieci.

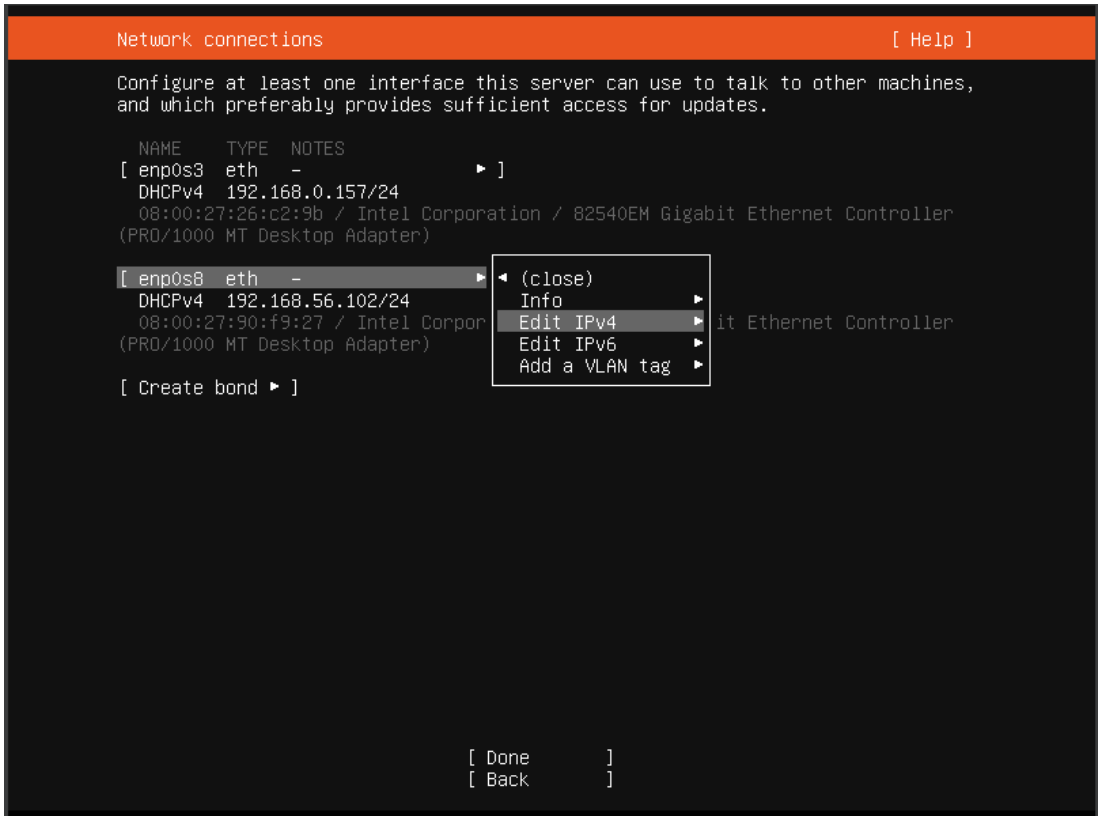


Obraz 6. Rozpoczęcie konfiguracji adresu IPv4 dla interfejsu enp0s8

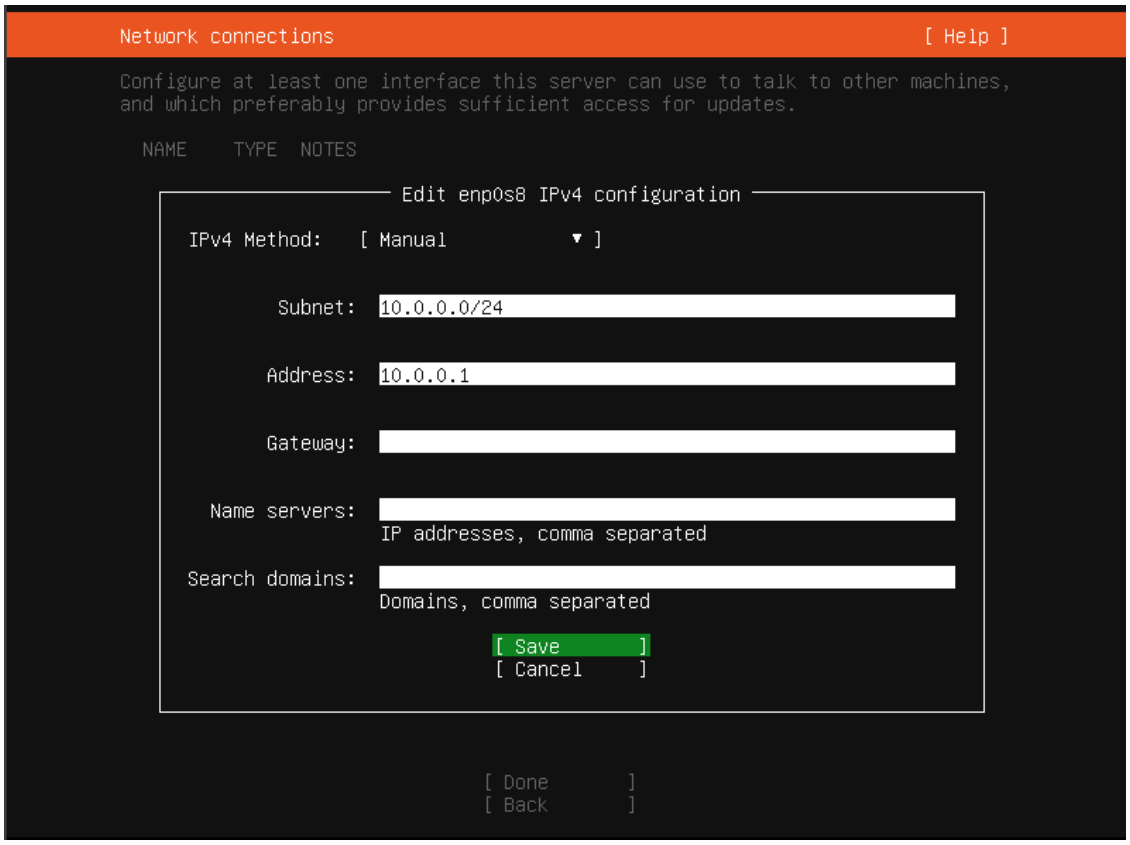


Obraz 7. Konfiguracja adresu IPv4 dla interfejsu enp0s8

Interfejs enp0s3 został skonfigurowany jako karta typu "Bridged" z automatycznym przydzielaniem adresu IP poprzez DHCP (192.168.0.157/24). Pozwala to maszynie wirtualnej na bezpośrednią komunikację w tej samej sieci co host.

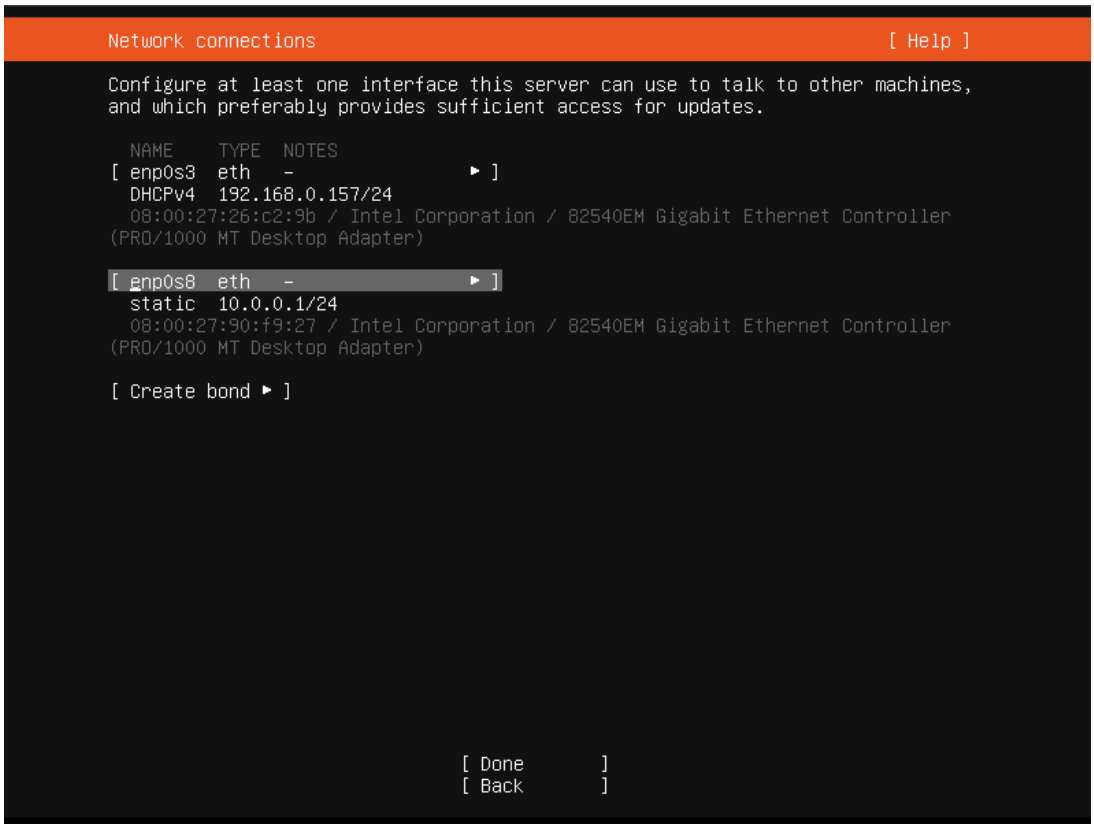


Obraz 8. Konfiguracja interfejsu sieciowego enp0s8 jako sieci wewnętrznej.



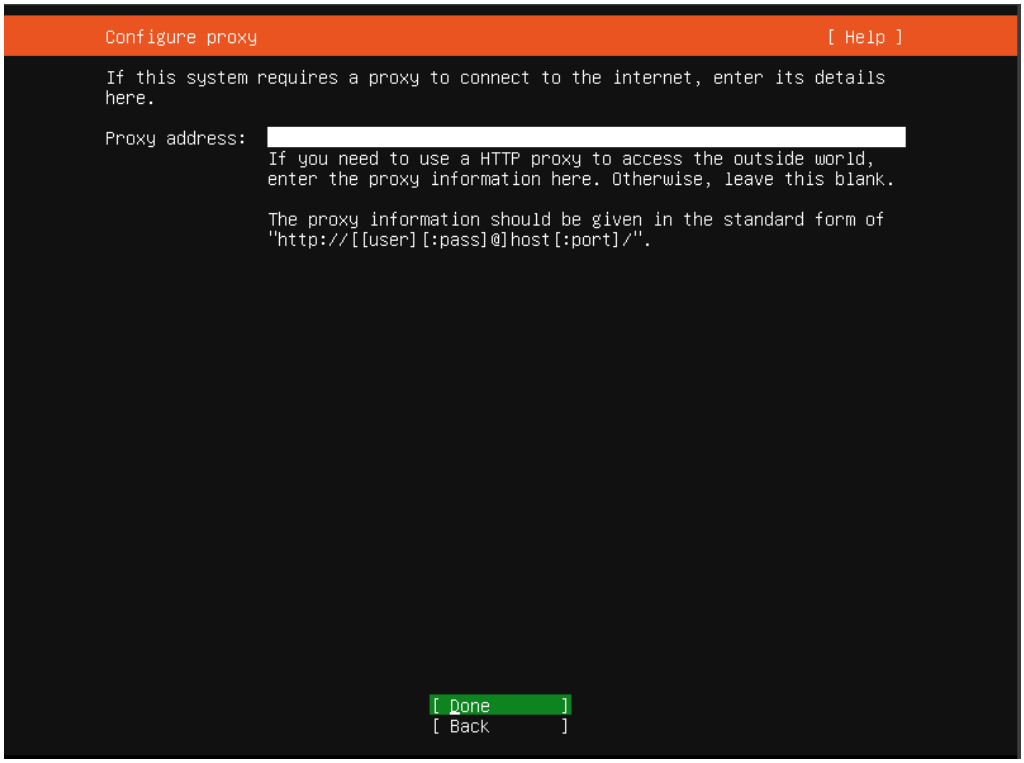
Obraz 9. Ręczna konfiguracja adresu IP dla sieci wewnętrznej enp0s8

Dla interfejsu enp0s8 skonfigurowano sieć wewnętrzną z podsiecią 10.0.0.0/24 i statycznym adresem IP 10.0.0.1. To ustawienie umożliwia izolowaną komunikację wewnętrzną, co jest kluczowe w środowisku testowym, np. przy konfiguracji serwera OpenVPN.



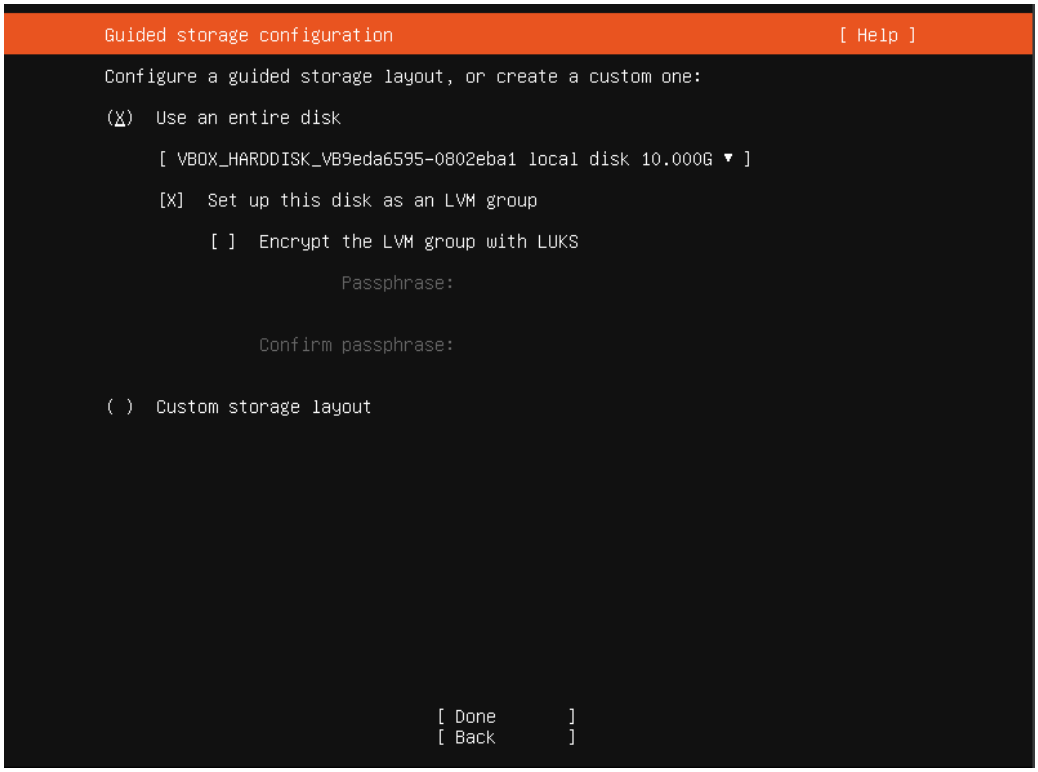
Obraz 10. Zakończona konfiguracja interfejsów sieciowych na serwerze.

Interfejs enp0s3 działa w trybie "Bridged Adapter" z automatycznie przydzielonym adresem IP 192.168.0.157/24, zapewniając dostęp do internetu. Interfejs enp0s8 został skonfigurowany jako statyczny z adresem IP 10.0.0.1/24, tworząc wewnętrzną sieć izolowaną, idealną do testowania usług serwera, takich jak OpenVPN.

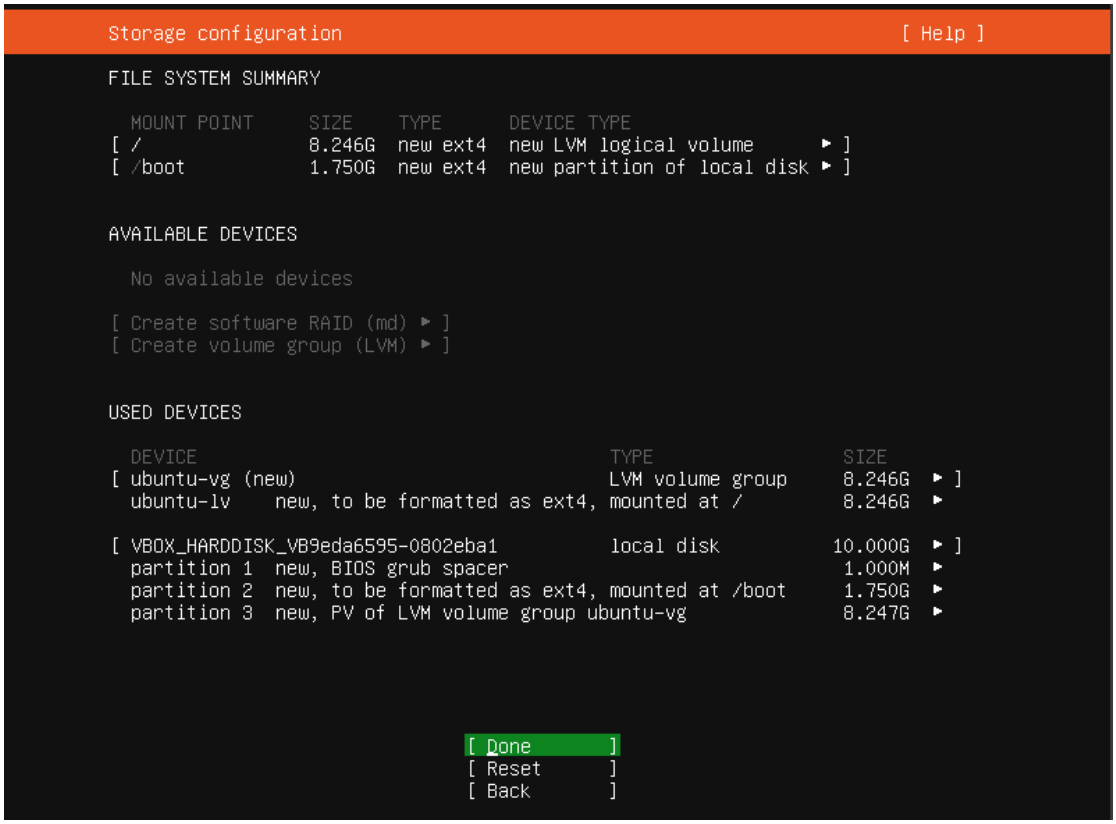


Obraz 11. Konfiguracja połączenia bez użycia serwera proxy.

Ekran konfiguracji serwera proxy został pozostawiony pusty, co oznacza bezpośredni dostęp do internetu bez pośrednictwa dodatkowych serwerów. W takim przypadku system będzie łączyć się z siecią zewnętrzną bez żadnych ograniczeń, co jest standardowym wyborem w większości środowisk testowych.

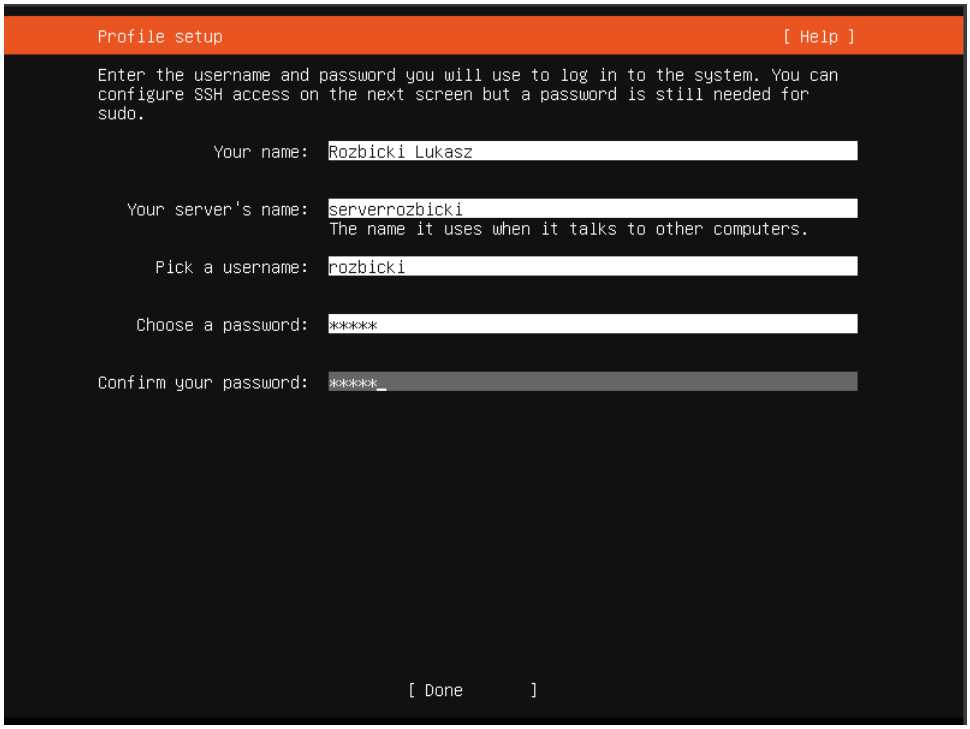


Obraz 12. Automatyczne tworzenie partycji z wykorzystaniem całego dysku.



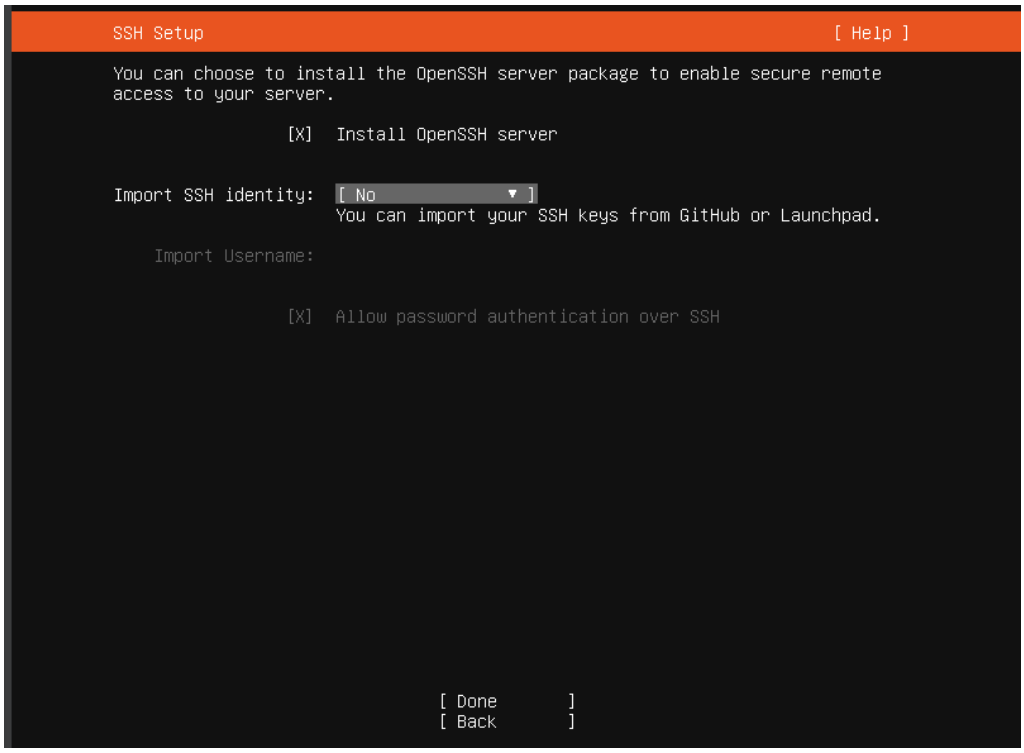
Obraz 13. Podsumowanie konfiguracji systemu plików i partycji.

Automatycznie utworzono partycje: główną partycję systemową / o rozmiarze 8,246 GB oraz partycję rozruchową /boot o rozmiarze 1,75 GB. Partycje sformatowano w systemie plików ext4, co zapewnia stabilność i efektywność pracy systemu. LVM został skonfigurowany dla dynamicznego zarządzania przestrzenią dyskową.



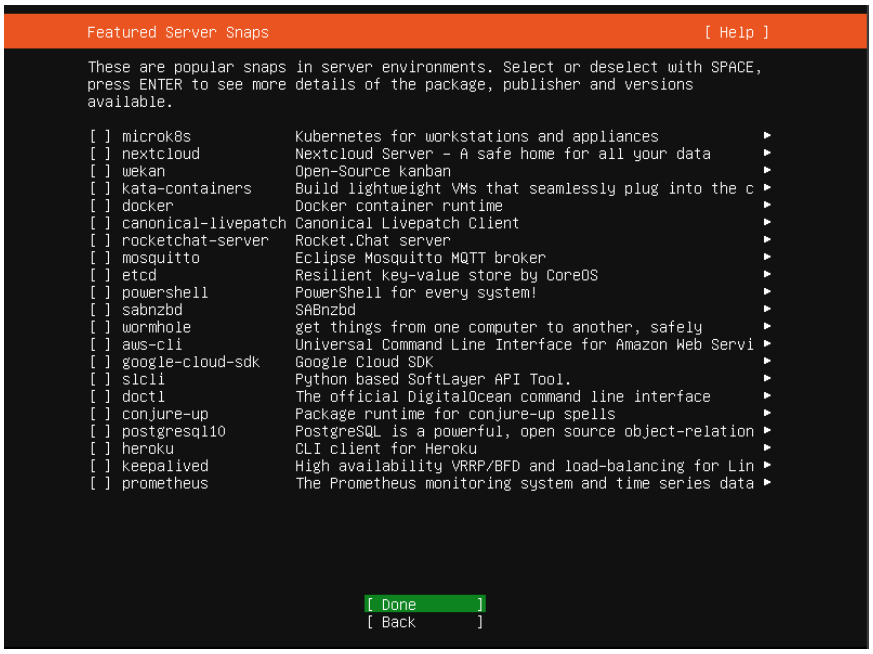
Obraz 14. Tworzenie profilu użytkownika i konfiguracja serwera.

Wprowadzono dane użytkownika i serwera podczas instalacji Ubuntu. Nazwa serwera to serverrozbicki, a nazwa użytkownika to rozbicki. Użytkownik zdefiniował silne hasło do późniejszego użycia podczas logowania i wykonywania komend z uprawnieniami administratora (sudo). Te ustawienia są kluczowe dla bezpieczeństwa i identyfikacji w sieci.



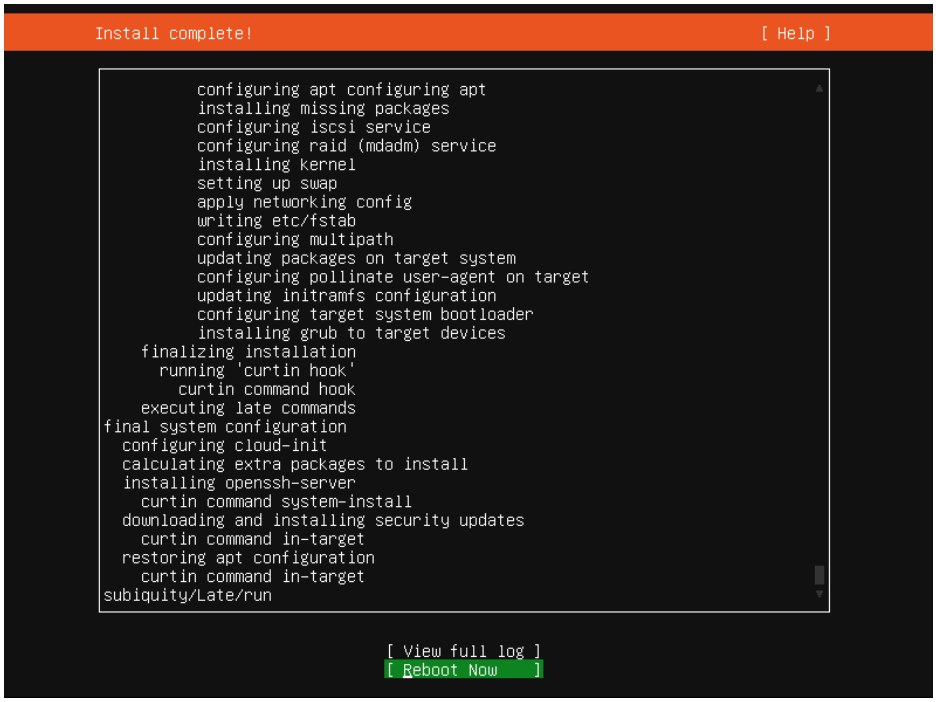
Obraz 15. Instalacja serwera OpenSSH i konfiguracja dostępu zdalnego.

Włączono instalację serwera OpenSSH, co umożliwia zdalne logowanie do maszyny za pomocą protokołu SSH. Import kluczy SSH został pominięty, a uwierzytelnianie hasłem jest dozwolone. To ustawienie jest kluczowe do zarządzania serwerem bez fizycznego dostępu do niego, zapewniając wygodę i bezpieczeństwo administracji.



Obraz 16. Pominięto instalację dodatkowych usług serwerowych.

Lista dostępnych pakietów serwerowych typu Snap, takich jak Docker, Nextcloud, PostgreSQL czy Prometheus, została pozostawiona bez wyboru. Użytkownik zdecydował się na minimalistyczną instalację, ograniczając dodatkowe oprogramowanie do niezbędnego minimum, co jest korzystne dla lepszej kontroli nad konfiguracją systemu i wydajnością.



Obraz 17. Zakończenie instalacji systemu i przygotowanie do restartu.

Instalacja Ubuntu Server została pomyślnie zakończona, obejmując konfigurację podstawowych usług, instalację jądra systemu, serwera OpenSSH oraz aktualizacje zabezpieczeń. Użytkownik może teraz zrestartować system, aby zakończyć proces i przejść do logowania na nowo skonfigurowanym serwerze.

Obraz 18. Ekran logowania do nowo zainstalowanego serwera Ubuntu.

System uruchomił się pomyślnie po instalacji, wyświetlając klucze hosta SSH oraz informacje inicjalizacyjne. Użytkownik rozbicki może teraz zalogować się, wpisując swoje wcześniej skonfigurowane dane logowania. Serwer jest gotowy do dalszej konfiguracji, np. instalacji OpenVPN lub innych usług sieciowych.

```
  System information as of Mon 13 Jan 2025 06:09:32 PM UTC

  System load:             0.14
  Usage of /:              30.1% of 8.02GB
  Memory usage:            4%
  Swap usage:              0%
  Processes:               99
  Users logged in:         0
  IPv4 address for enp0s3: 192.168.0.157
  IPv6 address for enp0s3: 2a02:a31d:e0ec:c200:a00:27ff:fe26:c29b
  IPv4 address for enp0s8: 10.0.0.1

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

rozbicki@serverrozbicki:~$ sudo touch /etc/cloud/cloud-init.disabled
[sudo] password for rozbicki:
rozbicki@serverrozbicki:~$ reboot
```

Obraz 19. Wyłączenie inicjowania pakietu Cloud-Init i restart systemu.

Użytkownik rozbicki wykonał polecenie sudo touch /etc/cloud/cloud-init.disabled, aby wyłączyć pakiet Cloud-Init, który jest używany do inicjalizacji systemu w chmurze. Po zakończeniu tej operacji został wykonany restart systemu (reboot), aby zmiany zaczęły obowiązywać. Wyłączenie Cloud-Init przyspiesza start serwera w środowiskach lokalnych, gdzie jego funkcje nie są potrzebne.

```
serverrozbicki login: rozbicki
Password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-204-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon 13 Jan 2025 06:12:34 PM UTC

  System load:             0.41
  Usage of /:              30.2% of 8.02GB
  Memory usage:            4%
  Swap usage:              0%
  Processes:               104
  Users logged in:         0
  IPv4 address for enp0s3: 192.168.0.157
  IPv6 address for enp0s3: 2a02:a31d:e0ec:c200:a00:27ff:fe26:c29b
  IPv4 address for enp0s8: 10.0.0.1


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Mon Jan 13 18:09:33 UTC 2025 on tty1
rozbicki@serverrozbicki:~$ sudo apt upgrade
```

Obraz 20. Aktualizacja systemu za pomocą sudo apt update i sudo apt upgrade

Po zalogowaniu na serwer Ubuntu użytkownik rozbicki uruchomił komendy sudo apt update oraz sudo apt upgrade, aby zaktualizować listę pakietów w repozytoriach i zainstalować najnowsze wersje oprogramowania. Taka procedura zapewnia bezpieczeństwo systemu oraz dostęp do najnowszych funkcji i poprawek błędów.

```
rozbicki@serverrozbicki:~$ sudo ufw allow OpenSSH
Rules updated
Rules updated (v6)
rozbicki@serverrozbicki:~$ ufw enable
ERROR: You need to be root to run this script
rozbicki@serverrozbicki:~$ sudo ufw enable
Firewall is active and enabled on system startup
rozbicki@serverrozbicki:~$ _
```

Obraz 21. Włączenie zapory sieciowej UFW i zezwolenie na połączenia SSH.

Za pomocą polecenia sudo ufw allow OpenSSH użytkownik zezwolił na połączenia zdalne poprzez protokół SSH. Następnie polecenie sudo ufw enable włączyło zaporę sieciową UFW (Uncomplicated Firewall), która jest teraz aktywna i będzie automatycznie uruchamiana przy starcie systemu. Dzięki temu serwer jest chroniony przed nieautoryzowanymi próbami dostępu.

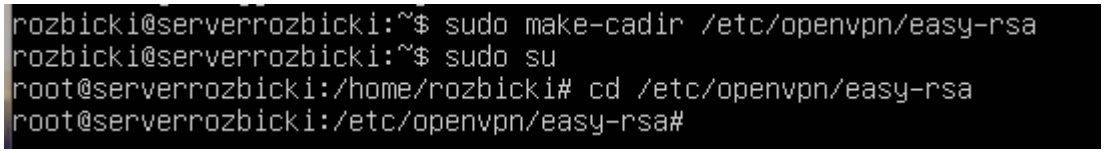
III.2. Instalacja i konfiguracja OpenVPN.

Instalacja OpenVPN i pakietu easy-rsa:

```
Unpacking pcscd (1.8.26-3) ...
Selecting previously unselected package libpkcs11-helper1:amd64.
Preparing to unpack .../3-libpkcs11-helper1_1.26-1_amd64.deb ...
Unpacking libpkcs11-helper1:amd64 (1.26-1) ...
Selecting previously unselected package opensc-pkcs11:amd64.
Preparing to unpack .../4-opensc-pkcs11_0.20.0-3_amd64.deb ...
Unpacking opensc-pkcs11:amd64 (0.20.0-3) ...
Selecting previously unselected package opensc.
Preparing to unpack .../5-opensc_0.20.0-3_amd64.deb ...
Unpacking opensc (0.20.0-3) ...
Selecting previously unselected package openvpn.
Preparing to unpack .../6-openvpn_2.4.12-0ubuntu0.20.04.2_amd64.deb ...
Unpacking openvpn (2.4.12-0ubuntu0.20.04.2) ...
Selecting previously unselected package easy-rsa.
Preparing to unpack .../7-easy-rsa_3.0.6-1_all.deb ...
Unpacking easy-rsa (3.0.6-1) ...
Setting up libccid (1.4.31-1) ...
Setting up libpkcs11-helper1:amd64 (1.26-1) ...
Setting up opensc-pkcs11:amd64 (0.20.0-3) ...
Setting up libpcsclite1:amd64 (1.8.26-3) ...
Setting up easy-rsa (3.0.6-1) ...
Setting up openvpn (2.4.12-0ubuntu0.20.04.2) ...
 * Restarting virtual private network daemon.                                    [ OK ]
Created symlink /etc/systemd/system/multi-user.target.wants/openvpn.service → /lib/systemd/system/op
envpn.service.
Setting up opensc (0.20.0-3) ...
Setting up pcscd (1.8.26-3) ...
Created symlink /etc/systemd/system/sockets.target.wants/pcscd.socket → /lib/systemd/system/pcscd.so
cket.
pcscd.service is a disabled or a static unit, not starting it.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
rozbicki@serverrozbicki:~$
```

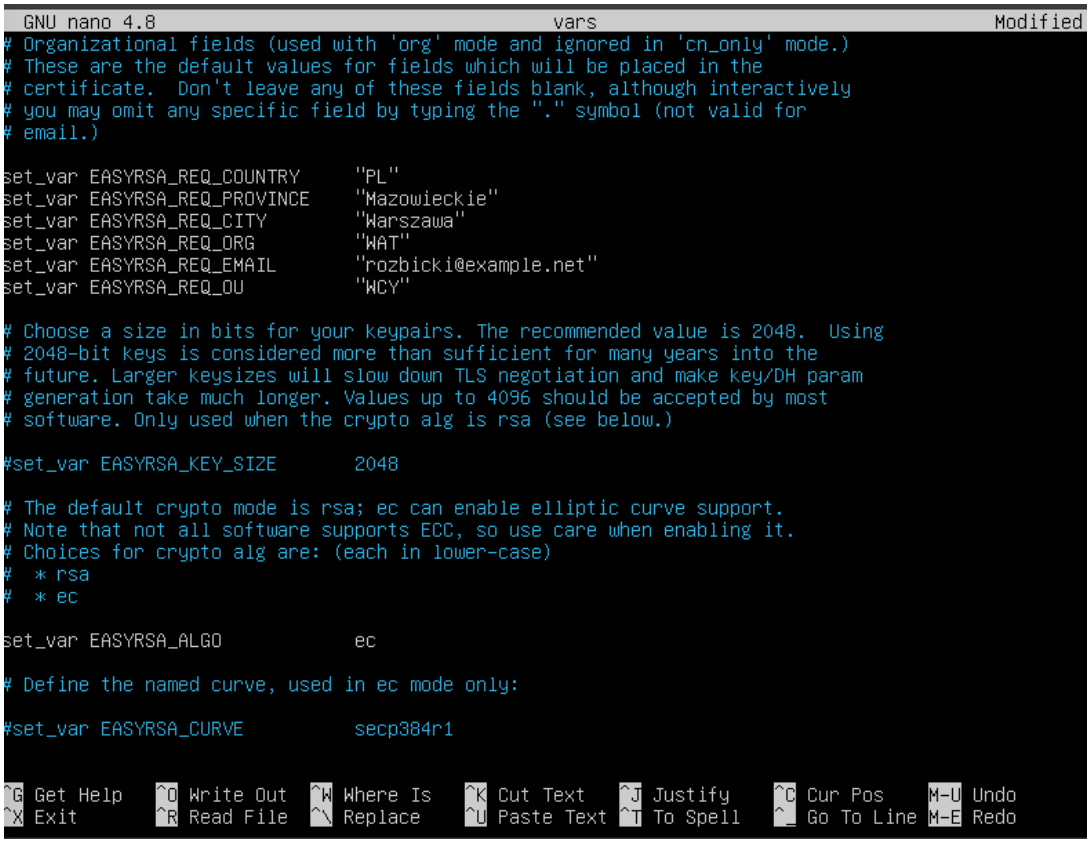
Obraz 22. Instalacja pakietów OpenVPN i Easy-RSA.

Użytkownik wykonał polecenie sudo apt install openvpn easy-rsa, co zainstalowało OpenVPN - oprogramowanie do zarządzania wirtualnymi sieciami prywatnymi, oraz Easy-RSA - narzędzie do zarządzania infrastrukturą kluczy publicznych (PKI). Proces instalacji zakończył się pomyślnie, przygotowując serwer do konfiguracji połączeń VPN i generowania certyfikatów zabezpieczających.



Obraz 23. Tworzenie katalogu dla infrastruktury CA za pomocą Easy-RSA.

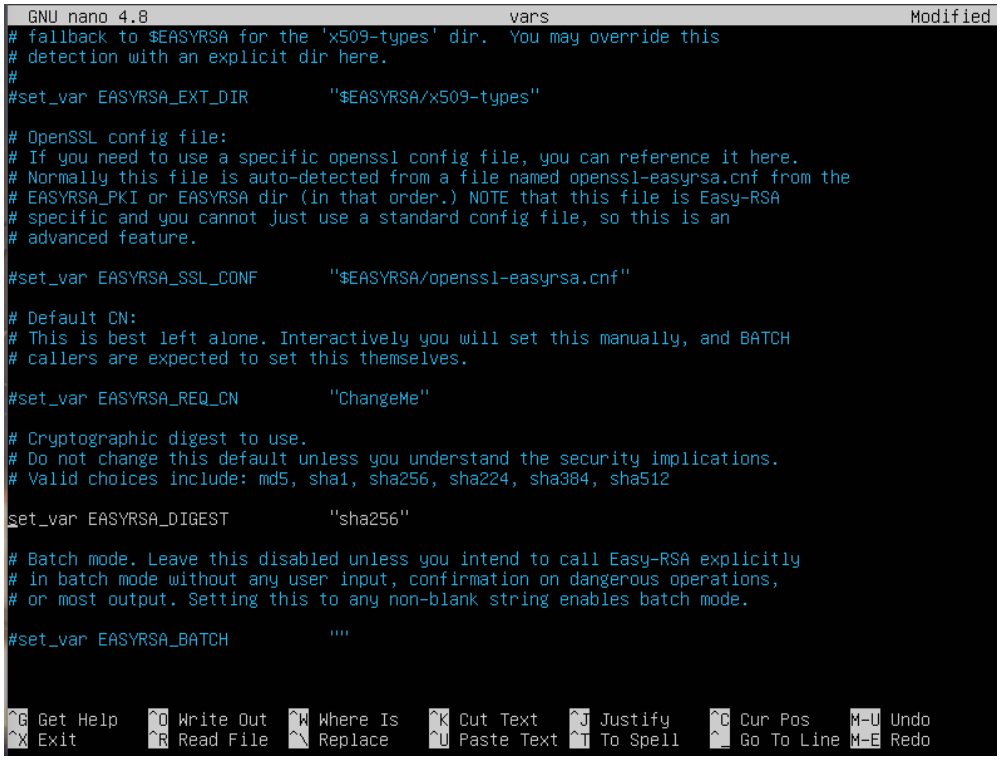
Użytkownik wykonał polecenie sudo make-cadir /etc/openvpn/easy-rsa, które utworzyło dedykowany katalog do przechowywania plików związanych z infrastrukturą klucza publicznego (CA - Certificate Authority). Następnie przełączono się na konto superużytkownika (sudo su) i zmieniono bieżący katalog na /etc/openvpn/easy-rsa. Ten krok jest niezbędny do rozpoczęcia procesu generowania kluczy i certyfikatów potrzebnych do bezpiecznego działania OpenVPN.



Obraz 24. Edycja pliku /etc/openvpn/easy-rsa/vars dla generowania certyfikatów..

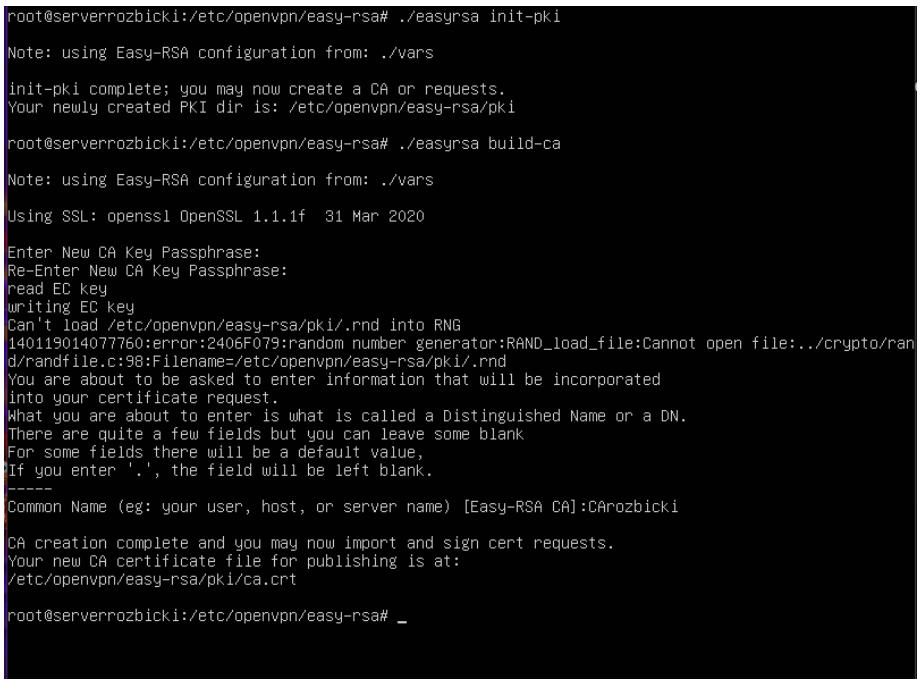
W pliku vars skonfigurowano podstawowe parametry dla infrastruktury kluczy publicznych (PKI). Ustawiono między innymi lokalizację (Polska, Mazowieckie, Warszawa), nazwę organizacji (WAT), oraz e-mail kontaktowy rozbicki@example.net. Wybrano algorytm

krzywych eliptycznych (ECC) do generowania kluczy, co zapewnia wysoki poziom bezpieczeństwa przy mniejszej długości klucza niż w przypadku RSA.



Obraz 25. Kontynuacja konfiguracji parametrów bezpieczeństwa w pliku vars

W dalszej części pliku vars skonfigurowano kluczowe opcje bezpieczeństwa, takie jak algorytm skrótu (sha256) używany w podpisywaniu certyfikatów oraz lokalizację plików konfiguracyjnych OpenSSL. Ustawienia te są kluczowe do zapewnienia bezpiecznej i poprawnej pracy serwera VPN opartego na OpenVPN.



Obraz 26. Tworzenie infrastruktury PKI i budowa urzędu certyfikacji (CA).

Polecenie ./easyrsa init-pki zainicjowało infrastrukturę PKI, tworząc niezbędne katalogi i pliki konfiguracyjne do zarządzania kluczami i certyfikatami. Następnie polecenie ./easyrsa build-ca utworzyło urząd certyfikacji (CA), który będzie podpisywał certyfikaty dla serwera i klientów OpenVPN. Wprowadzono nazwę wspólną (CN) CAnazwisko, co identyfikuje CA w procesie uwierzytelniania. Certyfikat CA został zapisany w katalogu /etc/openvpn/easy-rsa/pki/ca.crt.

```
root@serverrozbicki:/etc/openvpn/easy-rsa# ./easyrsa gen-req serverrozbicki nopass

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.1.1f  31 Mar 2020
Generating an EC private key
writing new private key to '/etc/openvpn/easy-rsa/pki/private/serverrozbicki.key.WStUimDk17'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [serverrozbicki]:serverrozbicki

Keypair and certificate request completed. Your files are:
req: /etc/openvpn/easy-rsa/pki/reqs/serverrozbicki.req
key: /etc/openvpn/easy-rsa/pki/private/serverrozbicki.key

root@serverrozbicki:/etc/openvpn/easy-rsa# _
```

Obraz 27. Generowanie klucza prywatnego i żądania certyfikatu dla serwera OpenVPN

Wykonano polecenie ./easyrsa gen-req servernazwisko nopass, które wygenerowało klucz prywatny serwera oraz plik żądania certyfikatu (CSR) z nazwą wspólną (CN) servernazwisko. Klucz prywatny został zapisany w katalogu /etc/openvpn/easy-rsa/pki/private/serverrozbicki.key, a żądanie certyfikatu w pliku /etc/openvpn/easy-rsa/pki/reqs/serverrozbicki.req. Te pliki są niezbędne do podpisania certyfikatu przez CA i zabezpieczenia komunikacji serwera.

```
root@serverrozbicki:/etc/openvpn/easy-rsa# ./easyrsa sign-req server serverrozbicki

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.1.1f  31 Mar 2020


You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 1080 days:

subject=
    commonName                = serverrozbicki


Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /etc/openvpn/easy-rsa/pki/safessl-easyrsa.cnf
Enter pass phrase for /etc/openvpn/easy-rsa/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'serverrozbicki'
Certificate is to be certified until Dec 29 18:32:52 2027 GMT (1080 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /etc/openvpn/easy-rsa/pki/issued/serverrozbicki.crt

root@serverrozbicki:/etc/openvpn/easy-rsa# _
```

Obraz 28. Podpisanie żądania certyfikatu serwera przez urząd certyfikacji (CA).

Wykonano polecenie ./easyrsa sign-req server servernazwisko, które podpisało żądanie certyfikatu dla serwera OpenVPN. Proces wymagał potwierdzenia poprawności danych oraz podania hasła do klucza prywatnego CA. Certyfikat serwera został pomyślnie wygenerowany i zapisany w katalogu /etc/openvpn/easy-rsa/pki/issued/serverrozbicki.crt. Certyfikat ten jest ważny przez 1080 dni i będzie używany do uwierzytelniania serwera w sieci VPN.

```
root@serverrozbicki:/etc/openvpn/easy-rsa# mkdir -p /etc/openvpn/server
root@serverrozbicki:/etc/openvpn/easy-rsa# cp /etc/openvpn/easy-rsa/pki/ca.crt  /etc/openvpn/server
root@serverrozbicki:/etc/openvpn/easy-rsa# cp /etc/openvpn/easy-rsa/pki/reqs/serverrozbicki.req /etc
/openvpn/server
root@serverrozbicki:/etc/openvpn/easy-rsa# cp /etc/openvpn/easy-rsa/pki/private/serverrozbicki.key /
etc/openvpn/server
root@serverrozbicki:/etc/openvpn/easy-rsa# cp /etc/openvpn/easy-rsa/pki/issued/serverrozbicki.crt /e
tc/openvpn/server
root@serverrozbicki:/etc/openvpn/easy-rsa# _
```

Obraz 29. Kopiowanie certyfikatów i kluczy do katalogu serwera OpenVPN.

Utworzono katalog /etc/openvpn/server, a następnie skopiowano do niego wymagane pliki:

- Certyfikat urzędu certyfikacji ca.crt
- Żądanie certyfikatu serwera serverrozbicki.req
- Klucz prywatny serwera serverrozbicki.key
- Certyfikat serwera serverrozbicki.crt

```
root@serverrozbicki:/etc/openvpn/easy-rsa# cd /etc/openvpn/client
root@serverrozbicki:/etc/openvpn/client# mkdir keys
root@serverrozbicki:/etc/openvpn/client#
```

Obraz 30. Tworzenie struktury katalogów dla kluczy i certyfikatów klienta OpenVPN.

Przełączono się do katalogu /etc/openvpn/client, a następnie utworzono podkatalog keys za pomocą polecenia mkdir keys. Katalog keys będzie przechowywał klucze prywatne i certyfikaty niezbędne dla klienta VPN do nawiązywania bezpiecznego połączenia z serwerem OpenVPN.

```
root@serverrozbicki:/etc/openvpn/easy-rsa# cd /etc/openvpn/client
root@serverrozbicki:/etc/openvpn/client# mkdir keys
root@serverrozbicki:/etc/openvpn/client# cd ..
root@serverrozbicki:/etc/openvpn# chmod -R 700 client
root@serverrozbicki:/etc/openvpn#
```

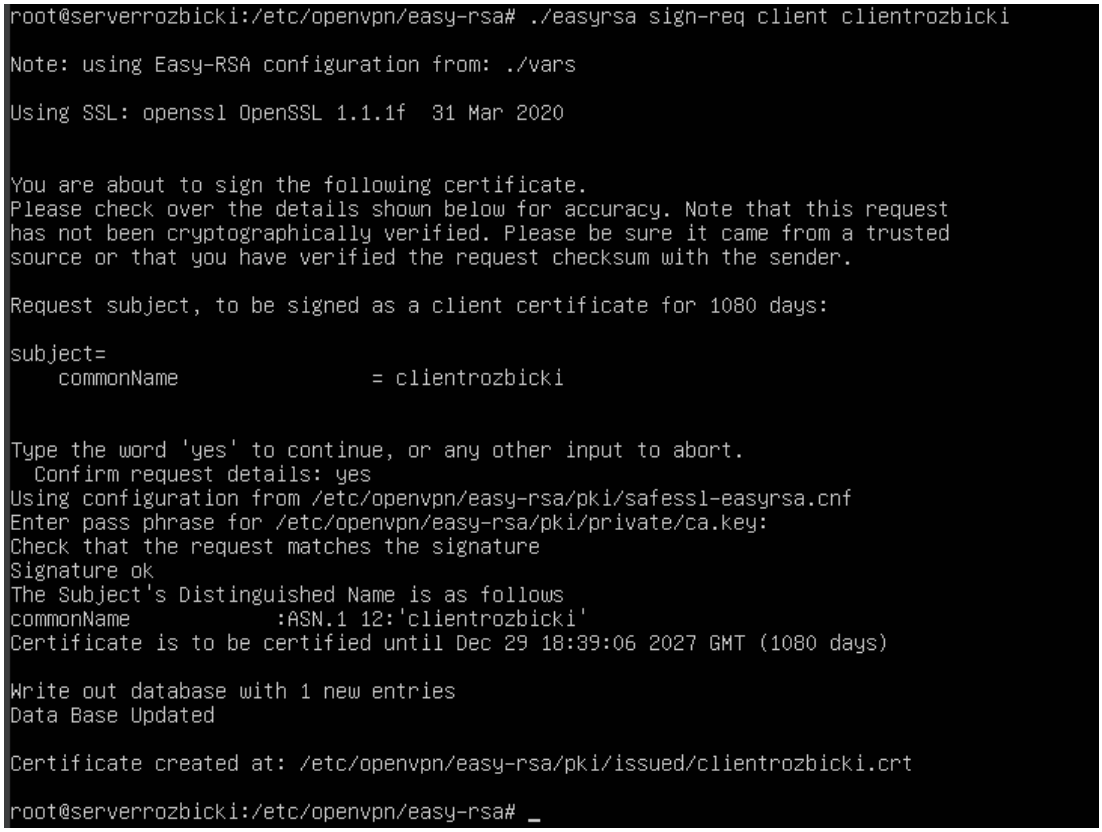
Obraz 31. Nadanie praw dostępu dla katalogu klienta i jego zawartości.

Wykonano polecenie chmod -R 700 client, które przypisało katalogowi /etc/openvpn/client, jego podkatalogom oraz plikom prawa dostępu tylko dla właściciela. Właściciel ma pełne prawa do odczytu, zapisu i wykonywania, podczas gdy inni użytkownicy nie mają żadnego dostępu.

```
root@serverrozbicki:/etc/openvpn# cd /etc/openvpn/easy-rsa
root@serverrozbicki:/etc/openvpn/easy-rsa# ./easyrsa gen-req clientrozbicki nopass

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.1.1f  31 Mar 2020
Generating an EC private key
writing new private key to '/etc/openvpn/easy-rsa/pki/private/clientrozbicki.key.bRWEYYEkod'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [clientrozbicki]:clientrozbicki

Keypair and certificate request completed. Your files are:
req: /etc/openvpn/easy-rsa/pki/reqs/clientrozbicki.req
key: /etc/openvpn/easy-rsa/pki/private/clientrozbicki.key

root@serverrozbicki:/etc/openvpn/easy-rsa# _
```

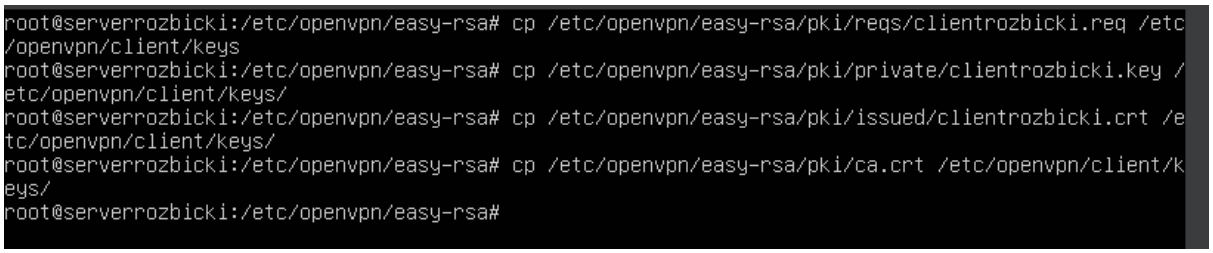
Obraz 32. Generowanie klucza prywatnego i żądania certyfikatu dla klienta OpenVPN.

Wykonano polecenie ./easyrsa gen-req clientrozbicki nopass, które wygenerowało klucz prywatny i plik żądania certyfikatu (CSR) dla klienta OpenVPN z nazwą wspólną (CN) clientnazwisko. Klucz prywatny został zapisany w katalogu /etc/openvpn/easy-rsa/pki/private/clientrozbicki.key, a plik żądania certyfikatu w /etc/openvpn/easy-rsa/pki/reqs/clientrozbicki.req. Te pliki są niezbędne do podpisania certyfikatu przez urząd



Obraz 33. Podpisanie żądania certyfikatu klienta przez urząd certyfikacji (CA).

Wykonano polecenie ./easyrsa sign-req client clientrozbicki, które podpisało żądanie certyfikatu klienta OpenVPN. Proces wymagał potwierdzenia poprawności danych i podania hasła do klucza prywatnego CA. Certyfikat klienta został wygenerowany i zapisany w katalogu /etc/openvpn/easy-rsa/pki/issued/clientrozbicki.crt. Certyfikat ten jest ważny przez 1080 dni i pozwala klientowi na bezpieczne łączenie się z serwerem OpenVPN.

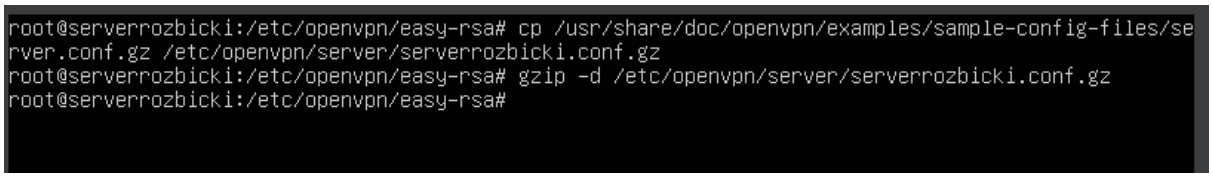


Obraz 34. Kopiowanie kluczy i certyfikatów klienta oraz CA do katalogu klienta OpenVPN.

Wszystkie wymagane pliki zostały skopiowane do katalogu /etc/openvpn/client/keys:

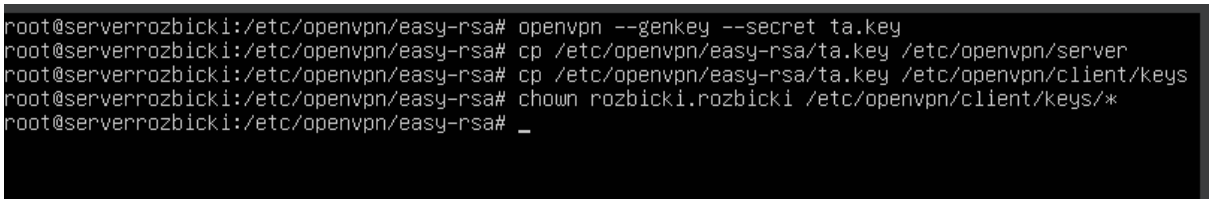
- Żądanie certyfikatu klienta clientrozbicki.req
- Klucz prywatny klienta clientrozbicki.key
- Certyfikat klienta clientrozbicki.crt
- Certyfikat urzędu certyfikacji ca.crt

III.3. Konfiguracja serwera OpenVPN



Obraz 35. Skopiowanie i rozpakowanie przykładowego pliku konfiguracyjnego serwera OpenVPN.

Wykonano polecenie sudo cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz /etc/openvpn/server/serverrozbicki.conf.gz, które skopiowało przykładowy plik konfiguracyjny serwera OpenVPN do katalogu serwera. Następnie plik został rozpakowany za pomocą sudo gzip -d /etc/openvpn/server/serverrozbicki.conf.gz, przygotowując go do edycji i dostosowania do konkretnej konfiguracji serwera VPN.



Obraz 36. Generowanie i dystrybucja współdzielonego klucza TLS-crypt.

Za pomocą polecenia openvpn --genkey --secret ta.key wygenerowano współdzielony klucz TLS-crypt (ta.key), który zabezpiecza proces uzgadniania połączenia, chroniąc przed atakami typu DoS. Klucz ten został skopiowany do katalogów:

- /etc/openvpn/server – dla serwera OpenVPN
- /etc/openvpn/client/keys – dla klienta

Następnie za pomocą polecenia chown rozbicki.rozbicki /etc/openvpn/client/keys/* zmieniono właściciela plików w katalogu klienta, co zapewnia odpowiedni dostęp do kluczy prywatnych i certyfikatów podczas konfiguracji i użytkowania.

Obraz 37. Edycja pliku konfiguracyjnego serwera OpenVPN.

Za pomocą polecenia nano serverrozbicki.conf otwarto do edycji plik konfiguracyjny serwera OpenVPN znajdujący się w katalogu /etc/openvpn/server. W pliku tym należy dostosować kluczowe ustawienia, takie jak:

- Lokalizacja kluczy i certyfikatów (ca.crt, server.crt, server.key, ta.key)
- Port nasłuchiwania serwera i protokół (domyślnie: UDP)
- Sieć wewnętrzna VPN i zakres adresów IP dla klientów
- Zasady routingu i przekazywania ruchu

```
  GNU nano 4.8                                serverrozbicki.conf
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
tls-crypt ta.key_

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"

# For compression compatible with older clients use comp-lzo
```

```
# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-GCM_
auth SHA256
# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"
```

```
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert server.crt
key server.key  # This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh none

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet
```

```
# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
user nobody
group nogroup

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status /var/log/openvpn/openvpn-status.log
```

```
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert serverrozbicki.crt
key serverrozbicki.key

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh none
```

```
root@serverrozbicki:/etc# nano sysctl.conf
```

Obraz 38. Edycja pliku sysctl.conf w celu włączenia przekazywania ruchu.

Otworzono plik konfiguracyjny systemu sysctl.conf za pomocą polecenia nano sysctl.conf. W pliku dodajemy linijkę: net.ipv4.ip_forward=1

```
  GNU nano 4.8                                sysctl.conf                                Modified
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#

#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

##############################################################3
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell ^_ Go To Line M-E Redo
```

Pozwala to na włączenie funkcji przekazywania ruchu sieciowego (IP forwarding) między interfejsami, co jest niezbędne do działania serwera OpenVPN jako bramy umożliwiającej dostęp do sieci lokalnej.

```
root@serverrozbicki:/etc# sudo sysctl -p /etc/sysctl.conf
net.ipv4.ip_forward = 1
root@serverrozbicki:/etc# _
```

Obraz 39. Przeładowanie ustawień sysctl w celu włączenia przekazywania ruchu sieciowego.

Polecenie sudo sysctl -p /etc/sysctl.conf wczytuje nowe ustawienia systemowe z pliku konfiguracyjnego sysctl.conf. W wyniku działania komendy włączono przekazywanie ruchu IPv4 (IP forwarding), co jest kluczowe dla prawidłowego działania serwera OpenVPN. Linia net.ipv4.ip_forward = 1 potwierdza, że funkcja została aktywowana.

III.4. Konfiguracja zapory sieciowej i routingu

```
root@serverrozbicki:/etc# ip route list default
default via 192.168.0.1 dev enp0s3 proto dhcp src 192.168.0.157 metric 100
root@serverrozbicki:/etc# _
```

Obraz 40. Identyfikacja interfejsu sieci publicznej dla ruchu wychodzącego.

Polecenie ip route list default pozwala na wyświetlenie domyślnej trasy routingu, wskazując interfejs sieciowy obsługujący ruch wychodzący do sieci publicznej. W wyniku działania komendy wykryto, że interfejsem publicznym jest enp0s3, przez który ruch przechodzi przez bramę 192.168.0.1. Ten interfejs będzie używany w dalszej konfiguracji zapory sieciowej (ufw).

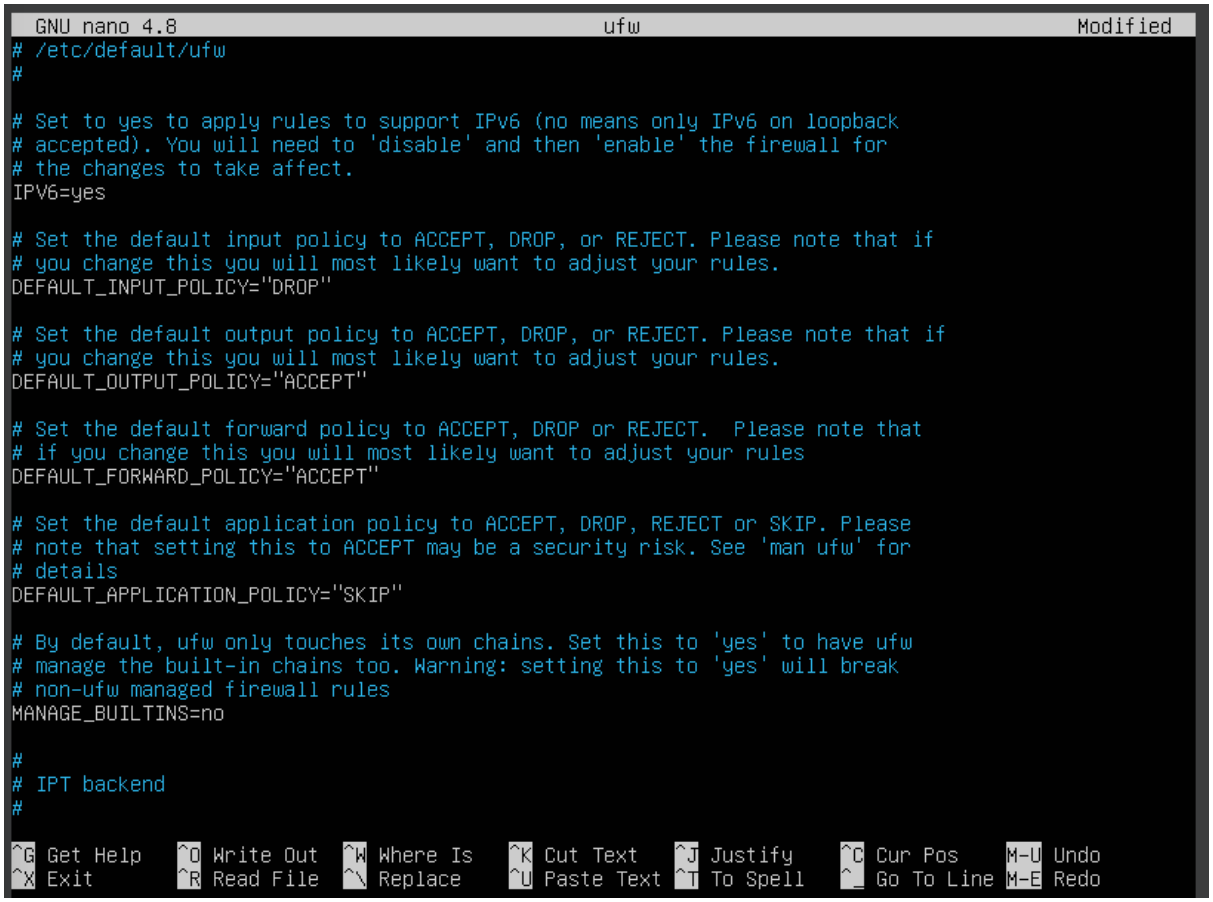
```
root@serverrozbicki:/etc/ufw# nano before.rules
```

Obraz 41. Edycja reguł zapory przed ich załadowaniem.

```
 GNU nano 4.8                          before.rules                          Modified
#
# rules.before
#
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
#   ufw-before-input
#   ufw-before-output
#   ufw-before-forward
#
# START OPENVPN RULES
# NATT table rules
*nat
:POSTROUTING ACCEPT [0:0]

#Allow traffic from OpenVPN client ens33
-A POSTROUTING -s 10.8.0.0/8 -o ens33 -j MASQUERADE
COMMIT
_
# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines


# allow all on loopback
-A ufw-before-input -i lo -j ACCEPT
-A ufw-before-output -o lo -j ACCEPT

# quickly process packets for which we already have a connection
-A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell ^_ Go To Line M-E Redo
```

Plik /etc/ufw/before.rules: Dodane zostały reguły NAT oraz przekazywanie ruchu z OpenVPN do interfejsu sieciowego. Reguły:

Dodanie reguły POSTROUTING pozwalającej na maskowanie adresów klientów VPN podczas przekazywania ruchu do sieci publicznej.



Plik /etc/default/ufw:

- Ustawiono politykę dla ruchu przychodzącego na DROP, co oznacza, że niezdefiniowany ruch będzie blokowany.
- Polityki OUTPUT i FORWARD są ustawione na ACCEPT, pozwalając na przekazywanie ruchu i dostęp do sieci.

Obraz 42. Konfiguracja portów

Port 1194/UDP: Jest on domyślnie używany przez OpenVPN, więc jego dostępność umożliwi połączenia klientów z serwerem.

Port dla OpenSSH: Zapewniono dostępność dla połączeń SSH w celu zdalnej administracji.

```
root@serverrozbicki:/etc/default# ufw disable
Firewall stopped and disabled on system startup
root@serverrozbicki:/etc/default# ufw enable
Firewall is active and enabled on system startup
root@serverrozbicki:/etc/default# _
```

Obraz 43. Reset zapory sieciowej dla zastosowania reguł

```
root@serverrozbicki:/etc/default# ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
1994/udp                   ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
1994/udp (v6)              ALLOW       Anywhere (v6)

root@serverrozbicki:/etc/default# _
```

Obraz 44. Status zapory UFW z otwartymi portami OpenSSH i OpenVPN.

Widok aktywnej zapory sieciowej (UFW) na serwerze, z otwartymi portami 22 (OpenSSH) i 1194/UDP (OpenVPN), umożliwiającymi zdalny dostęp i działanie VPN.

```
root@serverrozbicki:/etc/default# systemctl -f enable openvpn-server@serverrozbicki.service
Created symlink /etc/systemd/system/multi-user.target.wants/openvpn-server@serverrozbicki.service →
/lib/systemd/system/openvpn-server@.service.
root@serverrozbicki:/etc/default# _
```

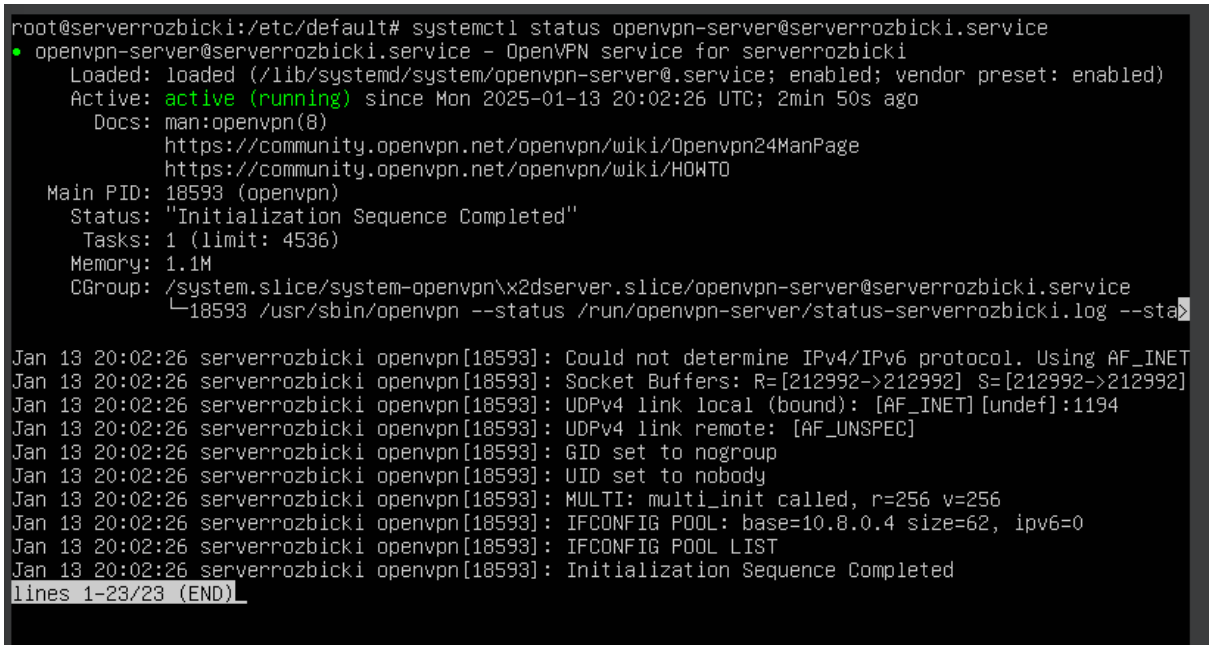
Obraz 45. Start serwera OpenVPN

Polecenie systemctl -f enable openvpn-server@serverrozbicki.service tworzy symlink i konfiguruje automatyczne uruchamianie serwera OpenVPN (instancji nazwanej) podczas startu systemu, umożliwiając stabilną obsługę połączeń VPN.

```
root@serverrozbicki:/etc/default# systemctl start openvpn-server@serverrozbicki.service
root@serverrozbicki:/etc/default#
```

Obraz 46. Uruchamianie serwera OpenVPN

Polecenie systemctl start openvpn-server@servernazwisko.service inicjuje działanie instancji serwera OpenVPN, co pozwala na natychmiastowe rozpoczęcie obsługi połączeń VPN przez skonfigurowaną usługę.



Obraz 47. Sprawdzanie statusu serwera OpenVPN

Polecenie systemctl status openvpn-server@serverrozbicki.service wyświetla szczegóły działania usługi OpenVPN, takie jak jej bieżący status, czas działania oraz ewentualne komunikaty diagnostyczne, co pozwala zweryfikować, czy serwer VPN działa poprawnie. W tym przypadku usługa jest aktywna i działa (status: **active (running)**).

III.5. Konfiguracja klienta OpenVPN

Obraz 48. Tworzenie katalogu dla plików konfiguracyjnych klienta OpenVPN

W ramach konfiguracji klienta OpenVPN, w katalogu /etc/openvpn/client utworzono podkatalog files przy użyciu polecenia mkdir files. Katalog ten będzie przechowywać niezbędne pliki konfiguracyjne i certyfikaty potrzebne do ustanowienia połączenia VPN.

Obraz 49. Kopiowanie domyślnego pliku konfiguracyjnego klienta OpenVPN

Skopiowano domyślny plik konfiguracyjny klienta OpenVPN z lokalizacji /usr/share/doc/openvpn/examples/sample-config-files/client.conf do katalogu /etc/openvpn/client pod nową nazwą baserozbicki.conf. Plik ten posłuży jako podstawa do konfiguracji klienta VPN.

```
root@serverrozbicki:/etc/openvpn/client# nano baserozbicki.conf
```

Obraz 50. Edycja pliku baserozbicki.conf

Edycja pliku konfiguracyjnym klienta OpenVPN:

```
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote your-server-ip 1194
```

```
# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
proto udp
```

```
# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
;ca ca.crt
;cert client.crt
;key client.key
```

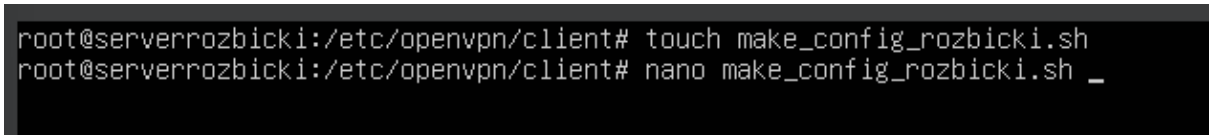
```
# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-GCM
auth SHA256

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
#comp-lzo

# Set log file verbosity.
verb 3

# Silence repeating messages
;mute 20
key-direction 1
```

W edytowanym pliku konfiguracyjnym klienta OpenVPN basenazwisko.conf wprowadzono następujące kluczowe ustawienia:

1. **Adres zdalny serwera i port** (remote your-server-ip 1194):

    - Ustalono IP lub nazwę hosta serwera VPN, na który klient ma się łączyć. Port ustawiono na domyślny 1194 dla protokołu UDP.
2. **Protokół komunikacji** (proto udp):
    - Określono użycie protokołu UDP, który jest szybszy i bardziej efektywny niż TCP w połączeniach VPN.
3. **Ścieżki do certyfikatów i klucza prywatnego**:
    - W tej sekcji pliku (po odkomentowaniu) klient użyje klucza prywatnego i certyfikatów do uwierzytelniania.
4. **Kryptografia** (cipher AES-256-GCM oraz auth SHA256):
    - Zastosowano szyfrowanie AES-256-GCM, zapewniające wysoki poziom bezpieczeństwa.
    - Weryfikacja integralności danych odbywa się za pomocą algorytmu SHA256.
5. **Kierunek klucza** (key-direction 1):
    - Konfiguracja odpowiada kierunkowi klucza TLS, wymaganym do poprawnego działania funkcji tls-auth lub tls-crypt.
6. **Dziennik i logi** (verb 3, mute 20):
    - Ustawienia poziomu szczegółowości logów (verb 3) oraz wyciszenie powtarzających się wiadomości logów (mute 20).

   Podsumowując, konfiguracja ta umożliwia bezpieczne połączenie klienta z serwerem OpenVPN, uwierzytelnienie za pomocą klucza i certyfikatów oraz zapewnia zaawansowaną ochronę danych dzięki protokołowi UDP i szyfrowaniu AES-256.



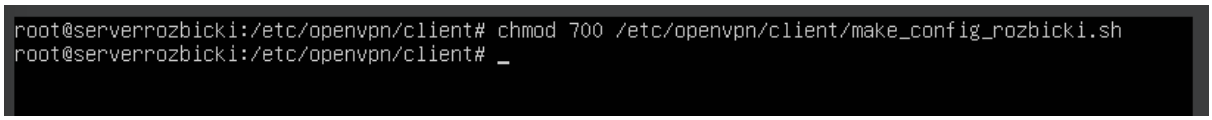
Obraz 51. Utworzenie i edycja pliku konfiguracyjnego skryptu klienta OpenVPN

Opis: Skrypt automatycznie generuje plik konfiguracyjny .ovpn klienta OpenVPN, łącząc bazowy plik konfiguracyjny (baserozbicki.conf) z odpowiednimi certyfikatami, kluczami i ustawieniami TLS.

Obraz 52. Struktura skryptu make_config_rozbicki.sh

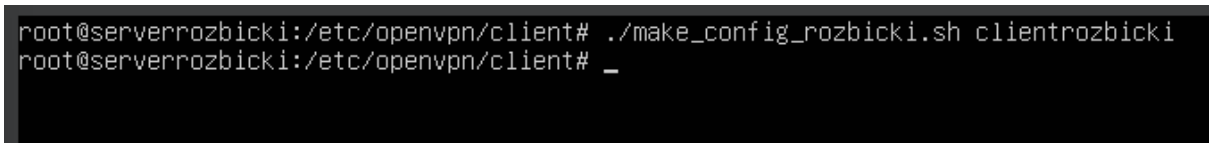
Główne elementy:

- KEY_DIR: Katalog przechowujący klucze i certyfikaty.
- OUTPUT_DIR: Katalog, w którym wygenerowane zostaną pliki .ovpn.
- BASE_CONFIG: Ścieżka do bazowej konfiguracji klienta.
- cat: Komenda służąca do odczytu i łączenia zawartości plików konfiguracyjnych i certyfikatów w jeden plik wyjściowy .ovpn.



Obraz 53. Nadanie odpowiednich praw dostępu do skryptu make_config_rozbicki.sh

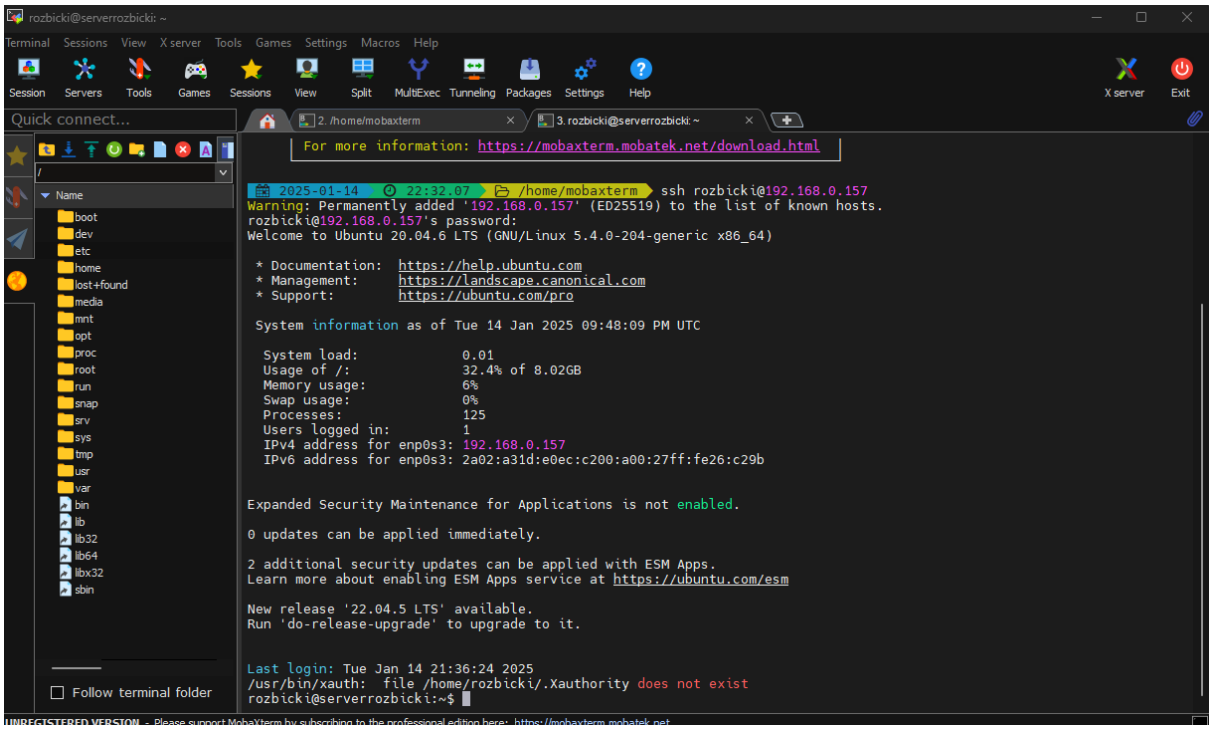
To ustawienie gwarantuje, że tylko właściciel (administrator) może modyfikować lub uruchamiać ten plik, co zwiększa bezpieczeństwo wrażliwych operacji na kluczach i konfiguracjach klienta OpenVPN.



Obraz 54. Uruchomienie skryptu make_config_rozbicki.sh

To polecenie powoduje wygenerowanie pliku konfiguracyjnego .ovpn dla klienta o nazwie clientrozbicki w katalogu docelowym (np. /etc/openvpn/client/files). Skrypt ten łączy bazowy plik konfiguracyjny z odpowiednimi certyfikatami i kluczami, tworząc kompletną konfigurację VPN gotową do użycia przez klienta.

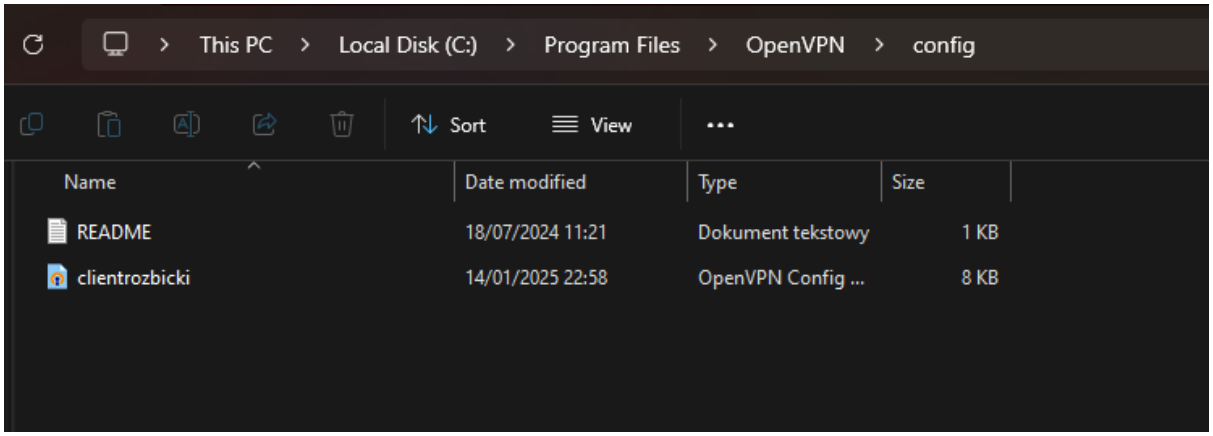
III.6. Kopiowanie plików konfiguracyjnych i zabezpieczenia



Obraz 55. Pobranie pliku konfiguracyjnego za pomocą programu MobaXterm

Otworzono MobaXterm i nawiązano połączenie SSH z serwerem 192.168.0.157. Połączenie zostało ustanowione, co umożliwia bezpośrednią interakcję z serwerem. Otworzono zakładkę SFTP/Explorer po lewej stronie interfejsu MobaXterm. Przejrzano katalogi serwera, przechodząc do lokalizacji, w której zapisano wygenerowany plik konfiguracyjny .ovpn (np. /etc/openvpn/client/files).

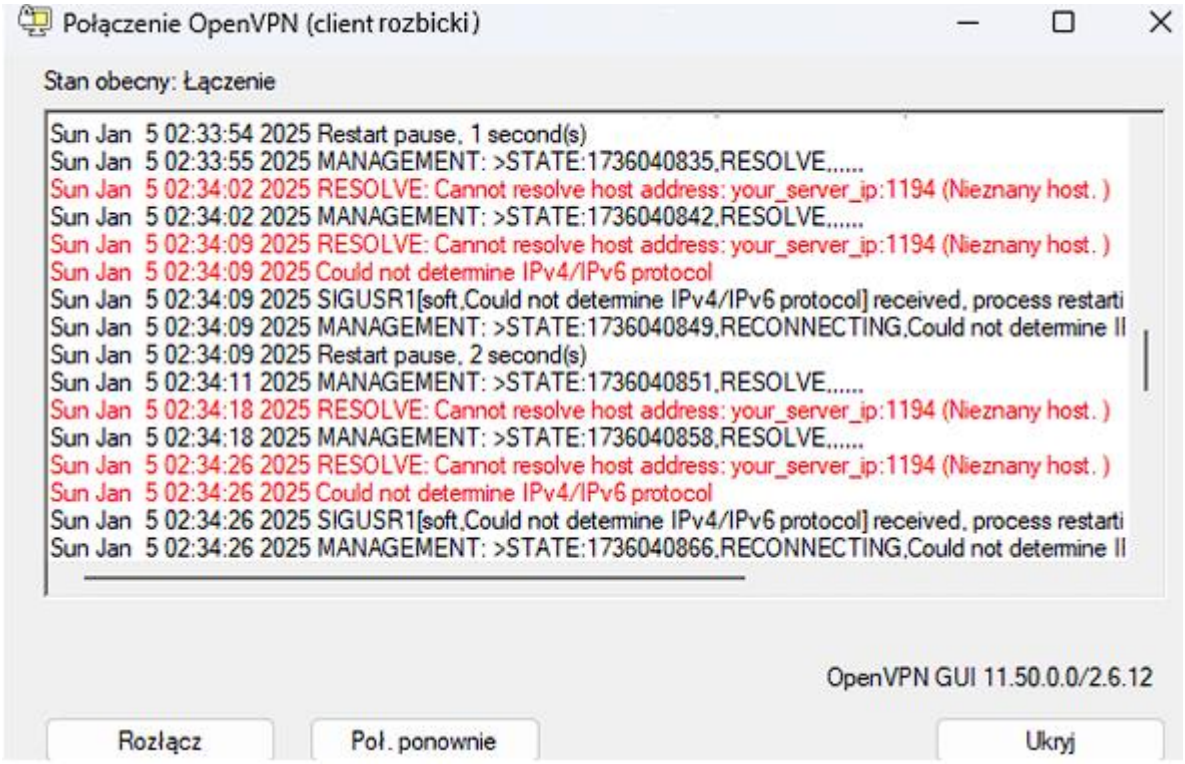
Skopiowano plik .ovpn na lokalną maszynę Windows, przeciągając go z eksploratora serwera lub używając opcji Download dostępnej w menu kontekstowym. Plik jest teraz dostępny na komputerze lokalnym do zaimportowania w kliencie OpenVPN.



Obraz 56. Umieszczenie pliku konfiguracyjnego do folderu OpenVPN

kopiowano plik **clientrozbicki.ovpn** do folderu konfiguracyjnego OpenVPN na Windowsie: **C:\Program Files\OpenVPN\config**. Plik jest teraz gotowy do użycia w kliencie OpenVPN.

Następnym krokiem jest uruchomienie OpenVPN GUI, wybranie odpowiedniego profilu, a następnie nawiązanie połączenia z serwerem VPN.



Komunikat błędu wskazuje na problem z rozpoznaniem adresu serwera VPN, co jest spowodowane błędnym lub nieprawidłowym wpisem w sekcji remote pliku clientrozbicki.ovpn.

Po zmianie remote 192.168.0.157 1194 na obrazie widzimy okno połączenia OpenVPN z komunikatem Stan obecny: Połączony, co oznacza, że tunel VPN został pomyślnie zestawiony. Ważne wpisy w logach:

- VERIFY OK dla certyfikatów: CN=caserozbicki oraz CN=serverrozbicki – weryfikacja certyfikatów przebiegła pomyślnie.
- TLSv1.3, cipher TLS_AES_256_GCM_SHA384 – Połączenie korzysta z szyfrowania AES-256-GCM w ramach protokołu TLS 1.3, zapewniając wysoki poziom bezpieczeństwa.
- Przyznany IP: 10.8.0.6 – Klientowi przydzielono adres IP z puli sieci VPN.
- Bajtów pobranych/wysłanych – Wskazuje na aktywną wymianę danych.

Podsumowanie: VPN działa poprawnie, a wszystkie kluczowe elementy (szyfrowanie, certyfikaty i przydział IP) zostały skonfigurowane bez błędów.

Rozdział IV. Podsumowanie

Cel projektu:

Celem tego projektu było skonfigurowanie w pełni działającego środowiska sieciowego opartego na technologii OpenVPN, zapewniającego bezpieczny zdalny dostęp do sieci poprzez szyfrowane połączenie VPN. W projekcie przeprowadzono szczegółowe etapy instalacji, konfiguracji i weryfikacji poprawności działania zarówno po stronie serwera, jak i klienta, co pozwoliło na stworzenie stabilnego i bezpiecznego połączenia.

I. Przygotowanie środowiska

Pierwszym krokiem było zainstalowanie serwera OpenVPN na maszynie z systemem Ubuntu. W tym celu zainstalowano wymagane pakiety, takie jak **openvpn** i **easy-rsa**, które posłużyły do generowania materiałów kryptograficznych niezbędnych do utworzenia certyfikatów bezpieczeństwa.

II. Utworzenie CA i materiałów kryptograficznych

Jednym z kluczowych etapów projektu było utworzenie urzędu certyfikacji (CA – Certificate Authority) oraz wygenerowanie materiałów kryptograficznych, które zapewniają integralność i poufność przesyłanych danych:

- **Utworzenie CA**: Wygenerowano główny certyfikat CA przy użyciu narzędzia Easy-RSA.
- **Certyfikat serwera**: Wygenerowano żądanie certyfikatu serwera, które zostało podpisane przez CA, aby zagwarantować zaufanie i autentyczność.

- **Certyfikat klienta**: Analogicznie wygenerowano i podpisano certyfikat klienta, co umożliwia jego autoryzację podczas nawiązywania połączenia.
- **Klucz TLS-crypt**: Wygenerowano dodatkowy klucz TLS-crypt, który zabezpiecza proces negocjacji połączenia VPN przed atakami typu DoS i MitM (Man-in-the-Middle).

III. Konfiguracja serwera OpenVPN

Serwer OpenVPN został skonfigurowany w oparciu o dostarczony szablon konfiguracyjny, który odpowiednio dostosowano do potrzeb projektu. Najważniejsze ustawienia obejmowały:

- **Port i protokół**: OpenVPN został skonfigurowany na porcie UDP (domyślnie 1194).
- **Certyfikaty**: Wskazano ścieżki do wygenerowanych certyfikatów i kluczy.
- **Ustawienia sieciowe**: Zdefiniowano zakres adresów IP przydzielanych klientom VPN oraz włączono przekierowanie ruchu (IP forwarding).
- **Ochrona TLS**: Włączono dodatkową warstwę bezpieczeństwa przy użyciu klucza TLS-crypt.

IV. Konfiguracja zapory sieciowej i routingu

W celu zapewnienia bezpiecznego dostępu do serwera OpenVPN i prawidłowego routingu ruchu sieciowego skonfigurowano zaporę sieciową UFW (Uncomplicated Firewall):

1. **Otwarcie odpowiednich portów**: Umożliwiono ruch na porcie 1194/UDP oraz dostęp SSH do serwera.
2. **Przekierowanie ruchu**: W pliku /etc/ufw/before.rules skonfigurowano reguły NAT, aby umożliwić przekazywanie ruchu z klientów VPN do zewnętrznej sieci.
3. **Włączenie i aktywacja UFW**: Zapora została włączona i ustawiona na automatyczne uruchamianie wraz ze startem systemu.

V. Konfiguracja klienta OpenVPN

Kolejnym etapem była konfiguracja klienta VPN. W tym celu:

1. **Skopiowano przykładowy plik konfiguracyjny klienta** do odpowiedniego katalogu na serwerze.
2. **Dostosowano konfigurację**: Wskazano adres serwera, protokół, certyfikaty i klucze oraz zabezpieczenia TLS.

3. **Stworzono skrypt automatyzujący generację plików klienckich**: Skrypt w prosty sposób generował pliki konfiguracyjne **.ovpn** dla każdego klienta, łącząc wszystkie niezbędne klucze i certyfikaty w jednym pliku.
4. **Przeniesiono plik konfiguracyjny klienta na maszynę z systemem Windows** w celu jego uruchomienia.

VI. Weryfikacja poprawności konfiguracji

Weryfikacja poprawności została przeprowadzona poprzez:

- **Uruchomienie usługi OpenVPN na serwerze** i sprawdzenie jej statusu.
- **Skopiowanie pliku konfiguracyjnego klienta na maszynę kliencką**.
- **Uruchomienie połączenia VPN na kliencie** i monitorowanie logów w poszukiwaniu ewentualnych błędów.
- **Potwierdzenie poprawnego zestawienia połączenia** – adres IP 10.8.0.0/24 przydzielony klientowi potwierdził, że tunel VPN działa poprawnie.

Wnioski

Projekt zakończył się sukcesem, a jego rezultat to w pełni funkcjonalne środowisko OpenVPN zapewniające bezpieczny zdalny dostęp do zasobów sieciowych. Dzięki zastosowaniu certyfikatów i kluczy szyfrujących oraz dodatkowych zabezpieczeń TLS-crypt, połączenie jest dobrze zabezpieczone przed nieautoryzowanym dostępem i potencjalnymi atakami sieciowymi.

Podczas realizacji projektu można było zaobserwować znaczenie dokładnej konfiguracji sieciowej i zapory UFW, ponieważ nawet drobne błędy mogły prowadzić do problemów z połączeniem. Ostateczna weryfikacja potwierdziła poprawność wszystkich etapów konfiguracji.