Uniwersytet w Białymstoku

Instytut Informatyki

Maciej Kozłowski

Prywatne sieci wirtualne oraz tunelowanie połączeń sieciowych

Praca dyplomowa napisana
pod kierunkiem
dr. Mariusza Żynela

Białystok 2021

Składam serdeczne podziękowania
dr. Mariuszowi Żynelowi
za zrozumienie, cierpliwość i
poświęcony mi czas
podczas pisania pracy licencjackiej.

Maciej Kozłowski

Spis treści

Wstęp

Ciężko jest sobie wyobrazić codzienne funkcjonowanie bez dostępu do sieci Internet. Wszystko wokół jest ze sobą połączone, a niektórych rzeczy nie jest się w stanie robić bez stałego łącza. Sieć Internet dzielimy na sieci typu **WAN** (Wide Area Network) oraz **LAN** (Local Area Network). Publiczną siecią jest oczywiście Internet, z którego korzystamy na co dzień łącząc się z całym światem, a prywatne to sieci, w których dane są przekazywane między komputerami w konkretnej organizacji, firmie lub domu, a Router pełni w tym wypadku rolę łącznika między siecią prywatną, a publiczną.

Początkowo prywatne sieci komputerowe były wykorzystywane w firmach, dzięki czemu mogły bardzo szybko i bezpiecznie przesyłać potrzebne informacje między pracownikami jak również urządzenia w tej sieci miały dostęp na przykład do drukarek podłączonych do tej sieci. Gdy inne odziały firm znajdowały się stosunkowo blisko siebie, firma decydowała się budować własne łącze lub dzierżawić już istniejące. W sytuacji jednak, gdy poszczególne oddziały firmy dzielił duży dystans rozwiązanie te okazało się zbyt kosztowne i zaczęto wykorzystywać publiczne łącze do tworzenia w nich prywatnego połączenia za pomocą takich technik jak połączenia tunelowe SSH, czy VPN.

Celem tej pracy jest przedstawienie zasady działania połączeń SSH oraz VPN, oraz potencjalnych problemów występujących podczas ich realizacji. W pierwszym rozdziale skupiliśmy się na opisie działania VPN, jego historii i różnicach występujących między różnymi protokołami, zagłębiając się bardziej w sam OpenVPN. Drugi rozdział odpowiada za wytłumaczenie w jaki sposób zadbać o bezpieczeństwo podczas połączenia VPN. Trzeci natomiast skupił się stricte na działaniu SSH. W tym rozdziale zostało przedstawione i wytłumaczone w jaki sposób zaimplementować działanie port forwardingu. Ostatnim czwartym rozdziałem jest konfiguracja VPN. Przedstawiliśmy tam adresację adresów, przedstawienie różnicy między bridgingiem, a routingiem, koncentrując się na końcu na przedstawieniu w jaki sposób można skonfigurować połączenia połączenia typu point-to-point, a także klient-serwer.

Rozdział 1

Sieci VPN

1.1 Idea i zastosowanie

VPN (*Virtual Private Network*) został stworzony aby zniwelować problem dużych odległości między sieciami prywatnymi i ułatwia poruszanie się po sieci publicznej tworząc w niej prywatne połączanie. Bezpieczeństwo w VPN jest możliwe dzięki kilku narzędziom takimi jak firewall, tunelowanie pakietów, szyfrowanie i uwierzytelnianie danych. Dzięki temu nasze dane są szyfrowane i niewidoczne dla innych użytkowników, którzy nie są połączeni to tej prywatnej sieci VPN.

1.2 Zasada działania

VPN działa na zasadzie utworzenia tymczasowej sieci prywatnej z wykorzystaniem sieci publicznej, co oznacza stworzenie wirtualnego (jak sama nazwa mówi) połączenia.

1.2.1 Tunelowanie

Tunelowanie polega na utworzeniu "tunelu" w WAN, który ma za zadanie przekazywać komunikację między użytkownikami tunelu. Tworząc takie połączenie, informacje w nim przechodzące są niedostępne dla osób spoza tunelu. W zwykłym tunelowaniu jest możliwość przesyłania tylko jednego protokołu, a VPN umożliwia przesyłanie wielu protokołów na raz.

1.3 Rodzaje VPN

1.3.1 PPTP

Protokół **PPTP** (ang. Point-to-Point Tunneling Protocol) zaczął swoją historię w 1999 roku. Został stworzony przy współpracy kilku firm między in-

nymi Microsoft i 3Com. Dzięki szybkości jaką oferował PPTP był stosowany w większości VPN, lecz przez zbyt słabą formę zabezpieczeń przestało się z niego korzystać. Słabym punktem tego protokołu było uwierzytelnianie, które bazowało miedzy innymi na protokole MS-CHAP, który został wielokrotnie złamany i nie stanowił już żadnego problemu w odszyfrowaniu "zabezpieczonych" danych.

1.3.2 IPsec

IPsec (ang. Internet Protocol Security) działa na dwóch wersjach **IP** (ang. Internet Protocol), IPv4 oraz IPv6. IPsec jest w stanie chronić każdy protokół, który obsługuje IP, między innymi TCP, UDP, a także ICMP. Zabezpieczenia za jakie IPsec jest odpowiedzialny to szyfrowanie i uwierzytelnianie (por. [3]).

1.3.3 L2TP

L2TP (ang. Layer Two Tunneling Protocol) jest protokołem, który powstał z połączenia zalet dwóch innych protokołów, wcześniej wspomnianego PPTP i **L2F** (ang. Layer 2 Forwarding). Mowa o szybkości działania PPTP oraz szyfrowaniu IPsec, który był wspierany przez L2F.

1.3.4 GRE

GRE (ang. Generic Routing Encapusulation) został stworzony przez CISCO Systems w 1994 roku. Jego działanie polega na połączeniu pakietów wewnętrznych z zewnętrznym IP, co umożliwia komunikację z siecią Internet. Pakiety przesyłane przez ten protokół, są pakietami wewnętrznymi i są niewidoczne, aż do momentu osiągnięcia swojego celu. Gdy ten cel zostanie osiągnięty dane, które były publiczne zostaną usunięte, a odbiorca otrzyma interesującą go wiadomość.

1.3.5 OpenVPN

Pierwsza wersja protokołu **OpenVPN** została stworzona przez Jamesa Yonana w maju 2001 roku. Pełniła wtedy tylko kilka podstawowych funkcji, takie jak na przykład tunelowanie pakietu IP przez UDP. Teraz oprócz samego protokołu ma własne oprogramowanie, które ułatwia korzystanie z jego funkcjonalności. OpenVPN jest aktualnie najbardziej rozpoznawalnym VPN na świecie. Oferuje on zabezpieczenia point-to-point oraz site-to-site. Za uwierzytelnianie odpowiada między innymi "pre-shared key", czyli klucz który jest wysyłany przed uzyskaniem dostępu. Klucz po stronie serwera i klient musi się zgadzać. Kolejnym sposobem autoryzacji jest "Certificate-Base", polega on na stworzeniu certyfikatu na dysku klienta, dzięki któremu serwer jest w

stanie zidentyfikować połączenie. Ostatnim i najbardziej znanym jest wcześniej wspominana metoda logowania, która polega na podaniu odpowiedniego loginu i hasła (por. [4]).

Porównanie do IPSec i PPTP

OpenVPN w porównaniu do IPSec jest bardziej przyjazny w różnym środowisku. Można go bez żadnego problemu zainstalować na każdym systemie operacyjnym, a IPSec za każdym razem wymaga nowej implementacji. PPTP natomiast jest narzędziem, które jest już zainstalowane w systemie Windows więc nie wymaga zaangażowania przy jego instalowaniu. Największymi zaletami OpenVPN jest to, że w domyśle wspiera wszystkie potrzebne mechanizmy, takie jak wsparcie dynamicznego adresu IP, współgra z firewall, operacjami NAT, jak również obsługuje domyślnie wiele protokołów.

Rozdział 2

Bezpieczeństwo w VPN

2.1 Uwierzytelnianie

Uwierzytelnianie jest procesem który polega na stwierdzeniu, że dane połączenie jest dozwolone. Jest wiele sposób na autoryzacje połączenia, jednak najbardziej popularnym jest wykorzystanie logowania za pomocą danych, które zna tylko dana osoba. Na przykład dane do logowania. Jeżeli określony użytkownik poda odpowiedni login i hasło to dostęp to poszczególnych danych będzie mu udostępniony. W dzisiejszych czasach popularną dodatkową opcją jest **2FA** (ang. Two-factor authentication), czyli weryfikacja dwuetapowa. Jak sama nazwa mówi, aby się zalogować musimy podać dane do logowania i dodatkowy kod, który jest generowany w sposób losowy. Najczęstszym sposobem na wysyłanie takiego kodu jest przesłanie go na email (por.[2]).

2.2 Firewall

Ściana przeciw ogniowa (ang. ***Firewall***) nazywany również zaporą sieciową pełni dwie podstawowe funkcje. Pierwsza funkcja polega na kontrolowaniu połączeń przychodzących jak i wychodzących, a druga na monitorowaniu tych połączeń. Firewall jak sama nazwa mówi jest swojego rodzaju ścianą, która dzieli sieć wewnętrzną (LAN) z Internetem i zapobiega tworzeniu niepożądanych połączeń (por.[1]).

2.2.1 Application layer firewall

Tłumacząc Application layer firewall można już domyślić się jakie zadanie to oprogramowanie pełni. To nic innego jak warstwa (ang. *leyer*) lub sekcja w firewallu, która odpowiada za przepływ pakietów wysyłanych od samej aplikacji jak i pakietów przychodzących do niej. Jej najważniejszym zadaniem jest kontrolowanie przetwarzanych informacji, tak aby nie przepuścić niepożądanych treści, takich jak złośliwe oprogramowanie, czy szkodliwe strony internetowe.

2.2.2 Network layer firewall

Jak nazwa wskazuje Network Layer Firewall działa w warstwie sieci modelu OSI. Może kontrolować przepływ pakietów na podstawie adresów IP (warstwa 2 internetowa TCP/IP) i portów (warstwa 3 transportowa TCP/IP)

- **Stateful firewall** gromadzi informacje o aktywnych sesjach oraz wykorzystuje je do sprawdzenia stanu pakietu, dzięki czemu znacznie przyśpiesza to proces ponownej weryfikacji. Zajmuje się śledzeniem stanu połączeń, sprawdza on czy strumienie TCP, adresy IP oraz porty TCP i UDP pastują do siebie, to znaczy czy informację przychodzące nie różnią się od informacji wychodzących.
- **Stateless** sprawdza każdy pakiet indywidualnie. Domyślnie sprawdzany jest tylko nagłówek pakietu co daje lepszą wydajność. Dzięki czemu lepiej sobie radzi przy dużym obciążeniu łącza. Wadą tego zastosowania jest to, że po jakimś upływie czasu lub po zakończeniu połączenia dane nie zostają zapisywane przez co zostają utracone i przy ponownym połączeniu wszystko odbywa się od początku (por.[6]).

2.3 Szyfrowanie

Szyfrowanie może być rozumiane jako metoda zmieniająca treść danych, która jest rozumiana tylko nadawcy i osobie lub osobom, które są jego odbiorcami. Jest to możliwe dzięki kluczowi szyfrującemu, który posiadają tylko osoby mające dostęp do tego połączenia.

Rozdział 3

SSH i tunele

SSH (*Secure Shell Protocol*) jest bardzo popularną metodą komunikacyjna używana w sieciach TCP/IP. Jego założeniem jest wykorzystanie połączeń terminalowych do komunikacji. Aby zapewnić bezpieczeństwo, SSH wykorzystuje narzędzia szyfrujące, które modyfikują informacje wychodzące z komputera, dzięki czemu gdy dotrą do odpowiedniego komputera zostają automatycznie rozszyfrowywane przez SSH, a urządzenia poza tym połączeniem dostają zakodowaną wiadomość. Dodatkowo SSH jest wyposażony w mechanizm przekazywania portów. Daje on między innymi możliwość omijania zapory ogniowej, jak i również połączenia się z siecią prywatna z komputera domowego.

3.1 Szyfrowane połączenie terminalowe

Połączenie, które odbywa się między klientem, a serwerem zabezpieczane jest za pomocą jednego z kilku algorytmów **RSA**, **DSA**, lub **ECDSA**. Jeden z nich postaramy się szczegółowo opisać.

- **RSA** (*Rivest-Shamir-Adleman*)

  Algorytm RSA stworzony w 1977 roku, a swoją nazwę otrzymał po jego założycielach Ron Rivest, Adi Shamir i Leanard Adleman. Jego działanie polega na utworzeniu dwóch powiązanych ze sobą kluczy, jeden jest publiczny, a drugi prywatny. Ten pierwszy nie jest ukryty i każdy ma możliwość za pomocą tego klucza zaszyfrować swoje dane, lecz aby je odszyfrować potrzebny jest klucz prywatny. To samo tyczy się klucza prywatnego, osoba posiadająca klucz prywatny jest w stanie wykorzystać go do zaszyfrowania swoich danych, ale tylko osoby posiadające pasujący klucz publiczny są wstanie je odszyfrować. Algorytm ten składa się z dwóch następujących kroków:

  - **Generowanie klucza**
    Generowanie kluczy polega na utworzeniu klucza publicznego i klucza prywatnego.

  - **Szyfracja i deszyfracja klucza**

    Aby zaszyfrować i odszyfrować klucze należy dokonać określone działania matematyczne, dzięki którym nasza wiadomość będzie możliwa do odczytania tylko dla osób posiadającym pasujący klucz.

- **DSA** (*Digital Signature Algorithm*)

  Algorytm ten zaczął swoją historię w 1991 roku dzięki Narodowemu Instytutu Standaryzacji i Technologii w USA na potrzeby **DSS** (*Digital Signature Standard*). Jego działanie opiera się na czterech operacjach:

  - **Generowanie klucza**

    Polega na utworzeniu jak w większości algorytmów klucza publicznego i prywatnego.

  - **Dystrybucja klucza**

    Osoba podpisująca wysyła klucz publiczny do odbiorcy, a klucz prywatny zachowuje dla siebie.

  - **Podpisywanie**

    Tworzenie podpisu działa na zasadzie obliczeń określonych działań matematycznych, powiązanych z resztą z dzielenia.

  - **Weryfikacja podpisu**

    Jeżeli klucze z podpisu są sobie równe, to oznacza, że podpis jest prawidłowy (por.[10]).

- **ECDSA** (*Elliptic Curve Digital Signature Algorithm*)

  ECDSA stosuje ideę DSA z wykorzystaniem techniki kryptografii krzywych eliptycznych. Sam pomysł z wykorzystaniem krzywych eliptycznych został wymyślony przez Neala Koblitza oraz Victora Soula Millera w 1985 roku.

  W porównaniu do jego poprzedników ECDSA wykorzystuje krótszy klucz przy tym samym poziomie zabezpieczeń, dzięki czemu jest wydajniejszy. Dla porównania rozmiar ECDSA mieści się w zakresie od 163 do 517 bitów, gdzie RSA wymaga od 1024 do 15360 bitów.

  Liczba bazowa do stworzenia klucza musi być losową liczbą pierwszą. Weryfikacja prawidłowości podpisu polega na mnożeniu punktów krzywej eliptycznej przez skalar. Aby stwierdzić prawdziwość podpisu algorytm ten sprawdza czy punkt, który wybieramy leży na krzywej eliptycznej (por. [11]).

3.2 Przekazywanie portów (port forwarding)

Rysunek 3.1: Przekazywanie portów przez SSH.

Port forwarding jest metodą, która umożliwia przekazywanie niezabezpieczonego ruchu TCP w bezpiecznym tunelu SSH przez różnego rodzaju blokady, takie jak firewall, czy router. Mechanizm ten najczęściej jest używany do udostępniania różnych usług znajdujących się w innej sieci LAN, do której nie mamy dostępu. Dzieje się to poprzez zamienienie adresu i portu z jednego urządzenia z innej sieci, na adres i port docelowego urządzenia w innej sieci, na przykład z komputera domowego do innego komputera znajdującego się w firmie. Cały ruch odbywa się za pośrednictwem łącza publicznego czyli Internet. Protokoły takie jak HTTP, POP3, TELNET, SMTP, FTP i wiele innych, mogą być bezpiecznie przesłane przez zaszyfrowany tunel SSH.

Istnieją trzy rodzaje port forwardingu:

- Local,
- Remote,
- Dynamic.

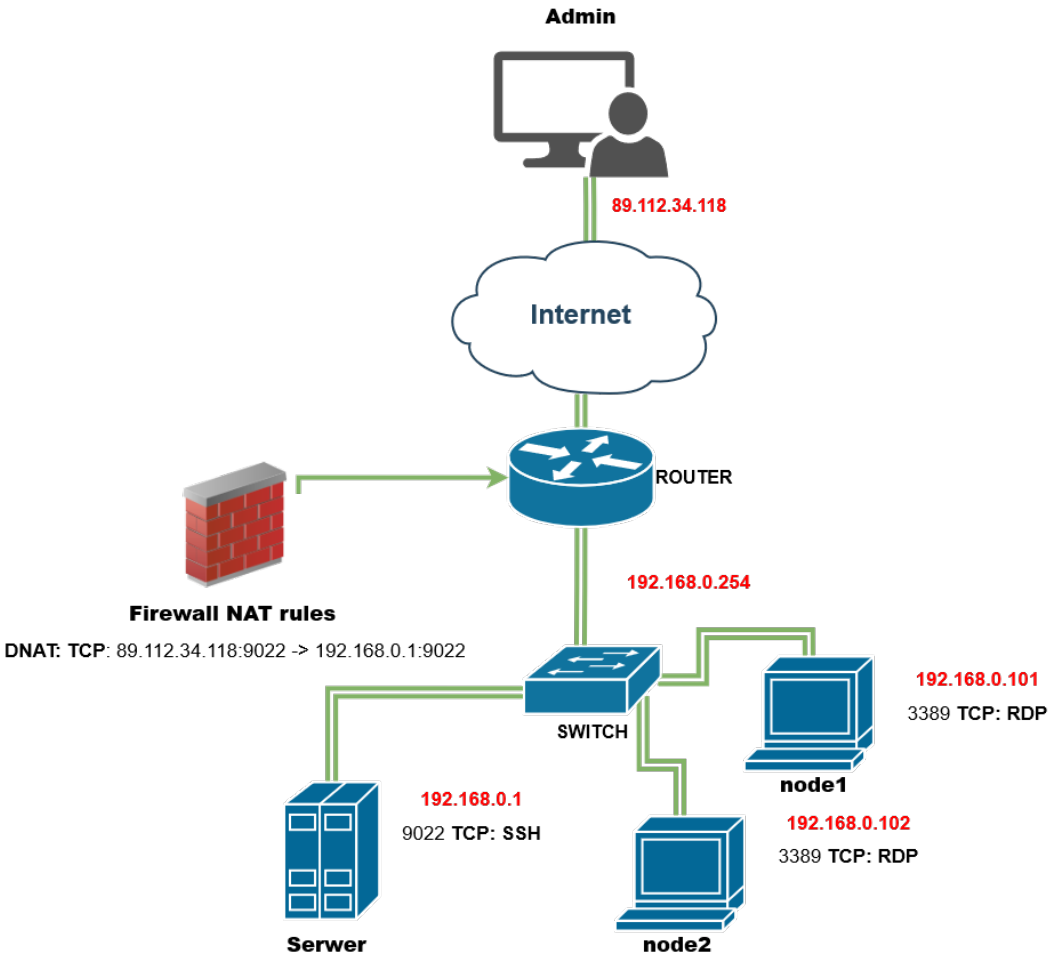
Potrzebne komendy do port forwardingu

Lista wszystkich dostępnych komend wywołuje się za pomocą `ssh -h` lub `ssh -help`.

```
C:\Users\Komeb>ssh -h
unknown option -- h
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```

Rysunek 3.2: Dostępne opcje SSH.

Nas najbardziej interesują trzy "-L" (Local), "-R" (Remote), "-D" (Dynamic).



Rysunek 3.3: Lokalne przekazywanie portów.

3.2.1 Lokalne (Local)

Jest to najczęściej wykorzystywany rodzaj port forwardingu. Załóżmy, że serwer SSH jest umieszczony w firmie. Lokalne przekierowywanie portów stosuje się do podłączenia z klienta SSH (na przykład z naszego lokalnego komputera), do serwera SSH w firmie. Dzięki temu mamy możliwość obejścia zapory sieciowej, która blokuje określone usługi.

Przykładowa konfiguracja

Jesteśmy administratorem odpowiedzialnym za sieć, komputery i oprogramowanie w pewnej organizacji. Organizacja ta posiada łącze internetowe ze stałym adresem publicznym 89.112.34.118. W sieci lokalnej tej organizacji mamy serwer z prywatnym adresem 192.168.0.1 oraz komputery biurowe `node1`,

node2 z adresami 192.168.0.101, 192.168.0.102 odpowiednio. Port TCP 9022 interfejsu w sieci publicznej na routerze został przemapowany za pomocą odpowiednich reguł DNAT na port 9022 serwera, na którym mamy uruchomiony serwer SSH na tym właśnie porcie (9022 zamiast 22 aby nieco utrudnić życie hackerom). Zakładamy, że jako administrator mamy dostęp do serwera organizacji jako sysadm poprzez SSH z ustalonych lub dowolnych adresów IP w sieci Internet. Na komputerach organizacji, czyli node1, node2 z Windows mamy uruchomione RDP i hasło administratora (rys. 3.4).

Aby na przykład skonfigurować dostęp do drukarki sieciowej na komputerze node1, najpierw tworzymy tunel ze swego komputera admin do node1 w następujący sposób:

```
ssh -p 9022 -nNTq -L 9999:192.168.0.101:3389 \
sysadm@89.112.34.118
```

Polecenie to wykonujemy na swoim komputerze admin.Teraz, na tym samym komputerze, możemy uruchomić połączenie RDP do node1:

```
rdesktop localhost:9999
```

Komunikacja na port 9999 naszego komputera przekazywana jest w tunelu SSH, za pośrednictwem serwera organizacji, na port 3389 na node1.

Jeśli często łączymy się z naszego biura, z różnych komputerów, do komputerów w obsługiwanej organizacji, to możemy utworzyć tunel SSH na stałe. Załóżmy, że w naszym biurze mamy serwer o adresie lokalnym 172.16.0.1. Na tym serwerze wykonujemy polecenie:

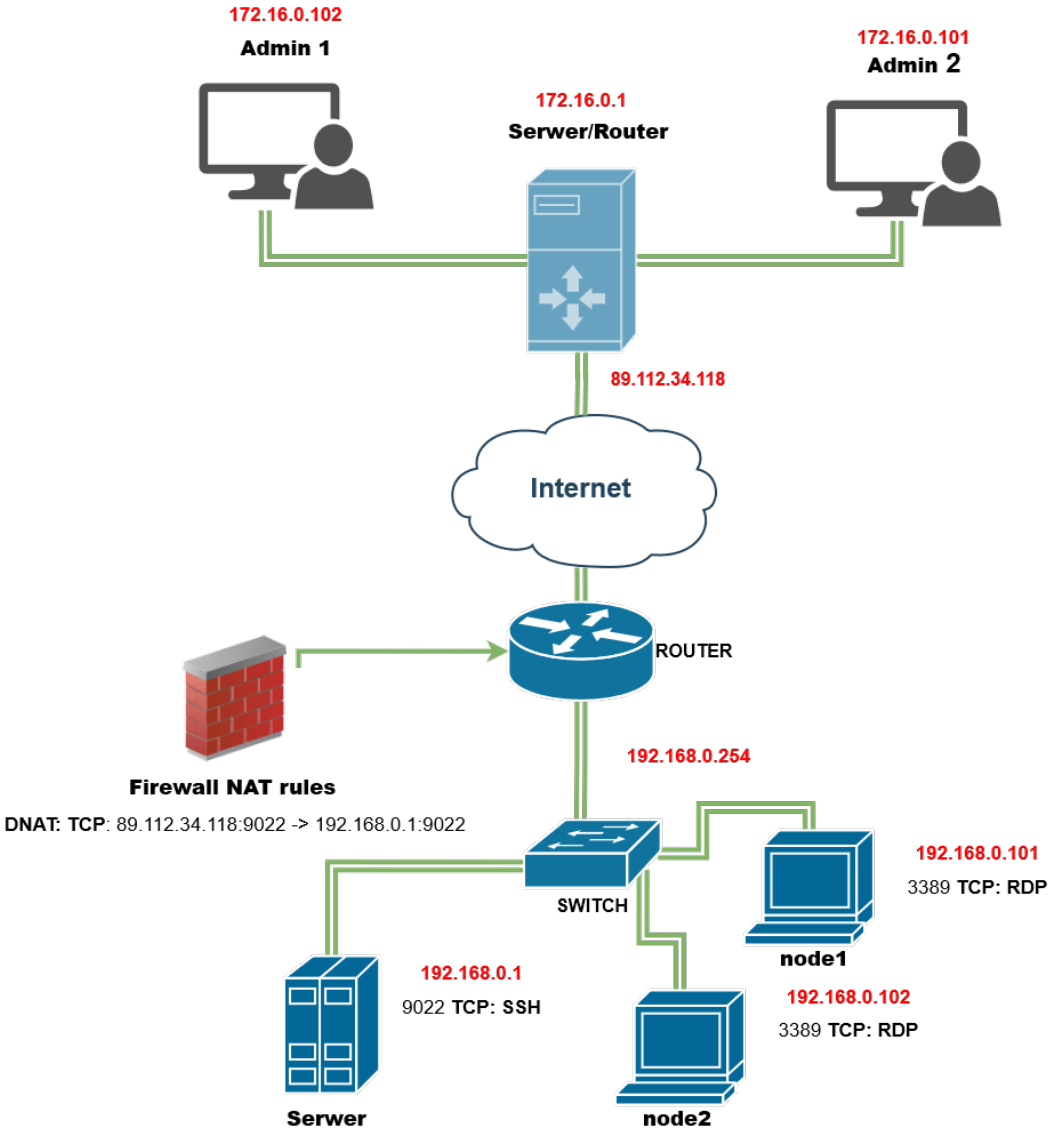
```
ssh -p 9022 -nNTq -L 172.16.0.1:9999:192.168.0.101:3389 \
sysadm@89.112.34.118
```

Tworzy ono tunel SSH, którego jednym końcem po naszej stronie jest 172.16.0.1:9999 a drugim końcem jest 192.168.0.101:3389 w organizacji. Teraz, z dowolnego komputera w naszej sieci lokalnej, admin1, admin2 możemy połączyć się przez RDP do node1:

```
rdesktop 172.16.0.1:9999
```

Dobierając odpowiednio porty możemy uzyskać połączenia nie tylko RDP, ale także inne, do serwerów HTTP, drukarek, baz danych itp. pomiędzy naszym biurem a organizacją. Jedynym wymaganiem jest posiadanie dostępu przez SSH do serwera organizacji, z włączonym przekazywaniem portów. Na routerze organizacji nie musimy mapować (korzystając z DNAT) innych portów poza jedynym portem TCP serwera SSH.

Aby takie połączenia były trwałe musimy zadbać, aby po zerwaniu zostały one wznowione na naszym serwerze. Przy większej ilości takich tuneli wygodniejsze z pewnością jest zastosowanie VPN.



Rysunek 3.4: Lokalne przekazywanie portów z zastosowaniem *172.16.0.1.*

3.2.2 Zdalne (Remote)

Remote port forwarding jest czyś odwrotnym w stosunku do Local port forwarding. Sieć lokalna naszej organizacji chroniona jest firewallem i dostęp do naszego komputera node1 jest niemożliwy z sieci Internet.

Przykład zastosowania remote port forwarding

Powiedzmy, że znajdujemy się w biurze i wybieramy się do domu skąd chcielibyśmy dokończyć naszą pracę. W tym celu, za pośrednictwem serwera SSH naszej organizacji, tworzymy tunel odwrotny, którego jeden koniec to port 3389 (usługa RDP) na node1, a drugi to port 9999 na naszym komputerze domowym. Zakładamy oczywiście, że w domu mamy serwer SSH nasłuchujący na porcie TCP 9022 i znamy swój adres publiczny, na przykład 32.135.201.172. Oczywiście port TCP 9022 może być przemapowanym portem interfejsu WAN naszego domowego routera. Nasz komputer nie musi mieć bezpośredniego połączenia z dostawcą Internetu (rys. 3.5).

Przed wyjściem z pracy, na serwerze SSH w biurze wykonujemy polecenie:

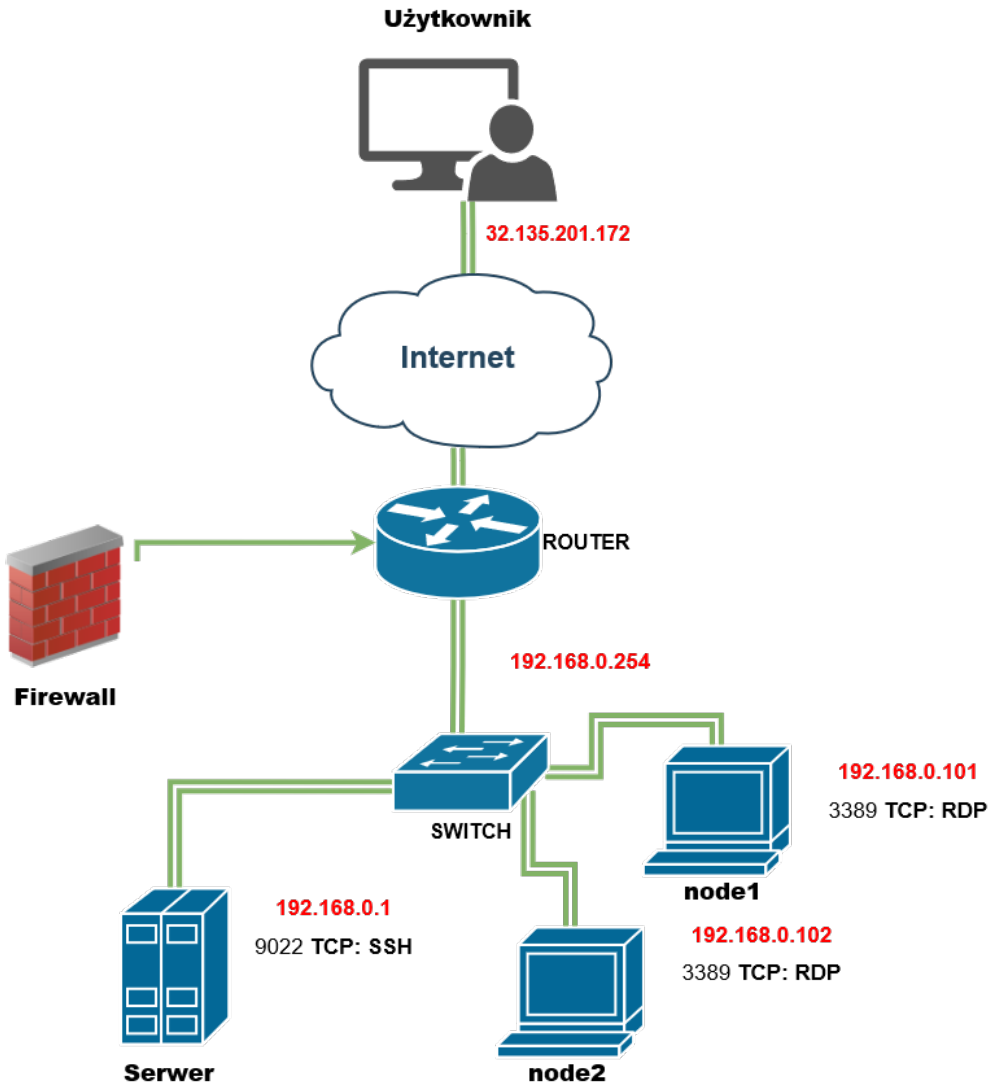
```
ssh -p 9022 -nNTq -R 9999:192.168.0.101:3389 user@32.135.201.172
```

Po powrocie do domu, łączymy się do swego komputera w pracy:

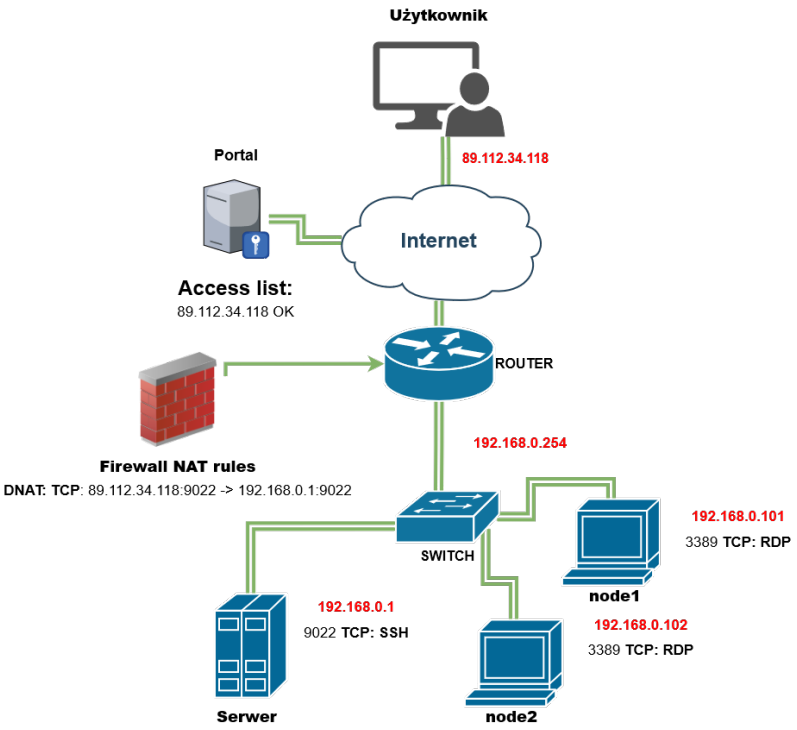
```
rdesktop localhost:9999
```

Dzięki utworzonemu tunelowi, komunikacja na port 9999 naszego domowego komputera zostanie przekazana za pośrednictwem serwera SSH w biurze do komputera node1.

Dobrze skonfigurowany firewall będzie zamykał połączenia, które przez dłuższy czas są bezczynne (ang. idle). Jeśli obawiamy się, że zanim połączymy się z domu do naszego komputera w biurze, zestawiony tunel zostanie zamknięty przez firewalla naszej organizacji, to należy na drugim jego końcu, czyli w domu, uruchomić program, który co jakiś czas będzie przesyłał porcję danych poprzez tunel. Może to być na przykład program top uruchomiony zdalnie z domu na node1 w pracy.



Rysunek 3.5: Zdalne przekazywanie portów.



Rysunek 3.6: Dynamiczne przekazywanie portów.

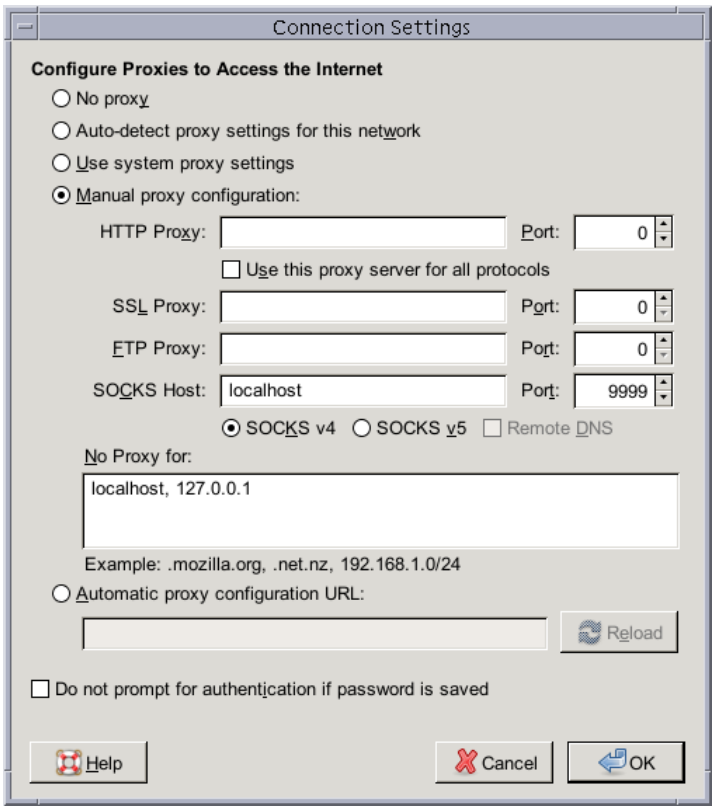
3.2.3 Dynamiczne (Dynamic)

Dynamiczne jest najrzadziej wykorzystywaną metodą. Jej działanie jest bardzo skuteczne, lecz wymaga bardzo dużo pracy aby to skonfigurować. Zamienia klienta SSH w serwer SOCKS proxy. Charakteryzuje się on tym, że aby on działał trzeba ustawić te połączenie do określonego celu, a gdy chcemy użyć go ponownie wszystko trzeba robić on nowa.

Przykład zastosowania dynamic port forwarding.

Powiedzmy, że organizacja, w której pracujemy, posiada dostęp do pewnego portalu, którym może być wirtualna biblioteka, magazyn części zamiennych itp.. Dostęp do tego portalu posiadają tylko wybrane instytucje (rys. 3.6). Wejść można do niego tylko z określonych adresów IP w sieci Internet. Jeśli jesteśmy jednym z użytkowników pracujących zdalnie w domu, co jest akurat na czasie, to nie mamy dostępu do portalu. Gdy mamy dostęp do serwera SSH jako user w naszej organizacji to problem możemy obejść w następujący sposób. Na swoim domowym komputerze tworzymy dynamiczny tunel poleceniem:

```
ssh -p 9022 -nNTqC -D 9999 user@89.112.34.118
```



Rysunek 3.7: Konfiguracja proxy w przeglądarce Firefox.

Teraz w przeglądarce, konfigurujemy proxy jak na rys. 3.7:

Od tej pory, gdy serfujemy po sieci, robimy to za pośrednictwem serwera SSH w naszej organizacji. Jeśli w przeglądarce wpiszemy `http://pokapoka.pl/`, aby sprawdzić z jakiego adresu IP się łączymy to zobaczymy 89.112.34.118, czyli publiczny adres IP biura naszej organizacji. Możemy w pełni korzystać z zasobów portalu, tak jakbyśmy byli fizycznie w tym biurze. Przy dobrych łączach szerokopasmowych, dość powszechnych obecnie, może nawet nie zauważymy spowolnienia w otwieraniu stron internetowych.

Po skończeniu pracy w przeglądarce należy wyłączyć proxy przywracając poprzednie ustawienia i zamknąć tunel przerywając proces ssh uruchomiony na naszym komputerze, na przykład CTRL+C w oknie terminala (por. [12]).

3.3 Tunelowanie SSH, a VPN

Oba te standardy są do siebie podobne, jedno i drugie tworzy bezpieczny tunel za pomocą szyfrowanego połączenia. Każdy ruch w VPN odbywa się przez jego tunel i użytkowanie jest dużo przyjemniejsze. Gdy chcemy połączyć się do firmowego VPN otrzymujemy od razu dostęp do różnych urządzeń czy pli-

ków, bez uwagi na to z jakiego miejsca jest połączenie. Podłączając się do serwera VPN, wszystko wygląda tak jakbyśmy byli fizycznie w tej firmie. SSH natomiast głównie polega na połączeniu terminalowym i mógłby działać podobnie jak VPN, lecz wymagałoby to skonfigurowania indywidualnie każdego połączenia.

Rozdział 4

Konfiguracja VPN

W ramach tej pracy wykonaliśmy kilka różnych konfiguracji połączeń VPN w oparciu o OpenVPN. Dlatego w tym rozdziale, jeśli nie jest powiedziane inaczej mamy na myśli właśnie OpenVPN, choć spora część przedstawionych tutaj informacji jest ogólna.

4.1 Planowanie adresacji IP w sieciach lokalnych

Konfiguracja VPN zwykle sprowadza się do połączenia ze sobą kilku, różnych sieci lokalnych. W sieciach lokalnych używa się prywatnych adresów IP. Organizacja IANA (ang. Internet Assigned Numbers Authority), odpowiedzialna za koordynowanie adresacji w sieci Internet, przewidziała 3 zakresy adresów prywatnych do stosowania w sieciach lokalnych LAN. Te adresy prywatne nie mogą być używane w sieci Internet. Są one odrzucane przez routery spinające rezległe sieci WAN.

Adresy prywatne			
Nazwa klasy	**Zakres IP**	**Ilość adresów**	**Maska podsieci**
A	10.0.0.0 10.255.255.255	16 777 216	10.0.0.0/8 (255.0.0.0)
B	172.16.0.0 172.31.255.255	1 048 576	172.16.0.0/12 (255.240.0.0)
C	192.168.0.0 192.168.255.255	65 536	192.168.0.0/16 (255.255.0.0)

Tabela 4.1: Klasy sieci prywatnych (por. [7]).

Adres IPv4 dzieli się na cztery oktety, które są oddzielone między sobą kropką. Każdy z nich składa się z 8 bitów, jest liczbą z przedziału od 0 do

255. Klasy sieci wykorzystywane są do podziału sieci zależnie od ilości adresów jakich potrzebujemy.

Ilość możliwych adresów w sieci określa maska podsieci (ang. subnet mask). Jest ona nierozłączną częścią definicji podsieci. Razem z adresem IP jednoznacznie wyznacza adres bazowy podsieci, adres rozgłoszeniowy i zakres dostępnych adresów hostów. Można powiedzieć, że maska wyznacza podział adresu IP na adres bazowy sieci oraz adres host w tej podsieci.

Maski podsieci i ich limity		
Maska	**Bity maski**	**Ilość adresów**
255.0.0.0	/8	16 777 216
255.128.0.0	/9	8 388 608
255.192.0.0	/10	4 194 304
255.224.0.0	/11	2 097 152
255.240.0.0	/12	1 048 576
255.248.0.0	/13	524 288
255.252.0.0	/14	262 144
255.254.0.0	/15	131 072
255.225.0.0	/16	65 536
255.225.128.0	/17	32 768
255.225.192.0	/18	16 384
255.225.224.0	/19	8 192
255.225.240.0	/20	4 096
255.225.248.0	/21	2 048
255.225.252.0	/22	1024
255.225.254.0	/23	512
255.225.255.0	/24	256
255.225.255.128	/25	128
255.225.255.192	/26	64
255.225.255.224	/27	32
255.225.255.240	/28	16
255.225.255.248	/29	8
255.225.255.252	/30	4
255.225.255.254	/31	2
255.225.255.255	/32	1

Tabela 4.2: Maski podsieci (por. [8]).

Jeżeli potrzebujemy bardzo dużej sieci, w której przydzielimy adresy wielu urządzeniom, na przykład w strefach z otwartą siecią WiFi (OpenWiFi), wykorzystamy klasę sieci A, gdyż daje ona możliwość podpięcia ponad szesnastu milionom urządzeniom jednocześnie. Pierwszy oktet w klasie A jest wykorzystywany jako adres sieci, a dzięki temu zwolnione są pozostałe trzy na adres użytkownika (ang hosta) w tej sieci. Daje nam to 8 bitów na adres sieciowy i

24 bity do wykorzystania na adresy hostów. Przy tak dużej przestrzeni adresów jaką zapewnia klasa A, należy jednak uważać na pakiety rozgłoszeniowe. Każdy z takich pakietów będzie rozesłany do 2^{24} hostów, co może powodować zatory nawet przy zastosowaniu wydajnych przełączników sieciowych (and. Switch).

W klasie B adres sieci jest przypisany do dwóch pierwszych oktetów i dwa ostatnie do hosta, co daje po 16 bitów na oba i możliwość podpięcia $2^{16} - 2$ hostów w każdej z $2^4 = 16$ prywatnych sieci klasy B.

Klasa C jest najczęściej wykorzystywaną klasą w sieciach lokalnych. Dzięki 24 bitom przeznaczonym na adres sieci i 8 bitom na adres hosta jego maksymalna ilość możliwych hostów w sieci wynosi 254, a prywatnych sieci klasy C jest 256 o czym decyduje przedostatni oktet (por.[7]).

Podstawowy problem przy połączeniach VPN to konflikty adresów IP. Są dwa rodzaje tych konfliktów:

- nakładanie się na siebie (ang. overlapping) zakresów adresów w różnych podsieciach łączonych poprzez VPN,
- zdalne połączenia z sieci stosujących tę samą pulę adresów co sieć, do której następuje połączenie.

Pierwsza opisywana sytuacja ma miejsce, gdy mam dwie sieci LAN, w różnych, odległych nawet lokalizacjach, które chcemy spiąć za pomocą VPN, ale używające nierozłącznych zakresów adresów IP, na przykład: 10.0.0.0/24 i 10.0.0.0/23. Pierwsza to podsieć klasy C o 256 adresach z zakresu 10.0.0.0 – 10.0.0.255. Druga sieć posiada maskę o jeden bit krótszą, a więc dopuszczającą większy zakres adresów 10.0.0.0 – 10.0.1.255. Jak widać adresy z pierwszej sieci występują w drugiej i pojawia się konflikt.

Drugi rodzaj konfliktu pojawia się na przykład wtedy, gdy jesteśmy w kafejce internetowej i próbujemy połączenia VPN z siecią w biurze. Jeśli w biurze i w kafejce używana jest ta sama popularna pula adresów 192.168.0.0/24, to nasz laptop nie będzie wiedział czy 192.168.0.1 to brama sieci lokalnej w kafejce, czy w naszym biurze.

Problem nakładających się zakresów adresów, gdy jesteśmy administratorami obu sieci, w których występuje konflikt, można rozwiązać stosując NAT (ang. Network Address Translation). Znacznie jednak komplikujemy w ten sposób konfigurację VPN i należy się liczyć ze sporymi problemami podczas diagnozowania ewentualnych awarii w sieci.

Najlepiej jest unikać konfliktów. Jeśli dopuszczamy połączenia z przypadkowych lokalizacji, nad którymi nie mamy kontroli, w sieciach LAN, którymi zarządzamy, powinniśmy stosować te zakresy adresów prywatnych, które są mało popularne, i dla których prawdopodobieństwo wystąpienia konfliktu jest niskie. Sytuacja jest znacznie prostsza, gdy administrujemy sieciami łączonymi za pomocą VPN. Wtedy mamy możliwość takiego zaplanowania adresacji

IP, aby nie powstały konflikty. Gdy tworzymy nową sieć i wydaje się, że nie będzie ona łączoną z innymi sieciami poprzez VPN, mimo to warto zadbać o unikalność adresów na wszelki wypadek. Przeadresowanie sieci LAN w średniej wielkości biurze może bowiem oznaczać sporą rewolucję i konieczność wyłączenia całej sieci na parę dni, co raczej nie spotka się z aprobatą szefostwa.

4.2 Bridge kontra router

Planując konfigurację serwera VPN musimy podjąć decyzję, czy używamy łączy typu most (ang. bridge) czy typu router.

Most, jaki mamy tutaj na myśli, jest software'owym odpowiednikiem sprzętowego przełącznika (ang. switch). O ile taki sprzętowy przełącznik łączy ze sobą fizyczne karty sieciowe, to za pomocą software'owego mostu łączymy logiczne interfejsy sieciowe (fizyczne lub wirtualne) na jednej maszynie pozwalając w ten sposób współdzielić im jedną podsieć IP. Łącząc za pomocą mostu interfejs fizyczny z interfejsem wirtualnym związanym z połączeniem VPN dwóch lokalizacji, logicznie łączymy obie sieci Ethernet w jedną, spójną całość.

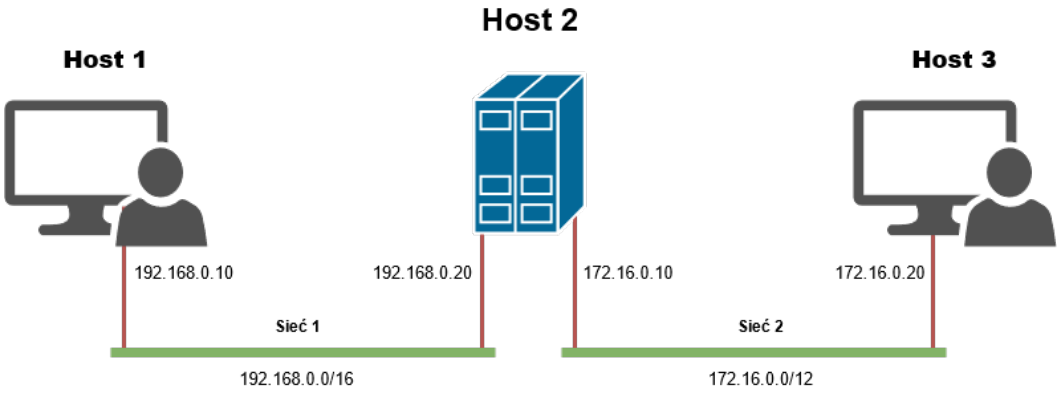
Kiedy klient łączy się za pośrednictwem mostu do sieci zdalnej, otrzymuje adres IP w tej sieci i w ten sposób może komunikować się ze wszystkimi urządzeniami w niej znajdującymi się, tak jakby był do tej sieci przyłączony bezpośrednio (za pomocą przełącznika). Przy tego rodzaju połączenia, każdy klient musi otrzymać adres IP w sieci zdalnej.

Natomiast, gdy klient łączy się za pośrednictwem routera to nie dostaje adresu w sieci zdalnej, ale nadal używa swojej sieci lokalnej, w której skonfigurowano trasy pakietów, tak aby możliwa była komunikacja z urządzeniami w sieci zdalnej. Trasy pakietów mogą być ustawione na domyślnym routerze (bramie) sieci klienta, albo bezpośrednio na nim. W tym wypadku w roli klienta może występować nie jedna maszyna, ale cała podsieć złożona z wielu maszyn.

Podstawowa, funkcjonalna różnica pomiędzy połączeniami typu most a router polega na tym, że w połączeniach typu router nie są przekazywane pakiety rozgłoszeniowe (ang. broadcast). Natomiast w połączeniach typu most nie jest możliwe korzystnie przez klienta z usług (HTTP, SMB/CIFS, SSH itp.) oferowanych w sieci zdalnej przez serwer VPN.

4.2.1 Problemy związane z metodą bridge.

Mamy dwie sieci (rys. 4.1), której Host 2 jest łącznikiem i pełni role routera. Aby połączyć Hosta 3 do tych dwóch sieci musimy zainstalować sterownik TAP, oraz OpenVPN na Host 2, oraz Host 3. Na Host 2 jako, że jest łącznikiem instalujemy serwer OpenVPN, a na Host 3 klienta. Dzięki temu zestawieniu Host 3 będzie podłączony do Sieć 1. Mimo tego, że Host 3 jest w stanie porozu-



Rysunek 4.1: Połączenie typu bridge.

mieć się z każdym hostem, to problemem tego rozwiązania jest to, że nie może połączyć się do Host 2, używając jego adresu z Sieć 1. Dodatkowo, gdy połączenie zostanie przerwane, wcześniejsze połączenie trzeba usunąć i ustawić na nowo.

4.3 TUN/TAP

TUN i TAP wykorzystuje się w połączeniach między maszynami wirtualnymi. TUN jest skrótem od "TUNel" i mieści się w trzeciej warstwie modelu ISO OSI RM, gdzie jak sama nazwa mówi odpowiada za tworzenie wirtualnych tuneli. Znalazł on swoje zastosowanie we wcześniej wspomnianym routingu. TAP natomiast znajduje się w drugiej warstwie modelu ISO OSI RM, czyli w warstwach sieciowych i zajmuje się rozpoznawaniem pakietów przychodzących jak i wychodzących. Jest używany przy połączeniach typu bridging.

4.4 Połączenie router-router (point-to-point)

Na początku połączenie point-to-point z wykorzystaniem pre-shared key było jedynym dostępnym rozwiązaniem do użycia OpenVPN. Obecnie istnieje wiele innych i lepszych rozwiązań. Połączenie te charakteryzuje się tym, że może łączyć się tylko do dwóch urządzeń jednocześnie. Gdy chcemy przesłać dane między użytkownikami dwóch różnych instytucji, najpierw dane przekazywane są od użytkownika do jego serwera OpenVPN, następnie serwer ten przekazuje informację do drugiego serwera VPN w drugim oddziale firmy, a ten zaś wysyła to do miejsca docelowego. W tym czasie nie może być wykonywana inna akcja między tymi serwerami. Jeżeli chodzi o funkcjonalność obu urządzeń to mimo, że jedno określa się jako klienta, a drugie jako serwer to ich działanie jest do

siebie mniej lub bardziej porównywalne.

- **Zalety**
  - Bardzo łatwe w konfiguracji
  - Niema potrzeby korzystania z PKI (public key infrastructure) czy certyfikatów.
  - Nie wymaga zaawansowanych urządzeń do stworzenia połączenia.
- **Wady**
  - Komunikacja jest możliwa tylko między dwoma urządzeniami przy użyciu pojedynczego połączenia.
  - Brak wsparcia dla Androida i iOS.
  - Klucz prywatny musi być skopiowany przy użyciu bezpiecznego kanału, na przykład przy pomocy SSH, co może czasami być ryzykowne w kwestiach bezpieczeństwa.
  - Słaby poziom zabezpieczeń.

4.4.1 Przykładowa konfiguracja

Najprostszym przykładem zastosowania tego typu połączenia, jest ustawienie jednego urządzenia do wysyłania żądania połączenia, a drugie w tym czasie nasłuchuje. Można tego dokonać w następujący sposób.

Najpierw ustawiamy serwer, który ma za zadanie wymuszania połączenie:

```
[MK@serwer] # openvpn \
  --ifconfig 10.200.0.1 10.200.0.2 \
  --dev tun
```

Następnie na kliencie OpenVPN wpisujemy:

```
[MK@klient] # openvpn \
  --ifconfig 10.200.0.2 10.200.0.1 \
  --dev tun \
  --remote openvpnserver.example.com
```

W tym samym czasie w innym oknie terminala na kliencie OpenVPN:

```
[root@client] # ip addr show tun0
```

I to już wszystko, teraz można sprawdzić czy połączenie jest aktywne, wykorzystując do tego opcję `ping`.

```
[MK@klient] $ ping 10.200.0.1
PING 192.168.0.1 (10.200.0.1) 32 bytes of data.
32 bytes from 10.200.0.1: icmp_seq=1 ttl=64 time= 5.21 ms
32 bytes from 10.200.0.1: icmp_seq=1 ttl=64 time= 5.23 ms
32 bytes from 10.200.0.1: icmp_seq=1 ttl=64 time= 5.11 ms
32 bytes from 10.200.0.1: icmp_seq=1 ttl=64 time= 5.45 ms

--- 192.168.0.1 ping statistics ---
4 packets transmitted, 4 recived, 0% packet loss, time 2100ms
```

4.5 Połączenie klient-serwer (client-server)

Najpopularniejszym modelem OpenVPN jest pojedynczy serwer z wieloma klientami. Model klient-serwer pierwszy raz został przedstawiony wraz z wersją OpenVPN 2.0. Każdy klient, który połączy się do serwera OpenVPN dostaje od niego przydzielony adres IP z dostępnej puli adresów. Żaden klient nie ma bezpośredniego połączenia między sobą. Cały ruch dzieje się poprzez serwer, co ma swoje plusy i minusy:

- **Zalety**
  - Serwer VPN zarządza ruchem w wirtualnej sieci. Decyduje jaki ruch może być przekazywany między klientami. Domyślnie ruch między użytkownikami jest niedozwolony, lecz za pomocą opcji client-to-client, ustawieniu zapory sieciowej, czy za pomocą określonych ról routingu, można takie połączenie umożliwić.
  - Jest prosty w konfiguracji.
- **Wady**
  - Przy dużej liczbie użytkowników, serwer ten może stracić na wydajności. Jest to spowodowane tym, że gdy chcemy połączyć się od klienta A do klienta B, najpierw musimy połączyć się z serwerem. Czyli "klient A - serwer - klient B" i na odwrót.

Najczęstszym wykorzystaniem takiego połączenia jest założenie go w firmach. Umożliwia to połączenie się z urządzeniami dostępnymi w firmie, na przykład serwery danych, drukarki, czy inne komputery w tej sieci, w sytuacji gdy jesteśmy poza siecią lokalną firmy, z domu, czy gdy jesteśmy w delegacji.

4.5.1 Przykładowa konfiguracja

Do opracowania połączenia wykorzystano środowisko CentOS w wersji 8.4.2105.

Uwierzytelnianie

Zanim będziemy mogli skonfigurować Open VPN klient/serwer, najpierw musimy zabezpieczyć nasze połączenie. Do tego przykładu użyjemy wcześniej wspomnianej infrastruktury klucza publicznego (**PKI** ang. *public key infrastructure*), który zapewnia klucze publiczne i prywatne, dla klienta i serwera.

Najlepszym sposobem na utrzymanie bezpieczeństwa tych plików jest trzymanie ich w innym komputerze, niż jest serwer. Najpierw należy skopiować potrzebne pliki do serwera. Zrobimy to używając następujących komend:

```
[MK@serwer] # mkdir -p /etc/openvpn/movpn
[MK@serwer] # chmod 700 /etc/openvpn/movpn
[MK@serwer] # cd /etc/openvpn/movpn
[MK@serwer] # PKI=<PKI_DIR>/ssladmin/active
[MK@serwer] # cp -a $PKI/ca.crt movpn-ca.crt
[MK@serwer] # cp -a $PKI/Mastering_OpenVPN_Server.crt server.crt
[MK@serwer] # cp -a $PKI/Mastering_OpenVPN_Server.key server.key}
```

Potrzebny będzie również plik parametrów Diffie-Hellman (DH), Pozwala on utworzyć klucze tymczasowe, które wykorzystywane są podczas nawiązywania połączenia. W tym celu należy wpisać:

```
[root@server] # cd /etc/openvpn/movpn
[root@server] # openssl dhparam -out dh2048.pem 2048
```

Konfiguracja połączenia klient/serwer

W celu uruchomienia serwera OpenVPN, należy utworzyć następujący plik konfiguracyjny serwera:

```
proto udp
port 9021
dev tun
server 192.168.0.0 255.255.255.0
topology subnet
persist-key
persist-tun
keepalive 10 60
```

```
dh       /etc/openvpn/movpn/dh2048.pem
ca       /etc/openvpn/movpn/movpn-ca.crt
cert     /etc/openvpn/movpn/server.crt
key      /etc/openvpn/movpn/server.key

user  nobody
group nobody  # use 'group nogroup' on Debian/Ubuntuverb 3

daemon
log-append /var/log/openvpn.log
```

Ten plik zapisujemy jako `movpn-04-01-server.conf`

Do jego uruchomienia użyjemy następującej komendy:

```
[MK@serwer] # openvpn --config movpn-04-01-server.conf
```

W tym kroku tworzymy plik konfiguracyjny dla klienta:

```
client
proto udp
remote openvpnserver.example.com
port 1194
dev tun
nobind
ca   /etc/openvpn/movpn/movpn-ca.crt
cert /etc/openvpn/movpn/client1.crt
key  /etc/openvpn/movpn/client1.key
```

I zapisujemy `jako movpn-04-01-client.conf`

Teraz należy przenieść w bezpieczny sposób pliki na komputer klienta. W tym celu można użyć komend scp:

```
[MK@klient]# mkdir -p /etc/openvpn/movpn
[MK@klient]# chmod 700 /etc/openvpn/movpn
[MK@klient]# cd /etc/openvpn/movpn
[MK@klient]# PKI_HOST=openvpnserver.example.com
[MK@klient]# PKI=<PKI_DIR>/ssladmin/active
[MK@klient]# scp root@$PKI_HOST:$PKI/ca.crt movpn-ca.crt
[MK@klient]# scp root@$PKI_HOST:$PKI/client1.crt client1.crt
[MK@klient]# scp root@$PKI_HOST:$PKI/client1.key client1.key
```

Aby uruchomić klienta OpenVPN należy użyć:

```
[MK@klient]# openvpn --config movpn-04-01-client.conf --suppress-timestamps
```

Do sprawdzenia czy konfiguracja przebiegła pomyślnie posłuży nam opcja `ping`:

```
[MK@klient] $ ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1) 32 bytes of data.
32 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time= 5.21 ms
32 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time= 5.23 ms
32 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time= 5.11 ms
32 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time= 5.45 ms

--- 192.168.0.1 ping statistics ---
4 packets transmitted, 4 recived, 0% packet loss, time 2100ms
```

Opracowano na podstawie (por. [13]).

Rozdział 5

Podsumowanie

Aspekt teoretyczny tej pracy opiera się na przedstawieniu czym jest VPN i SSH, a także o tym w jaki sposób zapewnić bezpieczeństwo w tych dwóch technologiach. Najwięcej trudności sprawiło mi rozumienie specyfiki działania czy to SSH, czy VPN, ale również problemów, jakie można spotkać, przy ich implementacji. Większość pracy opracowana jest na podstawie artykułów ze strony wikipedia, jak również z oficjalnej dokumentacji OpenVPN. Przy aspekcie praktycznym bardzo pomogła mi książka Mastering VPN, na podstawie której napisany został w większości czwarty rozdział. W pracy można znaleźć potrzebne zagadnienia, które pozwalają zrozumieć, w jaki sposób działają opisywane technologie, jak również w jaki sposób je zaimplementować.

Bibliografia

[1] Charlie Scott, Paul Wolfe, Mike Erwin,
Virtual Private Networks,
Wydawnictwo O'Reilly, 1999.

[2] Bates J. Regis,
Virtual Private Networks, Wydawnictwo TELECOM, 2000.

[3] Naganand Doraswamy, Dan Harkins,
IPSec, The New Security Standard for the Internet, Intranets, and Virtual Private Networks,
Wydawnictwo Prentice Hall PTR, 2003.

[4] Markus Feilner,
OpenVPN,
Wydawnictwo Packt Publishing, 2006.

[5] Firewall,
`https://en.wikipedia.org/wiki/Firewall_(computing`
Dostęp na dzień 01.05.2021.

[6] Mateusz Maciejczuk,
Policy routing oraz VPN nafirewallu opartym o IPFilter,
Praca licencjacka, Instytut Informatyki, Uniwersytet w Białymstoku, 2021.

[7] Private network,
`https://en.wikipedia.org/wiki/Private_network`
Dostęp na dzień 18.05.2021.

[8] Classless Inter-Domain Routing,
`https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing`
Dostęp na dzień 22.05.2021.

[9] RSA,
`https://en.wikipedia.org/wiki/RSA_(cryptosystem)`
Dostęp na dzień 01.06.2021.

[10] DSA,
`https://en.wikipedia.org/wiki/Digital_Signature_Algorithm`
Dostęp na dzień 03.06.2021.

[11] Jack Doerne, Yashvanth Kondi, Eysa Lee, Abhi Shelat,
Secure Two-party Threshold ECDSA from ECDSA Assumptions,
Wydawnictwo IEEE, 2018.

[12] Justin Peltier,
Next Generation SSH2 Implementation,
Syngress Publishing, 2009.

[13] Eric F. Crist, Jan Just Keijser,
Mastering OpenVPN,
Packt Publishing, 2015.