AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

INSTYTUT ELEKTRONIKI

Projekt dyplomowy inżynierski

Zastosowanie platformy wirtualizacyjnej do zdalnego użytkowania serwerowej przestrzeni dyskowej

The utilisation of a virtualisation platform facilitates for the remote access of server storage space

Autor: *Kacper Czepiec*
Kierunek studiów: Elektronika i Telekomunikacja
Opiekun pracy: *dr inż. Jacek Kołodziej*

Kraków, 2024

Spis treści

Spis stosowanych symboli i skrótów (ułożony alfabetycznie): ... 3
Wstęp ... 6
Cel pracy ... 7
Założenia projektowe ... 7
1. Charakterystyka technologii wykorzystanych do realizacji sieci prywatnej ... 9
1.1. Wirtualizacja ... 9
1.1.1. Wirtualizacja jako fundament działania maszyn wirtualnych ... 9
1.1.2. Wirtualizacja CPU ... 10
1.1.3. Podział wirtualizacji ... 11
1.1.4. Wirtualizacja pamięci ... 13
1.1.5. Wirtualizacja urządzeń wejścia/wyjścia ... 15
1.1.6. Zalety wirtualizacji ... 16
1.2. Internet ... 17
1.2.1. Uproszczone podejście w warstwie sieciowej ... 17
1.2.2. Protokoły warstwy transportowej ... 17
2. Model sieci prywatnej ... 19
2.1. Urządzenie Supermicro z płytą główną X8SIL-F ... 19
2.2. System operacyjny Debian ... 20
2.3. Platforma wirtualizacyjna Proxmox ... 20
2.4. Firewall (zapora ogniowa) ... 21
2.5. iptables ... 21
2.6. ebtables ... 23
2.7. Protokół TLS ... 24
2.8. Serwer VPN ... 25
2.9. OpenVPN ... 25
2.10. Infrastruktura PKI ... 25
2.11. Easy-RSA 3 ... 26
2.12. SFTP (SSH File Transfer Protocol) ... 26
2.13. OpenMediaVault (OMV) ... 27
2.14. FileZilla portable client ... 27
3. Sieć prywatna – budowa modelu ... 29
3.1. Opis infrastruktury ... 29
3.2. Opis konfiguracji poszczególnych elementów infrastruktury ... 31
3.2.1. Fizyczny serwer Supermicro ze środowiskiem Proxmox ... 31
3.2.2. Urząd certyfikacji infrastruktury klucza publicznego - VM easyRSA ... 34

3.2.3. Serwer wirtualnej sieci prywatnej - VM openVPN 36
3.2.4. Zapora sieciowa kontrolująca przesył pakietów - VM Firewall 39
3.2.5. Serwer usługi SFTP - VM OMV (OpenMediaVault) 42
3.2.6. Połączenie się z serwerem SFTP 44
3.3. Model serwerowej przestrzeni dyskowej 47
3.3.1. Analiza zestawiania tunelu w programach do obserwowania ruchu sieciowego 48
3.3.2. Zestawienie tunelu do serwera OpenVPN 50
3.3.3. Materiał wideo 52
4. Podsumowanie wyników pracy 53
5. Bibliografia 54
6. Spis załączników 56

Spis stosowanych symboli i skrótów (ułożony alfabetycznie):

3-way handshake - trójetapowa procedura zestawienia połączenia w protokole TCP
adres IP prywatny - adres protokołu IP w wersji 4 opisany w RFC1918 [1]
adres IP publiczny - adres protokołu IP w wersji 4 niebędący adresem prywatnym ani specjalnego przeznaczenia, możliwy do trasowania w Internecie
AES - (ang. *Advanced Encryption Standard*) symetryczny szyfr blokowy do szyfrowania transmisji
AMD - jeden z głównych producentów układów scalonych
CA - (ang. *Certificate Authority*) urząd certyfikacji do podpisywania certyfikatów klientów infrastruktury PKI
CLI - (ang. *Command Line Interface*) interfejs tekstowy wiersza poleceń systemu komputerowego
CMD - tekstowy wiersz poleceń systemów operacyjnych z rodziny Windows
CSR - (ang. *Certificate Signing Request*) żądanie podpisania certyfikatu wystosowane do CA
CRL - (ang. *certificate revocation list*) lista infrastruktury PKI o unieważnionych certyfikatach danego CA
cron - program do automatycznego uruchamiania zadań w systemach operacyjnych z rodziny UNIX
DHCP - (ang. *Dynamic Host Configuration Protocol*) protokół do automatycznego przesyłania ustawień sieciowych
DMA - (ang. *Direct Memory Access*) technika dostępu do pamięci z pominięciem procesora
easy-rsa - program do utworzenia i zarządzania infrastrukturą PKI na systemach Linux
easyRSA - inna nazwa maszyny wirtualnej nr 102 projektu inżynierskiego
GUI - (ang. *Graphical User Interface*) interfejs graficzny systemu komputerowego do interakcji z użytkownikiem
Intel - jeden z głównych producentów układów scalonych
IP/IPv4 - (ang. *Internet Protocol*) protokół warstwy sieciowej w wersji czwartej
ISO/OSI - jeden z modeli działania Internetu, obecnie stosowany głównie do opisu warstw sieciowych

ISP - (ang. *Internet Service Provider*) dostawca usługi Internetu

klient - (ang. *client*) najczęściej usługa inicjująca połączenie do danej usługi, np. serwera

KVM - (ang. *Kernel-based Virtual Machine*) maszyna wirtualna działająca w oparciu o pełną wirtualizację typu 1

L2 - (ang. *Layer 2*) warstwa łącza danych modelu ISO/OSI

L3 - (ang. *Layer 3*) warstwa sieci modelu ISO/OSI

L4 - (ang. *Layer 4*) warstwa transportowa modelu ISO/OSI

LAN - (ang. *Local Area Network*) sieć komputerowa działająca na niewielkim obszarze

MAC adres - (ang. *Media Access Control address*) unikatowy adres fizyczny urządzenia w warstwie łącza danych

MMU - (ang. *Memory Management Unit*) jednostka zarządzania pamięcią w systemie komputerowym

NAT - (ang. *Network Address Translation*) technika polegająca na podmianie adresów źródłowych lub docelowych pakietów

OpenVPN - protokół do zestawiania tuneli do wirtualnych sieci prywatnych

openVPN - inna nazwa maszyny wirtualnej nr 101 projektu inżynierskiego

OS - (ang. *Operating System*) system operacyjny systemu komputerowego

PKI - (ang. *Public Key Infrastructure*) infrastruktura klucza publicznego do hierarchicznego wydawania certyfikatów tożsamości

port - numer usługi w warstwie transportowej bądź inaczej interfejs

Proxmox - hipernadzorca typu 1 wykorzystywany w projekcie

QEMU - (ang. *Quick Emulator*) hipernadzorca typu 2

RAM - (ang. *Random Access Memory*) pamięć operacyjna systemu komputerowego, ulotna

RFC - (ang. *Request for Comments*) zbiór standardów opisujących działanie obecnego Internetu

RIPE - organizacja przydzielająca pule adresów publicznych w Europie i na Bliskim Wschodzie

RSA - (ang. *Rivesta-Shamira-Adlemana algorithm*) asymetryczny algorytm kryptograficzny

RTP - (ang. *Real-time Transport Protocol*) protokół transmisji danych w czasie rzeczywistym

serwer - (ang. *server*) najczęściej usługa nieinicjująca połączenia do danej usługi

SFTP - (ang. *SSH File Transfer Protocol*) protokół przesyłania plików w oparciu o warstwę protokołu SSH

SSH - (ang. *Secure Shell*) standard pozwalający na zdalną komunikację z systemem komputerowym

SSL - (ang. *Secure Socket Layer*) pierwotny protokół do szyfrowania komunikacji w Internecie, wyparty przez TLS

TCP - (ang. *Transmission Control Protocol*) połączeniowy protokół L4, zapewniający niezawodne dostarczenie danych

TCP/IP - współczesny model działania Internetu

tcpdump - program komputerowy systemów z rodziny Linux do przechwytywania pakietów internetowych

TLS - (ang. *Transport Layer Security*) protokół do szyfrowania komunikacji w Internecie, zwany także SSL/TLS

UDP - (ang. *Transmission Control Protocol*) bezpołączeniowy protokół L4, niezapewniający niezawodnego dostarczenia danych

VM - (ang. *Virtual Machine*) maszyna wirtualna

VMM - (ang. *Virtual Machine Monitor*) oprogramowanie do zarządzania maszynami wirtualnymi, hipernadzorca

VPN - (ang. *Virtual Private Network*) wirtualna sieć prywatna

VT - (ang. *Virtualization Technology*) technika wirtualizacji wydana przez firmę Intel na architekturę x86

WAN - (ang. *Wide Area Network*) sieć komputerowa działająca na dużym obszarze

Wireshark - program komputerowy do przechwytywania pakietów internetowych

X.509 - standard opisujący format certyfikatów i kluczy w infrastrukturze PKI

Wstęp

W przeciągu ostatnich 25 lat pojawiła się możliwość budowania własnej, publicznie dostępnej infrastruktury sieciowej nie tylko przez największe korporacje i centra danych, ale także i użytkowników prywatnych. Głównymi czynnikami umożliwiającymi wspomniany postęp jest rozwój technologii wirtualizacji, możliwość wykupienia całych pul adresowych protokołu IP czy obniżające się koszty podzespołów komputerowych, jak i fizycznych serwerów. Jednym z takich rozwiązań jest stworzenie domowej, wirtualnej sieci prywatnej, która umożliwia zdalny dostęp do zasobów sieciowych (zarządzanie i kontrola domem - kamery i inteligentne systemy sterowania, przestrzeń dyskowa itd.), jak i może pomóc przy omijaniu cenzury w Internecie (maskarada IP oraz przesyłanie ruchu przez szyfrowane tunele). W przypadku posiadania odpowiedniej infrastruktury (przyłączeniowej - nawiązania do sieci Internet po różnych dostawcach, realizowanie niezależnymi trasami; sieciowej - kopie zapasowe przesyłane na odrębne serwery, energetycznej - systemy zasilania awaryjnego, klimatyzowanie pomieszczeń), możliwe jest samodzielne i niezależne przechowywanie plików. Pozwala to na odseparowanie się od problemów, możliwych do wystąpienia u dostawców zewnętrznych, jak: niedostępność zasobów podczas awarii, utrata danych w przypadku błędów oprogramowania przeniesionych do wersji produkcyjnych ze środowisk testowych, a przede wszystkim znacznie redukuje ślad cyfrowy, zmniejszając ilość informacji możliwych do przechwycenia w Internecie. Rozwiązania takie wykorzystują także technologię wirtualizacji, która pozwala na logiczną separację wzajemnie współpracujących ze sobą maszyn na pojedynczym, fizycznym urządzeniu. Zapewnia ona większe bezpieczeństwo w dostępie do współdzielonych zasobów czy urządzeń peryferyjnych, a z drugiej strony zmniejsza zużycie energii oraz prawdopodobieństwo wystąpienia awarii, które głównie wynika z błędów oprogramowania aniżeli nieprawidłowego działania rzeczywistych podzespołów.

Cel pracy

Celem pracy jest utworzenie domowej infrastruktury sieciowej, dostępnej z publicznej sieci Internet, do przesyłania plików na wewnętrzny serwer SFTP, ulokowany na VM. Będzie on, wraz z innymi maszynami wirtualnymi, pełnił wyznaczone role. Są nimi: budowa zapory sieciowej, stworzenie urzędu certyfikacji do prawidłowego działania serwera OpenVPN i obsługa klientów podanego serwera. Środowiskiem maszyn wirtualnych będzie zarządzała platforma Proxmox [2] (hipernadzorca), uruchomiona na fizycznym urządzeniu Supermicro [3].

Realizacja postawionego celu pracy będzie pokazana w kolejnych rozdziałach, na wstępie zostaną omówione założenia projektowe, w pierwszym rozdziale zostaną przedstawione główne technikii użytych technologii systemów komputerowych. W drugim rozdziale rozwinięta będzie kwestia zastosowanych programów, środowisk oraz protokołów. W rozdziale trzecim zaprezentowane zostaną opisy utworzonych rozwiązań, wraz z przedstawieniem działania infrastruktury, a także materiał wideo przedstawiający działanie infrastruktury.

Założenia projektowe

Serwer SFTP ma umożliwiać na bezpieczny dostęp do domowych zasobów dyskowych poprzez wykorzystanie zewnętrznego nośnika danych. Takie rozwiązanie ma gwarantować zachowanie prywatności oraz anonimowości przez niebezpieczną i otwartą sieć, jaką jest Internet, co wskazano na Rys. 1. Wspomniany pendrive ma posiadać niezbędne pliki do: instalacji sterownika oprogramowania OpenVPN, zestawienia szyfrowanego tunelu oraz przesłania danych na serwer SFTP. Na fizycznym serwerze Supermicro z płytą główną X8SIL-F, który jest przeznaczony do bezawaryjnej pracy przez długi, nieprzerwany okres, uruchomiona będzie platforma wirtualizacyjna Proxmox.

Hipernadzorca ma zarządzać czterema maszynami wirtualnymi, które będą się składać na zestaw komplementarnych narzędzi rozwiązujących problem. Każda z nich ma mieć zainstalowany system operacyjny Debian 12 i realizować ściśle określony zestaw wytycznych. Pierwsza maszyna ma kontrolować przepływ pakietów, realizując rolę zapory sieciowej. Trzecia VM ma posiadać główny certyfikat PKI, służący do podpisywania żądań i generowania

certyfikatów dla klientów serwera VPN. Maszyna ta ma podlegać rygorystycznej kontroli dostępu już w ramach L2 i być najlepiej chronioną w tej infrastrukturze sieciowej. Druga, czyli serwer OpenVPN, ma być zależna od wcześniej wymienionych - poprzez odpowiednią konfigurację zapory sieciowej, a także dzięki wygenerowanym parom kluczy dla użytkowników, ma umożliwić klientom przyłączenie się do wirtualnej sieci prywatnej z publicznej sieci Internet. Czwarta będzie serwerem SFTP, o przydzielonej obszernej przestrzeni dyskowej, który dzięki wykorzystaniu wcześniej wspomnianej maszyny z serwerem VPN, pozwala podłączonym klientom na alokację zasobów bezpośrednio na niej.

Każda z instancji ma posiadać zestaw skryptów, które będą przyspieszać działanie i automatyzować wykonanie części zadań. Przyłączenie do serwera VPN będzie możliwe dzięki zainstalowaniu sterowników oprogramowania OpenVPN na wykorzystywanym komputerze z systemem Windows, a następnie na połączeniu się do serwera SFTP poprzez program klienta FTP.

Rys. 1 Uproszczony schemat zaproponowanego serwera

1. Charakterystyka technologii wykorzystanych do realizacji sieci prywatnej

Do zrealizowania projektu kluczową technologią jest wirtualizacja zasobów sprzetów, pozwalająca na utworzenie zależnych od siebie, ale odseparowanych instancji nazywanych maszynami wirtualnymi. Technologia ta umożliwia wykorzystanie jednego, fizycznego urządzenia i jego podzespołów do realizacji różnych, ściśle określonych zadań. Opisane zostaną metody wirtualizacji takich zasobów jak procesora, pamięci oraz urządzeń wejścia-wyjścia, które zostały bezpośrednio wykorzystane w projekcie. Zwrócono także uwagę na działanie Internetu, bez którego nie byłoby możliwe przesyłanie danych internetowych z jednej lokalizacji do drugiej. W tym procesie wykorzystywane są protokoły zdefiniowane przez organizacje międzynarodowe, dzięki którym format wysyłanych pakietów jest dokładnie taki sam, niezależnie od wykorzystywanego oprogramowania czy producenta sprzętu.

1.1. Wirtualizacja

1.1.1. Wirtualizacja jako fundament działania maszyn wirtualnych

Fundamentem pozwalającym na rozwój wirtualizacji, był zrealizowany na przełomie lat 60 XX wieku projekt komputera Atlas [4]. Wprowadził on możliwość rozszerzenia pamięci fizycznej komputera, która początkowo była w pełni zależna od wartości przydzielonej na etapie produkcji. Program uruchomiony na systemie operacyjnym, do prawidłowego działania i przetwarzania danych, wykorzystuje zadaną przestrzeń adresową, podzieloną na fragmenty zwane stronami [5]. Strony te podlegają mapowaniu na pamięć fizyczną, jednakże nie muszą znajdować się bezpośrednio w niej - mogą być zapisane na podłączonych do komputera dyskach zewnętrznych, tworząc wirtualną przestrzeń adresową. Efektem jest dołożenie kolejnych bloków pamięci, zwanych pamięcią wirtualną, które tworzą tzw. strony wirtualne. Jeśli urządzenie wspiera wirtualizację pamięci, to dane zamiast być bezpośrednio przekazywane na magistralę pamięci, są przesyłane do jednostki zarządzania pamięcią MMU. Jej zadaniem jest odwzorowanie pamięci wirtualnej, czyli wykorzystywanej przez program, bezpośrednio na adresy w pamięci fizycznej.

Pełna wirtualizacja po raz pierwszy została szczegółowo opisana już w 1974 r. [6], kiedy to Autorzy zwrócili uwagę na pewne wiążące cechy komputera VM/370. Najważniejszą z nich jest istnienie oprogramowania zarządzającego maszynami wirtualnymi, które emuluje

wykonywanie niektórych instrukcji, tworząc iluzję działania wielu instancji na pojedynczym urządzeniu. Jest to VMM, zwany także hipernadzorcą.

Każda z maszyn wirtualnych, podlegających temu rozwiązaniu, musi spełniać trzy określone założenia [7]:

- Bezpieczeństwo - definiuje, że hipernadzorca posiada absolutną kontrolę nad wirtualizowanym środowiskiem i wszelkimi zasobami,
- Wierność - działanie programu w środowisku wirtualnym powinno być identyczne jak podczas uruchomienia go na fizycznym odpowiedniku (nie licząc wyjątków związanych z czasowymi zależnościami i dostępnością zasobów systemowych),
- Wydajność - zdecydowana większość programów i instrukcji wykonywanych przez maszynę wirtualną, powinna być przeprowadzana bez udziału hipernadzorcy.

Najważniejsza sztuczka polega na ,,oszukaniu'' i przekonaniu maszyny wirtualnej, że jest ona jedyną działającą na sprzęcie fizycznym i posiada pełny dostęp zarówno do pamięci, rdzeni procesora czy urządzeń peryferyjnych. Nie powinna także odczuwać żadnych przeszkód przy wykonywaniu zleconych zadań, aby móc w wydajny sposób obsłużyć uruchamiane procesy. Przy opisywaniu typów wirtualizacji okaże się, że nowe rozwiązania niekoniecznie muszą się stosować do tego założenia.

1.1.2. Wirtualizacja CPU

Do wirtualizacji CPU, procesory posiadające tryb użytkownika oraz jądra uruchamiają niektóre instrukcje w inny sposób, w zależności od obecnego trybu. Nazywane są instrukcjami wrażliwymi i dotyczą przykładowo zmian konfiguracyjnych MMU czy operacji wejścia/wyjścia (ang. *I/O operations*). Instrukcjami uprzywilejowanymi będą za to te, które uruchomione w trybie użytkownika, powodują wywołanie rozkazu-pułapki (ang. *trap*). Sprzęt zmieni tryb na jądro każdorazowo przy aktywacji pułapki lub po wystąpieniu przerwania, w celu obsłużenia sygnału. Rezultatem będzie wniosek, że próba uruchomienia instrukcji w trybie użytkownika, która nie może być w nim wykonana, spowoduje wywołanie pułapki.

Ta właściwość, dotycząca założenia o wierności odwzorowania, nie była możliwa do zrealizowania na architekturze procesorów x86 i aż do roku 2005 uniemożliwiała sprzętową wirtualizację. Rozwiązania, wprowadzone zarówno przez Intel oraz AMD, otrzymały odpowiednie nazwy VT oraz SVM (ang. *Secure Virtual Machine*). Zapewniając prawidłową obsługę maszyn wirtualnych, hipernadzorca wspierający wirtualizację otrzymuje informację o wywołaniu rozkazu pułapki przez system operacyjny gościa. VMM następnie sprawdza, czy instrukcja została wywołana przez system operacyjny gościa, czy może przez program użytkownika maszyny. Jeśli przez OS - instrukcja jest wykonywana; jeśli przez program - hipernadzorca emuluje działanie jako instrukcję wrażliwą, uruchomioną w trybie użytkownika. Podejście to, nazywane trap and emulate, pozwala zamienić instrukcje wrażliwe na odwołania do naśladowania tych poleceń - nie są one zatem nigdy bezpośrednio wykonywane przez np. urządzenia wejścia-wyjścia.

1.1.3. Podział wirtualizacji

Podział typów wirtualizacji:

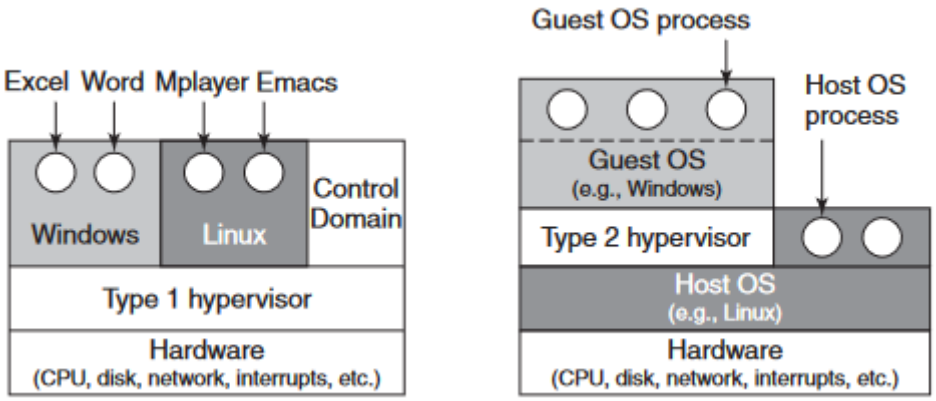
- pełna wirtualizacja
- hipernadzorca typu 1
- hipernadzorca typu 2
- parawirtualizacja
- wirtualizacja na poziomie systemu operacyjnego
- wirtualizacja poziomu procesu

Opisane wyżej metody, jak i rozwiązanie użyte w tej pracy, dotyczyć będą pełnej wirtualizacji, używając środowiska z hipernadzorcą typu 1. Pozostałe alternatywy warto jednak nadmienić, przedstawiając jak wiele się zmieniło w ciągu ostatnich dziesięcioleci w tej materii.

- hipernadzorca typu 1

Zwany jest także bare-metal, co można zdefiniować jako „działający bezpośrednio na sprzęcie”, bez warstwy abstrakcji ograniczającej go od dostępu do zasobów fizycznych urządzenia. Innymi słowy, pełni on rolę jedynego systemu operacyjnego (zwanego gospodarzem), działającego w najbardziej uprzywilejowanym trybie - jądra. Jak zostało przedstawione na Rys. 2, hipernadzorca typu 1 może uruchamiać maszyny wirtualne z różnymi

systemami operacyjnymi, np. Windows oraz dystrybucjami Linux. Maszyny wirtualne działają jako procesy użytkownika w trybie użytkownika, stąd nie mogą wykonywać instrukcji wrażliwych. Są jednak przekonane o byciu jedynym systemem na urządzeniu fizycznym, które działa w trybie jądra. Podejście to nazywane jest wirtualnym trybem jądra.



Rys. 2 Porównanie typów hipernadzorców przy pełnej wirtualizacji [8]

- hipernadzorca typu 2

Jest on zainstalowany bezpośrednio na innym systemie operacyjnym, nazywanym OS gospodarza. Uruchomione na nim maszyny wirtualne obsługują systemy operacyjne nazywane gośćmi. Działanie systemu nie różni się od opisanych wcześniej założeń dot. hipernadzorcy typu 1, natomiast problem mogą stanowić czasem sterowniki urządzeń wejścia-wyjścia - nie zawsze OS gospodarza jest w stanie prawidłowo je odczytać.

- wirtualizacja na poziomie systemu operacyjnego (ang. *OS-level virtualization*)

Technika polega na stworzeniu odseparowanych przestrzeni użytkownika w używanym systemie operacyjnym, nieco przypominając pełną wirtualizację typu 2. Takie obiekty, nazywane kontenerami (ang. *containers*), jednak nie posiadają własnego jądra systemu, więc współdzielą je z gospodarzem. Wciąż mogą mieć wydzielone zasoby fizyczne, jednak nie będą nigdy tak odizolowane, co maszyny wirtualne. Jeśli aplikacja uruchomiona w kontenerze posiada podatności bezpieczeństwa, może to skutkować osłabieniem całego systemu. Więzienia te, jak są także nazywane, są dużo trudniejsze do bezpiecznego uchronienia przed potencjalnymi atakami niż wcześniej wspomniane typy. Jednakże, posiadają szereg zalet nad

pełną wirtualizacją: zajmują dużo mniej przestrzeni dyskowej (wymagają jedynie określonych bibliotek i plików konfiguracyjnych do uruchomienia), przez co są dużo łatwiejsze do przenoszenia między środowiskami, uruchamiają się nawet kilkanaście razy szybciej od maszyn wirtualnych, obsługują tylko pojedyncze aplikacje, zaś kontenery są od siebie odseparowane i niezależne (ale niekoniecznie w pełni od jądra).

- parawirtualizacja

W porównaniu do innych hipernadzorców, parawirtualizacja pozwala na modyfikację systemu operacyjnego gościa. Dzięki temu, zamiast wykonywać wrażliwe instrukcje przez OS gościa, wywoływane są hiperwywołania do OS gospodarza. Przez to OS gościa działa jak program, wykonujący wywołania systemowe (ang. *system calls*). Dla tego typu nie istnieje wspomniana wcześniej iluzja o niewiedzy systemu operacyjnego gościa, że jest jednym z wielu aktywnych programów; jest świadom nieposiadania dostępu do np. urządzeń peryferyjnych OS. Wszelka komunikacja ze swoim nadzorcą jest jawna i wykonywana przez interfejs API (ang. *Application Programming Interface*) systemu operacyjnego gościa.

- wirtualizacja poziomu procesu

Podobnie jak przy technice parawirtualizacji, emulowanie zachowana aplikacji przez VMM jest znane obu stronom. Ta technika pozwala na uruchamianie programów bądź procesów powiązanych z wyłącznie jednym systemem operacyjnym na innym. Przykładem jest oprogramowanie Wine, które umożliwia włączanie aplikacji wyłącznych dla np. OS Windows, na systemach POSIX (jak macOS czy Linux), dzięki wykorzystaniu specjalnej warstwy kompatybilności.

1.1.4. Wirtualizacja pamięci

Wspomniana wcześniej pamięć wirtualna jest odwzorowaniem wirtualnej przestrzeni adresowej na adresy w pamięci fizycznej. Jeśli system operacyjny gościa chce wykonać mapowanie wirtualnych stron na fizyczne (np. po uruchomieniu jakiegoś procesu), to zostanie wywołana instrukcja wrażliwa, obsługiwana przez VMM. Jeśli żądane strony fizyczne są niewykorzystane, to zostają przydzielone; jeśli jednak używa ich inna VM, to hipernadzorca jest zmuszony znaleźć wolne strony fizyczne, a także stworzyć kolejne odwzorowania (tablice

stron) między pamięciami, zwane tablicami stron cieni. Obecnie stosowane są rozwiązania sprzętowego wsparcia procesu, zwane zagnieżdżonymi tablicami stron (przez AMD) oraz rozszerzonymi tablicami stron (przez Intel). Jeśli wymagane jest znalezienie określonego odwzorowania w pamięci, to sprzętowo urządzenie po kolei sprawdza znane tablice stron. Następnie, przeglądane są rozszerzone tablice stron, zawierające tablice stron cieni. Dzieje się to bez udziału programu oraz bez wymuszania obsługi przez hipernadzorcę poprzez brak istnienia pułapek, odwołań czy hiperwywołań.

Bez stosowania wspomnianej techniki, możliwe są także odmienne rozwiązania, które jednak zwiększają znacznie szansę na wystąpienie błędów zwanych błędami stron. Takie omyłki prowadzą do tzw. zakończeń VM (ang. *VM exits*), kiedy to hipernadzorca ratując się od uszkodzeń, przechodzi przez szereg procesów do odzyskania kontroli. Są one na tyle kosztowne, np. przekładając się na czas trwania wielu tysięcy cykli, że przedstawione wyżej rozwiązanie znacznie poprawiło działanie maszyn wirtualnych.

Przy technologii wirtualizacji możliwe jest, jak i powszechne, przydzielenie większej ilości pamięci maszynom niż fizyczna pojemność przestrzeni adresowej zastosowanych kości. Technika ta nazywana jest *overcommitment*. Inna metoda, zwana deduplikacją, pozwala zaoszczędzić zasoby pamięci poprzez współdzielenie stron o jednakowej treści, np. jądra Linux czy wspólnych procesów. Realizowana jest poprzez obliczanie haszów stron w pamięci - jeśli dodatkowo mają jednakową treść, to finalnie zostaje wykorzystana jedna strona fizyczna, posiadająca odwołania do zainteresowanych maszyn wirtualnych. W przypadku edycji współdzielonej strony w pamięci przez którąś z jednostek, nie jest modyfikowana oryginalna treść, a zmiana jest widoczna wyłącznie dla poprawiającego.

Dynamiczne zarządzanie zaalokowaną pamięcią możliwe jest przez technikę zwaną balonikowaniem. Każda maszyna wirtualna ma załadowany taki moduł, który komunikuje się z hipernadzorcą. W przypadku znacznego zajmowania przydzielonej pamięci, system operacyjny gościa sam decyduje które już zapisane strony należy odrzucić, by zwolnić zasoby na kolejny przydział. Ten sposób jest wygodny i praktyczny, gdyż wyklucza możliwość usunięcia ważnych i używanych stron, gdyby jednostką decyzyjną był OS gospodarza (VMM).

Dzięki istnieniu MMU, próba dostępu do przestrzeni adresowej (mapowanych stron) jednej maszyny wirtualnej przez drugą, zostaje zakończona błędem. Pozwala to na zabezpieczenie jednostek przed nadpisywaniem czy modyfikowaniem nie swoich zasobów.

1.1.5. Wirtualizacja urządzeń wejścia/wyjścia

Możliwa jest także wirtualizacja urządzeń peryferyjnych, czyli wejścia-wyjścia (ang. *I/O devices*). System operacyjny gościa wykonuje skanowanie sprzętu poprzez wywołanie instrukcji wrażliwej, zwracającej rozkaz-pułapkę do hipernadzorcy. Gdy OS maszyny wirtualnej dostaje odpowiedź, to ładuje sterowniki, próbujące nawiązać połączenie z I/O. Jest to kolejna instrukcja wrażliwa, wskazująca że komunikacja urządzeń może być zależna od VMM. Wyróżniane są 3 główne sposoby wirtualizacji peryferiów, poprzez zastosowanie: I/O MMU, domen urządzeń i SR-IOV (ang. *Single Root I/O Virtualization*).

- Wykorzystanie I/O MMU rozwiązuje problem używania DMA - korzysta on z tablic stron do mapowania pamięci, gdzie próba dostępu do nieodwzorowanych stron powoduje błąd. Dzięki temu, zapewniona jest izolacja między maszynami wirtualnymi, co zapobiega nadpisaniom bądź zmianom nie swojej przestrzeni adresowej. Tworzy także niejawny interfejs do komunikacji VM oraz urządzeń peryferyjnych, mapując przestrzenie adresowe obu stron (jeśli są niejednakowych rozmiarów) czy umożliwiając remapowanie przerwań (np. dopasowując komendy do uruchomionego procesora na maszynie wirtualnej).
- Domeny urządzeń podchodzą inaczej do problemu - utworzona jest osobna, specjalnie dedykowana maszyna wirtualna do przetwarzania operacji wejścia-wyjścia, pełniąca rolę przekaźnika - nazywana jest domeną 0. Widoczna jest także tutaj przewaga hipernadzorców typu 2 w porównaniu do typu 1: urządzenia peryferyjne korzystają ze sterowników zlokalizowanych na OS gospodarza, nie musząc instalować ich bezpośrednio na maszynie wirtualnej albo wykonywać odwołania do domeny 0.
- SR-IOV umożliwia pominięcie hipernadzorcy przy komunikacji VM ze sprzętem. Urządzenia wspierające tę technologię posiadają własną przestrzeń adresową, strumienie DMA czy przerwania dla każdego klienta. Dzięki temu, wszystkie hosty wierzą, że mają jedyny i bezpośredni dostęp do urządzenia, a także pomijane jest obsługiwanie rozkazów-pułapek.

Dzięki zastosowanym technikom, możliwe jest także modernizowanie fizycznych komponentów na uruchomionym środowisku, bez wiedzy maszyn wirtualnych. Jeśli nowo zamontowany dysk wspiera nowsze technologie niż dostępne u poszczególnych VM, to hipernadzorca przekształca komendy na znane maszynie wirtualnej. Dzięki tej właściwości, realne jest ulepszanie sprzętu przy zachowaniu jednakowego oprogramowania, bez potrzeby jego adaptacji do nowego rozwiązania.

Hipernadzorca może także pełnić rolę wirtualnego switcha, gdzie każda maszyna wirtualna posiada własny adres MAC, a VMM przełącza ramki między maszynami. Ta cecha będzie aktywnie wykorzystana w ramach tego projektu.

1.1.6. Zalety wirtualizacji

Wirtualizacja była krokiem milowym, z punktu widzenia zarządzania sprzętem i kwestią finansów, dla wielu firm, umożliwiając szereg korzyści jak:

- oszczędność zasobów komputerowych i zużycia energii,
- wydajniejsze wykorzystanie używanych urządzeń,
- zmniejszenie wykorzystywanej powierzchni serwerowej.

Z perspektywy rozwoju technologii, wprowadziła:

- wygodny dostęp do środowisk testowych dla deweloperów tworzących oprogramowanie na różne platformy,
- utworzenie chmury obliczeniowej,
- niemalże bezprzerwową dostępność krytycznych usług przez kolokowanie urządzeń w profesjonalnych centrach danych,
- możliwość uruchamiania wielu maszyn wirtualnych na jednym sprzęcie fizycznym,
- możliwość uruchamiania nowych sprzętów (dyski czy urządzenia peryferyjne) bez konieczności zmiany oprogramowania.

1.2. Internet

1.2.1. Uproszczone podejście w warstwie sieciowej

Obecnie Internet jest zdominowany zarówno przez model TCP/IP, jak i technikę komutacji pakietów. Jego działanie można przyrównać do sposobu przesyłania poczty. Nadawca wysyłając przesyłkę do innego podmiotu (np. w innym kraju), oznacza ją poprzez umieszczenie tam swoich danych, jak i adresata. Jeśli przedmiotów jest za dużo, należy je rozdzielić na więcej mniejszych pakunków. Taka paczka, przekazana do urzędu pocztowego, czeka na swoją kolej, gdzie w trakcie przetwarzania wybierana jest kolejna placówka i sposób dostarczenia, np. samochodem, samolotem czy koleją. Przesyłka analogicznie przechodzi pomiędzy wieloma oddziałami, by finalnie dotrzeć do zainteresowanego. Odnosząc się do uproszczonego przykładu, w warstwie sieciowej modelu ISO/OSI karty sieciowe tworzą pakiety, umieszczając w nich parametry jak adresy IP źródłowe i docelowe. Jeśli takie paczki danych są za duże, to podlegają fragmentacji. Wysyłany pakiet przechodzi przez kolejne urządzenia zwane routerami, które po analizie nagłówka L3 decydują do jakiego następnego urządzenia ma taki pakiet być przesłany. Decyzje podejmowane przez każdy z routerów na trasie, są niezależne i dopasowywane dynamicznie, np. przy niedostępności tras głównych podczas awarii. Końcowo, taki pakiet dociera do karty sieciowej adresata, umożliwiając przetwarzanie otrzymanych danych.

1.2.2. Protokoły warstwy transportowej

Nie można mówić o dzisiejszym Internecie bez wspomnienia o takich protokołach jak UDP oraz TCP. UDP jest protokołem bezpołączeniowym. Oznacza to, że inspekcji na zaporze sieciowej podlega każdy pakiet z osobna. Jest szybszy od TCP (ma krótszy nagłówek). W przypadku TCP, pierwsze musi być zestawione połączenie między stronami, zgodnie z procedurą zwaną 3-way handshake. Następnie, w zależności od ustawionych parametrów (flag), dane mogą być przesyłane w oparciu o stan połączenia, gdyż podlegają już nawiązanym i śledzonym wpisom. Pomimo, że taki sposób jest dużo bardziej obciążający dla procesora niż przy wykorzystaniu protokołu UDP, TCP gwarantuje prawidłową kolejność otrzymanych segmentów i zapewnia ich retransmisję.

UDP wykorzystywany jest głównie przy przesyłaniu obrazu na żywo czy przy niektórych grach wideo, zaś TCP wszędzie, gdzie ważne jest nawiązanie pewnego połączenia, np. przy

przeglądaniu sieci (bankowość internetowa) czy przy przesyłaniu plików. Oba protokoły korzystają z portów, czyli identyfikatorów wykorzystywanych usług. Istnieją tzw. ogólnie znane porty (ang. *well-known ports*), które standardowo są przypisane do najpowszechniejszych usług, jak 443 do protokołu HTTPS czy 22 do SSH.

2. Model sieci prywatnej

Zgodnie z założeniami projektowymi, sieć prywatna jest zbudowana w oparciu o serwer Supermicro hostujący platformę wirtualizacyjną Proxmox. Na owym hipernadzorcy uruchomione są maszyny wirtualne z systemem operacyjnym Debian 12. Do kontroli przepływu pakietów w poszczególnych warstwach modelu TCP/IP wykorzystano programy ebtables oraz iptables. Maszyna wirtualna „easyRSA" zarządza wydawaniem certyfikatów infrastruktury klucza publicznego, dzięki oprogramowaniu o jednakowej nazwie. Pozwala to na przyłączenie się klientów do serwera OpenVPN poprzez wykorzystanie wydanego certyfikatu, gdzie przesyłane dane do serwera SFTP chronione są przez używany protokół TLS.

2.1. Urządzenie Supermicro z płytą główną X8SIL-F

Fizyczny serwer widoczny na Rys. 3, hostuje środowisko do pełnej wirtualizacji typu 1. Płyta główna operuje na procesorze Intel Xeon 3420 o 4 wątkach i 4 rdzeniach. Maszyny wirtualne, jak i hipernadzorca uruchomione są na podzielonym dysku HDD WDC WD10JPCX-24U 1 TB. Sprzęt jest wyposażony w dwa interfejsy Ethernet, gdzie jeden z nich ustawiony jest jako interfejs WAN do połączenia z internetem. Zgodnie z dokumentacją, Supermicro wspiera wirtualizację VT-d od Intel, a także remapowanie przerwań przez hipernadzorcę, dzięki wykorzystaniu tablic DMAR (ang. *DMA remapping*) ACPI. Dzięki temu, w projekcie wirtualizowane są urządzenia peryferyjne poprzez użycie I/O MMU.

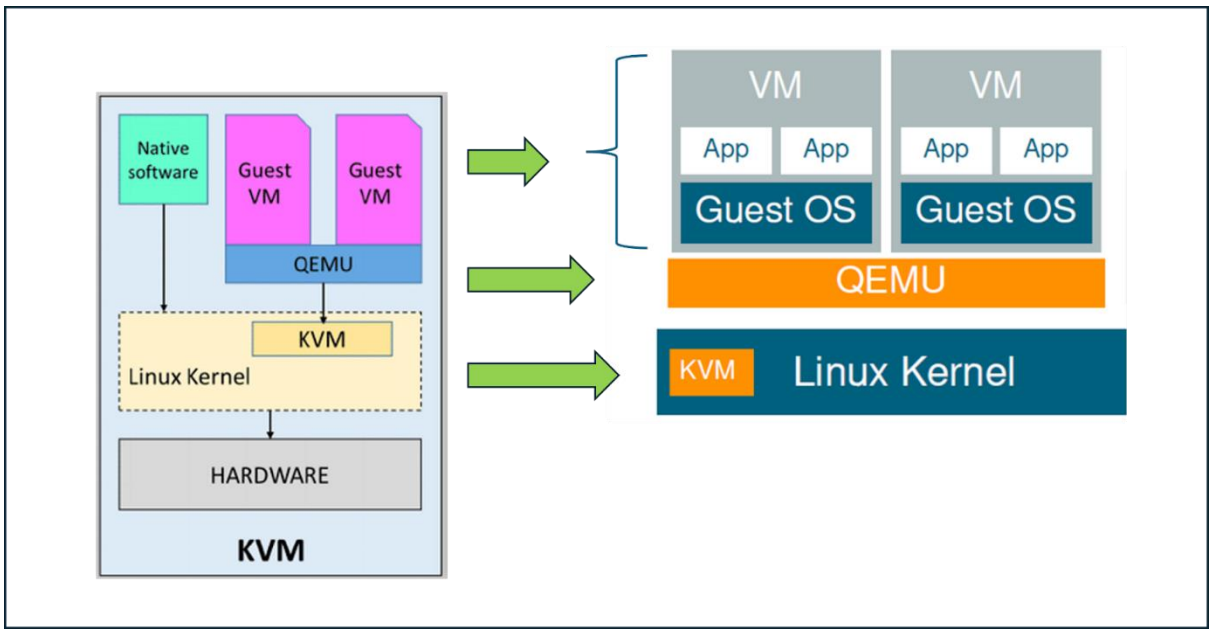
Rys. 3 Fizyczne urządzenie Supermicro wykorzystane w projekcie

2.2. System operacyjny Debian

Wszystkie maszyny wirtualne, jak i hipernadzorca, wykorzystują jądro dystrybucji Debian. Te pierwsze - najnowszą wersję 12, czyli Bookworm; ten drugi korzysta ze zmodyfikowanego jądra, ale na bazie wspomnianego systemu operacyjnego. Powodem zdecydowania się na takie rozwiązanie jest większa prostota użytkowania w środowisku wirtualnym (te same komendy, podstawowe konfiguracje itp.), wysoka stabilność systemów, ogromna liczba dostępnych paczek oraz duża popularność wśród społeczności.

2.3. Platforma wirtualizacyjna Proxmox

Hipernadzorca typu 1. Oprogramowanie służy do wirtualizacji zasobów (pamięć RAM, przestrzeń dyskowa, urządzenia peryferyjne) i przydzielaniu ich maszynom wirtualnym KVM lub kontenerom LXC (ang. *Linux containers*). Jest udostępniany w ramach darmowej wersji z otwartym kodem źródłowym (ang. *open-source*). Systemy operacyjne VM w tym projekcie są uruchamiane w oparciu o typ procesora kvm64. Pomimo, że Proxmox wykorzystuje QEMU do zarządzania środowiskiem i emulacji sprzętu (np. procesora kvm64), wirtualizacja odbywa się wyłącznie dzięki KVM i stąd uznawany jest tako VMM typu pierwszego. Zostało to przedstawione na Rys. 4.



Rys. 4 Budowa logiczna środowiska Proxmox [9]

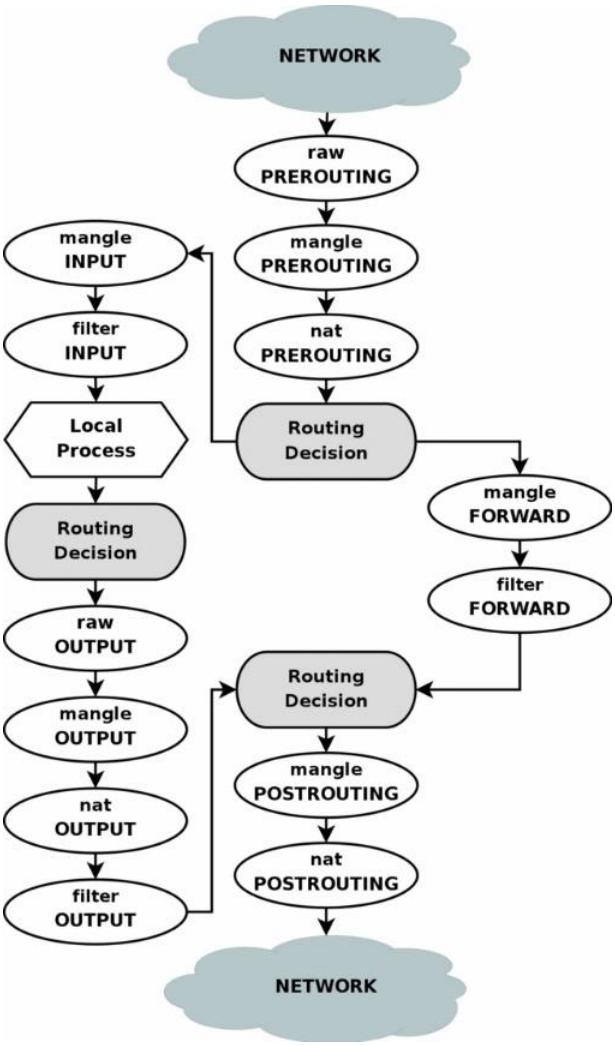
2.4. Firewall (zapora ogniowa)

Wykorzystuje oprogramowanie, jak iptables, do kontroli i zarządzania przepływem pakietów warstwy sieciowej oraz transportowej modelu ISO/OSI przez interfejsy fizyczne i logiczne kart sieciowych. Definiuje autoryzację, czyli uprawnia poszczególne maszyny wirtualne do nawiązywania określonych połączeń między sobą, jak i z publiczną siecią Internet. Reguły zapory są przetwarzane łańcuchowo, sekwencyjnie, stąd ich kolejność ma ogromne znaczenie przy analizie. Nieprawidłowo ulokowane wpisy mogą umożliwić nieautoryzowany dostęp do systemu. Program iptables (L3/L4) jest skonfigurowany na specjalnie przygotowanej do tego maszynie wirtualnej na platformie Proxmox, natomiast ebtables (L2) bezpośrednio na środowisku Proxmox.

2.5. iptables

Program Linux do kontroli ruchu sieciowego w ramach warstwy sieciowej oraz transportowej modelu ISO/OSI. Używany protokół to IPv4.

Zgodnie z dokumentacją [10], iptables nie śledzi ciągów pakietów pod kątem zawartości czy logicznego sensu transmitowanych danych, a jedynie sprawdza czy należą do tego samego potoku - jak stos TCP/IP. W projekcie wykorzystywane są tablice filter (łańcuchy: INPUT, OUTPUT, FORWARD) oraz nat (łańcuchy: PREROUTING, POSTROUTING, OUTPUT). Tablica filter służy do dopuszczania bądź odrzucania ruchu pomiędzy określonymi adresami IPv4 czy portami definiującymi usługi. Wspomniane łańcuchy należy rozpatrywać z punktu widzenia urządzenia, na którym uruchomiony jest program. Tablica nat odnosi się do translacji sieci IPv4 między interfejsem WAN, a klientami LAN w mostku Linux, umożliwiając hostom dostęp do Internetu, a także przekierowanie usług na portach. Podmieniany jest wtedy któryś z adresów IPv4 lub port. Istnieje określona kolejność przetwarzania pakietów w ramach inspekcji pakietu, co zostało pokazane na Rys. 5.



Rys. 5 Procedura przetwarzania pakietu w programie iptables [11]

Każdy pakiet przychodzący musi być sprawdzony w łańcuchu prerouting. Tutaj są podejmowanie decyzje co do śledzenia połączenia (tablica raw), oznaczania pakietów lub zmian pól TOS (ang. *terms of service*; tablica mangle) oraz przekierowania portów skierowanych do określonej sieci (tablica nat). Następnie, sprawdzany jest kierunek przekazania pakietu. Jeśli jest on skierowany bezpośrednio do urządzenia, to zostaje przetwarzany i ewentualnie filtrowany na łańcuchu input; jeśli będzie przeznaczony do innej sieci (poprzez brak adresu na interfejsach sprzętu przetwarzającego), to zostanie skierowany do łańcucha forward. Stamtąd przechodzi przez kolejne procesy segregacji aż do łańcucha postrouting, gdzie w tablicy nat może być podjęta decyzja, np. do wykonania maskarady. W przypadku pakietu wychodzącego z urządzenia, podejmowane są ustalenia co do trasowania pakietu. Po nich, pakiet jest analizowany przez kolejne tablice łańcucha output, analogicznie

jak w przypadku preroutingu, a po kolejnych inspekcjach w łańcuchu postrouting, zostaje wysłany na określony interfejs urządzenia.

Każdy z przetwarzanych pakietów TCP, może być także w jednym z 4 stanów [12]:

- new
  Jest to pierwszy pakiet dotyczący nieistniejącego wpisu w tablicy śledzenia połączeń.
- established
  Dotyczy ruchu dla znanych połączeń (tj. ruch był zarówno otrzymany, jak i wysłany przez obie strony).
- related
  Odnosi się do połączeń, które są związane z istniejącym, np. głosowa transmisja, zestawiona sesja na porcie 5060 w ramach komunikacji protokołu SIP, gdzie dane dźwiękowe przesyłane są poprzez protokół RTP (ang. *Real-time Transport Protocol*) na innych, dynamicznie przydzielanych portach.
- invalid
  Pakiet ten nie posiada stanu, np. poprzez błędy urządzenia przetwarzającego; jest najczęściej całkowicie odrzucany na zaporach sieciowych.

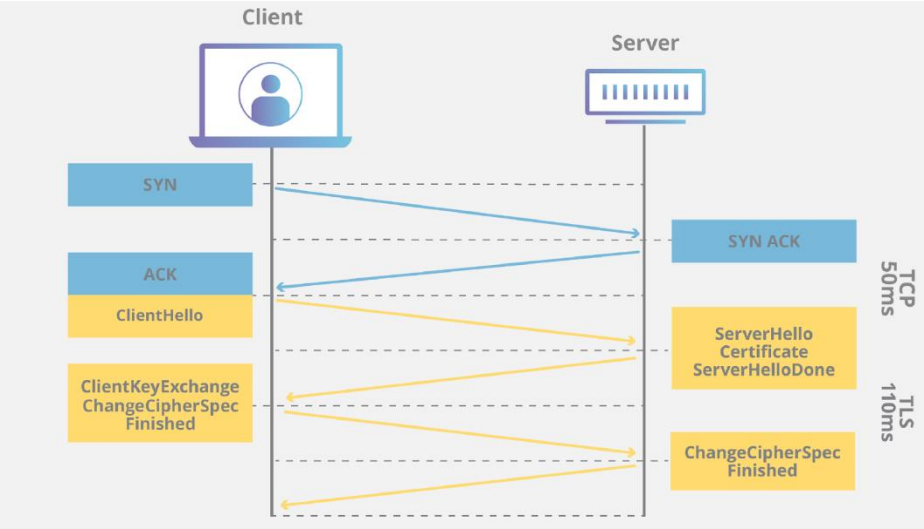
2.6. ebtables

Pomimo, że iptables można wykorzystać także do inspekcji ramek, program ten jest uruchomiony w projekcie bezpośrednio na hipernadzorcy. Działa on na poziomie warstwy 2 modelu ISO/OSI i dotyczy ruchu przesyłanego w ramach mostów Linux (ang. *bridge*). Jak zostało wspomniane w ramach wirtualizacji urządzeń wejścia-wyjścia, VMM może działać jako wirtualny switch, który przełącza ramki między maszynami wirtualnymi. Jeśli chodzi o ruch przesyłany z fizycznej karty sieciowej urządzenia Supermicro do innych hostów w Internecie, to iptables jest wystarczający do kontroli przesyłania danych, nawet jeśli działa na jednej z VM. Czasem jednak, w ramach warstwy łącza danych, nie jest w stanie w pełni odrzucić ruchu bezpośredniego między VM, stąd program ebtables będzie niezbędny do pełnej separacji maszyn wirtualnych. W tym projekcie dotyczy głównie izolacji maszyny z PKI.

2.7. Protokół TLS

Jest to protokół sieciowy zapewniający ochronę jednostek w ruchu sieciowym warstwy transportowej [13], dzięki potwierdzaniu tożsamości (obiekt jest tym, za kogo się podaje), zachowaniu integralności przesyłanych danych (nie zostały one zmienione) oraz poufności (tylko strony uczestniczące w wymianie informacji są w stanie wyświetlić zawartość). Przez wykorzystanie kryptografii asymetrycznej, możliwe jest spełnienie wymienionych zasad, a protokół znajdzie zastosowanie jako fundament przesyłania danych w sieciach VPN.

Jak wskazano na Rys. 6, TLS składa się z dwóch procedur: handshake, do uzgodnienia warunków połączenia, oraz record, do ochrony pakietów zgodnie z wynegocjowanymi parametrami. Zakładając stosowanie certyfikatów, zainteresowany klient wysyła wiadomość „Client Hello" do serwera, przekazując informacje o wspieranych wersjach TLS, algorytmach kryptograficznych i nawiązując sekretny klucz. Dostaje informację zwrotną, czyli „Server Hello" z certyfikatami i ustalonymi szczegółami połączenia (wybrany algorytm szyfrowania z zestawu zwanego ang. *cipher suite*), a także weryfikuje tożsamość serwera. Po takim procesie, zwanym TLS handshake, hosty wysyłają obustronne potwierdzenia zestawienia połączenia. Są one oznaczone jako *finished*, a od momentu otrzymania urządzenia są w stanie bezpiecznie wymieniać między sobą dane. Podczas wymiany informacji w ramach *handshake*, strony mogą także przesyłać swoje podpisy do weryfikacji tożsamości.



Rys. 6 Procedura handshake protokołu TLS [14]

2.8. Serwer VPN

Jest to usługa nieinicjująca połączenia TLS, ale odpowiadająca na takie żądania. Tworzy tunel, czyli połączenie (najczęściej szyfrowane) pomiędzy dwoma sieciami, wykorzystując protokoły jak L2TP, IPsec, PPTP, openVPN. Może być ono realizowane zarówno przez publiczną sieć Internet, jak i prywatny Intranet, do łączenia oddalonych od siebie lokalizacji, jeśli tylko infrastruktura sieciowa na to pozwala. Interfejs serwera jest w stanie pełnić rolę bramy sieciowej dla zestawionych tuneli, przekierowując cały ruch od jednego hosta przez swoje porty poprzez wstrzyknięcie odpowiednich tras. Realizowane jest to przez opcję 121 serwera DHCP, którą także może wykorzystać potencjalny atakujący [15], więc dlatego też przesyłane pliki oraz ruch będą szyfrowane.

2.9. OpenVPN

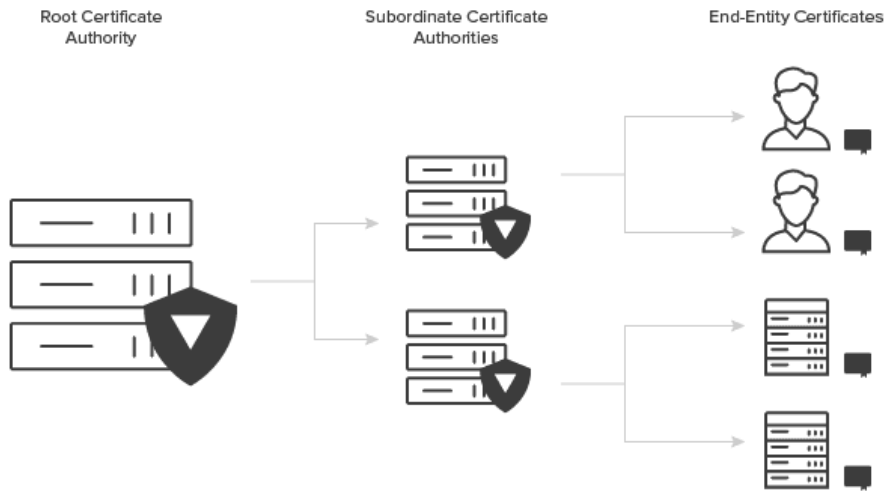
Oprogramowanie otwartoźródłowe, wykorzystujące SSL/TLS z AES256 do budowania wirtualnych sieci prywatnych. Poprawnie skonfigurowany jest jednym z najpewniejszych rozwiązań na rynku, wraz z protokołem WireGuard, dzięki swojej popularności i przemyślanej implementacji, zmniejszającej szanse na ewentualne wykorzystanie podatności. OpenVPN korzysta z otwartoźródłowej biblioteki openSSL do generowania par kluczy, podpisywania certyfikatów itd.

2.10. Infrastruktura PKI

Jest to model określający metody, protokoły, procesy zarządzania i przydzielania certyfikatów w standardzie X.509. Główną rolę, na szczycie hierarchii pełnią root CA, czyli najbardziej zaufane instytucje certyfikacji na świecie.

W typowej implementacji PKI, przedstawionej na Rys. 7, root CA podpisuje certyfikaty dla intermediate CA (sub-ca), a te dla issuing CA. Te wydają kolejne zaświadczenia dla firm zajmujących się dystrybucją takich zezwoleń dla klientów indywidualnych. Certyfikat root CA wgrywany jest już na poziomie implementacji produktu, będąc wysłanym producentom oprogramowania czy urządzeń. Następnie, issuing CA posiadają własne certyfikaty podpisane przez urzędy wyżej w hierarchii, w ramach procesu CSR. W taki sam sposób klienci chcący wykorzystywać PKI w swoich rozwiązaniach, wysyłają swoje CSR do issuing CA lub serwerów im podległych. Poprzez wprowadzenie takiego łańcucha, możliwe jest odpowiednie zarządzenie przydzielaniem certyfikatów i weryfikacją ich poprawności.

W przypadku, gdy któryś z nich zostanie skompromitowany (u klientów indywidualnych, ale w wyjątkowych okolicznościach może dotyczyć także i całego sub-), taki certyfikat zostaje wpisany na CRL i informacja o jego nieważności zostaje rozgłaszana innym podmiotom. Jest to przeprowadzane albo poprzez protokół OCSP (ang. *Online Certificate Status Protocol*) albo poprzez sprawdzenie CRL, z czego z pierwszego wycofują się niektóre CA (np. Let's Encrypt) ze względów bezpieczeństwa prywatności. Problemem jest jawna możliwość otrzymania informacji, jaki dokładnie adres IP odwiedza żądaną witrynę.



Rys. 7 Hierarchia typowej infrastruktury klucza publicznego [16]

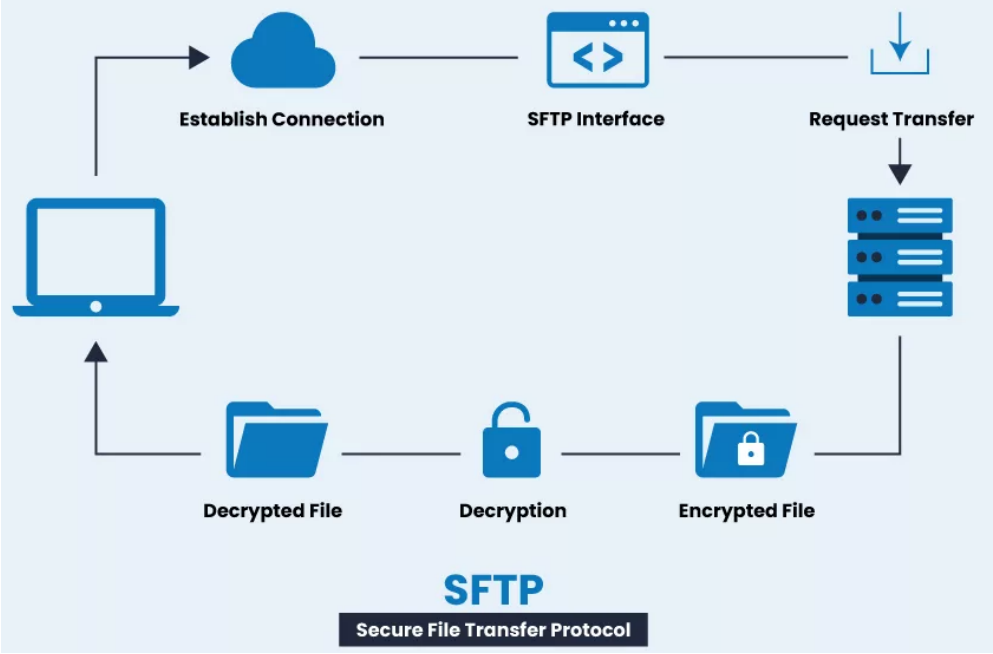
2.11. Easy-RSA 3

Oprogramowanie służące do utworzenia i zarządzania PKI [17], zalecane przez protokół OpenVPN. Każdy wystawiony certyfikat może posiadać jeden z trzech określonych typów: klient, serwer oraz ca (ostatni służy jako sub-ca, do ewentualnego kolejkowania podrzędnych CA).

2.12. SFTP (SSH File Transfer Protocol)

Protokół przesyłania plików między hostami w zestawionym połączeniu, najczęściej dwukierunkowo (full-duplex). Pomimo, że RFC nigdy nie został uchwalony [18], protokół jest dosyć powszechnie wykorzystywany. Transmisję zabezpiecza warstwa SSH, aniżeli jest to w przypadku często mylonego protokołu FTPS, który korzysta z certyfikatów TLS i szyfruje

także dane logowania. Wykorzystuje inny port warstwy transportowej niż używany do dostępu do maszyny za pomocą usługi SSH. Plik jest bezpiecznie rozszyfrowywany i przesyłany na urządzenie klienta, co wskazane zostało na Rys. 8.



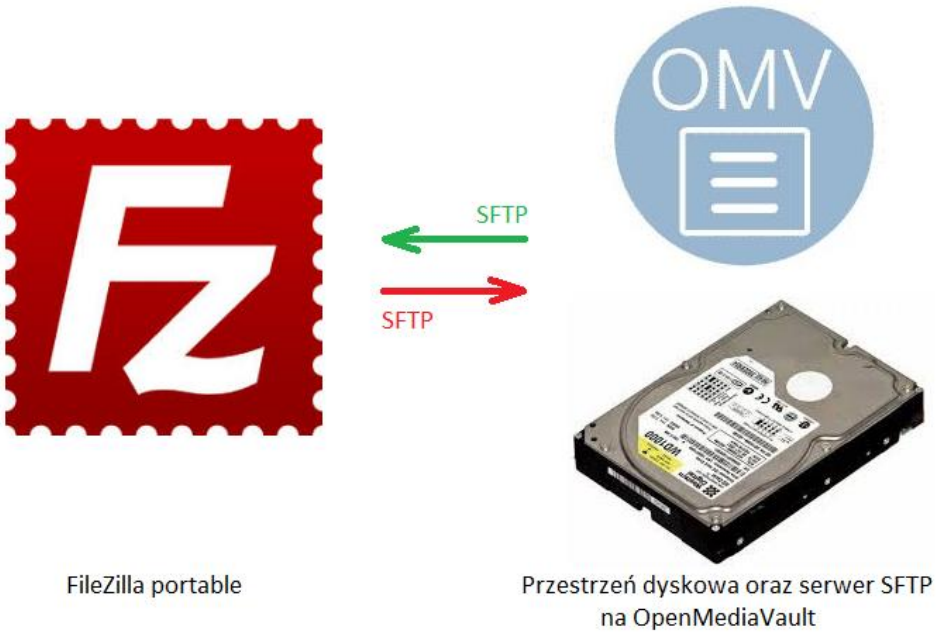
Rys. 8 Schemat przesyłania plików w ramach protokołu SFTP[19]

2.13. OpenMediaVault (OMV)

Oprogramowanie wykorzystuje technologię NAS (ang. *Network Attached Storage*), pozwalając uzyskiwać dostęp do zasobów dyskowych klientom nawiązującym połączenie do serwera. Jest ono w pełni publiczne i darmowe, zbudowane na podstawie zmodyfikowanego jądra OS Debian. Serwer SFTP jest ładowany do modułu jako gotowy plugin.

2.14. FileZilla portable client

Lekki program, będący klientem FTP, który łączy się do serwera pamięci masowej. Uwierzytelnienie jest wykonane zarówno przez podanie hasła i loginu użytkownika, jak i certyfikatu do połączenia dzięki abstrakcji warstwy SSH. Wersja portable, czyli przenośna, może być uruchomiona na zewnętrznej pamięci flash, uzyskując przepustowość zależną od wykorzystanego interfejsu USB, przepustowości karty sieciowej czy wykorzystanego systemu plików na urządzeniu.



Rys. 9 Schemat komunikacji pomiędzy wykorzystanymi programami

Zgodnie z uproszczonym zestawieniem przedstawionym na Rys. 9, klient FileZilla komunikuje się z serwerem SFTP, uruchomionym na maszynie wirtualnej. Dzięki temu możliwe jest bezpieczne przesyłanie plików w ramach stworzonej sieci prywatnej. Same strzałki opisujące wykorzystywany protokół przesyłania plików można uznać za interfejs – w rzeczywistości musi jeszcze nastąpić zestawienie tunelu VPN między kartą sieciową urządzenia, na którym uruchamiany jest klient FileZilla (np. komputer stacjonarny), a serwerem OpenVPN.

3. Sieć prywatna – budowa modelu

3.1. Opis infrastruktury

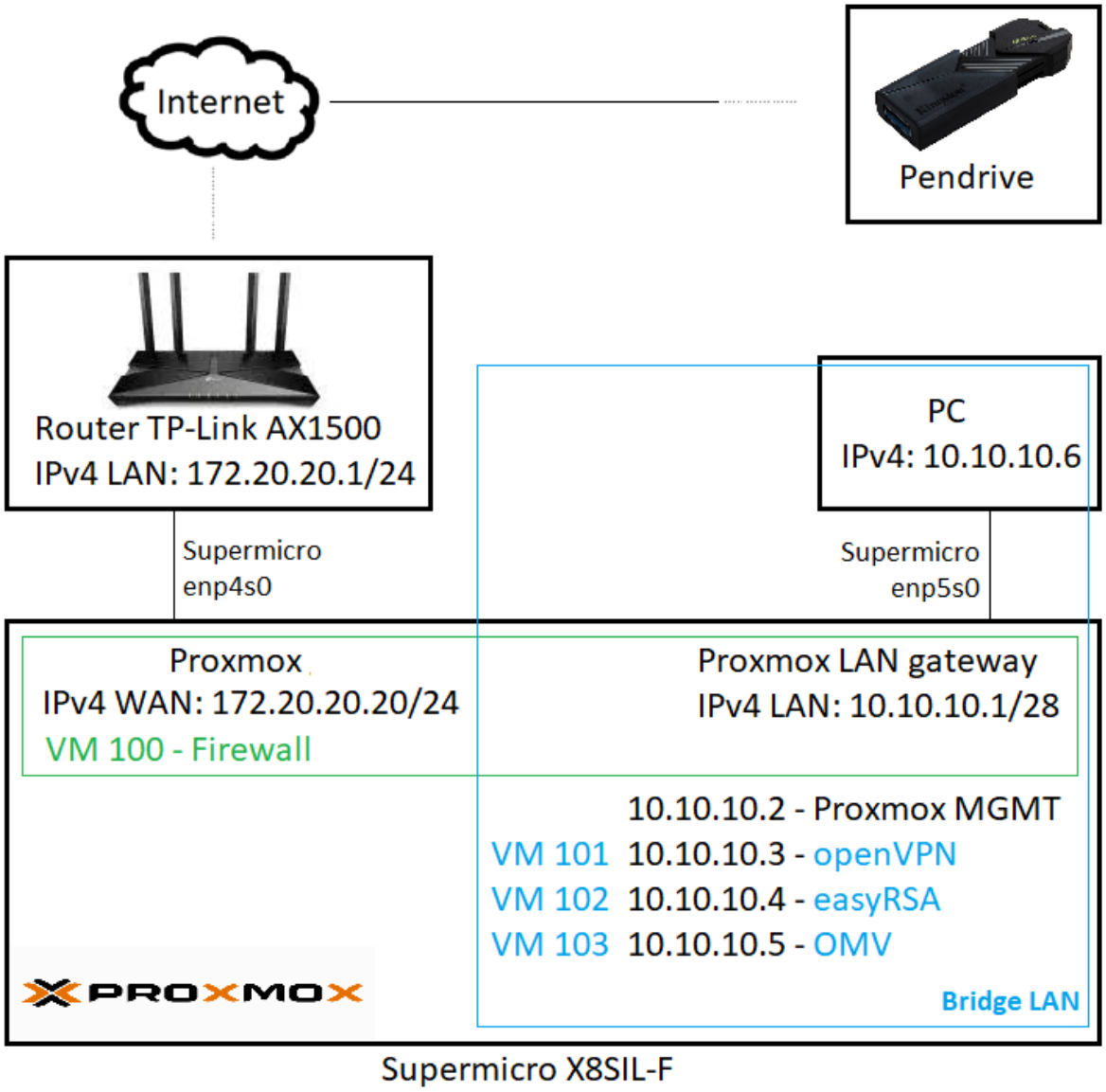
Rys. 10 Schemat najważniejszych komponentów sieci prywatnej

Do lepszego zobrazowania roli każdej z maszyn wirtualnych, rozpatrywane będą funkcjonalności w sposób pozwalający na złożenie informacji w logiczną całość, aniżeli opisując wedle kolejności numeracji.

Na fizycznym serwerze Supermicro uruchomione jest środowisko wirtualizacyjne Proxmox, co wskazano na Rys. 10. Hostuje ono 4 maszyny wirtualne, opisane kolejno numerami od 100 do 103 włącznie. Każda ze wspomnianych maszyn pełni określoną funkcję, zaś działanie wszystkich umożliwia użytkownikowi przyłączenie się do wirtualnej sieci prywatnej za pomocą nośnika pendrive i wymianę plików z serwerem SFTP. Należy zaznaczyć, że

urządzenie USB nie posiada wbudowanej karty sieciowej, wykorzystując do komunikacji sprzęt komputera, do której jest podłączone.

Serwer OpenVPN, uruchomiony na VM 101, wraz z całą infrastrukturą PKI z VM 102, zapewniają autoryzację, uwierzytelnienie i szyfrowanie danych przesyłanych przez klientów. Wszelkie reguły dotyczące filtrowania i trasowania pakietów, wraz ze śledzeniem stanu połączeń, realizowane są przez VM 100, czyli bramę domyślną sieci LAN maszyny Proxmox.



Rys. 11 Adresacja IP oraz połączenia logiczne między urządzeniami w sieci prywatnej

Serwer Supermicro posiada dwa interfejsy fizyczne, gdzie jeden jest na stałe podłączony do routera TP-Link (enp4s0), zaś do drugiego (enp5s0) można podłączyć kolejne sprzęty w sieci

lokalnej poprzez kabel Ethernet, np. komputer stacjonarny z ustawionym adresem IPv4 10.10.10.6. Dzięki temu, będąc we wspólnej domenie rozgłoszeniowej co środowisko Proxmox, użytkownik może uzyskać dostęp do GUI, czyli panelu zarządzania, po protokole HTTP, widocznego na Rys. 12. Adres IPv4 do wspomnianego nadzoru jest opisany jako „Proxmox MGMT" na Rys. 11. Nie jest zatem wymagane wyłączne zarządzanie z poziomu CLI, poprzez podłączenie się urządzeniami peryferyjnymi wyposażonymi w interfejsy szeregowe, ale możliwe. Środowisko Proxmox czy router TP-Link nie posiadają publicznego adresu IP na swoich interfejsach - ta rola realizowana jest na konwerterze światłowodowym od ISP. Tym samym, aby móc zestawić tunel z wykorzystaniem protokołu OpenVPN, wykonane jest przekierowanie portu usługi na całej trasie.

Podsumowanie najważniejszych komponentów infrastruktury:

- Fizyczny serwer Supermicro z zainstalowanym środowiskiem do wirtualizacji
- VM stanowiąca root CA dla oprogramowania OpenVPN - nr 102, easyRSA
- VM z uruchomionym serwerem OpenVPN - nr 101
- VM będąca zaporą sieciową urządzenia - nr 100, Firewall
- VM hostująca serwer SFTP - nr 103, OMV
- Pendrive jako klient serwera SFTP

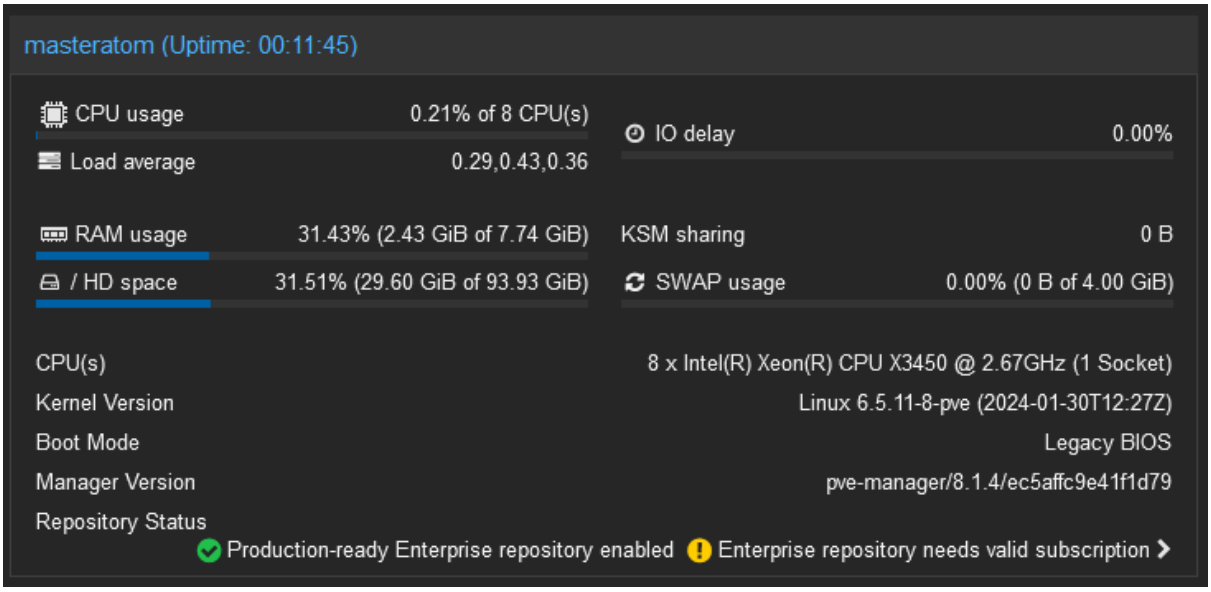
3.2. Opis konfiguracji poszczególnych elementów infrastruktury

3.2.1. Fizyczny serwer Supermicro ze środowiskiem Proxmox

Tab. 1 Wykaz parametrów serwera Supermicro

Płyta główna	X8SIL-F
Pamięć RAM	8 GB DDR3, Dual 1333 MHz
Procesor	Intel Xeon X3450, 2.67 GHz
Wirtualizacja	VT-d, I/O MMU
Karta sieciowa	2 x Intel Intel 82574L, 1 Gbps Ethernet

Środowisko Proxmox służy niemalże wyłącznie do hostowania maszyn wirtualnych, a także zapewnienia logicznego połączenia między nimi. Umożliwia ciągłe monitorowanie stanu pracy poszczególnych maszyn, jak i całego węzła (ang. *node*).



Rys. 12 Fragment widoku GUI węzła

Połączenie realizowane jest za pomocą mostu sieciowego, działającego w warstwie 2 modelu ISO/OSI. Jest on utworzony dla podsieci LAN, gdzie adres IPv4, do zarządzania środowiskiem z GUI, przydzielony jest następująco:

```
auto vmbr1
iface vmbr1 inet static
        address 10.10.10.2/28
        bridge-ports enp5s0
        bridge-stp off
        bridge-fd 0
```

Listing 1 Fragment ustawień interfejsu bridge z pliku /etc/network/interfaces

Drugi z mostów, vmbr0, łączy interfejs fizyczny enp4s0 z interfejsem wirtualnym maszyny wirtualnej obsługującej zaporę sieciową. Zapewnia on możliwość wydostania się pakietów z podsieci LAN do sieci WAN środowiska wirtualnego. Mosty te zostały dokładniej opisane na Rys. 11.Dostęp do platformy Proxmox realizowany jest przez skonfigurowanie użytkownika systemu Linux, więc dane logowania muszą być zapisane bezpośrednio na sprzęcie.

Musi być także spełnione założenie projektowe o separacji VM nr 102, czyli root CA. Domyślnie żadne z urządzeń w podsieci LAN, nie licząc dostępu ze środowiska wirtualizacyjnego, nie powinno mieć dostępu do głównego klucza prywatnego, podpisującego certyfikaty serwera i klientów serwera OpenVPN. Taka blokada jest zapewniona z poziomu hipernadzorcy, w stosunku do maszyn wirtualnych, poprzez:

- podejście trap and emulate dla środowisk x86, gdzie instrukcje wrażliwe wykonywane są bezpośrednio przez VMM, aniżeli udzielając dostęp hostom do zasobów bezpośrednio;
- stosowanie rozszerzonych tablic stron, umożliwiających dostęp do pamięci z pominięciem hipernadzorcy;
- korzystanie z mechanizmu deduplikacji, np. w odniesieniu do współdzielonego jądra OS Debian, używanego przez większość maszyn wirtualnych. Wspomniana także w części teoretycznej, technika overcommitment nie będzie miała zastosowania w tym projekcie, gdyż sumaryczny przydział pamięci RAM maszynom wirtualnym, nie przekracza wartości fizycznych kości RAM.

Jednakże z punktu widzenia sieci Ethernet, skoro bridge jest uruchomiony bezpośrednio na środowisku Proxmox, na nim musi ruch niechciany być odrzuconym w ramach działania warstwy drugiej. Realizowane jest to poprzez program ebtables, sterowany głównie skryptem, co zostało także nadmienione w Rozdziale 2.6. Skrypt ten jest w pełni dostępny w Dodatku A z załącznika 1. Umożliwia uruchomienie jednej z trzech opcji - drop, allow bądź status. Implementacja funkcji drop została przedstawiona w Listingu 2.

```
do_drop()
{
        ebtables -F
        # <komentarze zostały pominięte>
        ebtables -t filter -A FORWARD -p IPv4 -s XX:XX:XX:XX:XX:XX -j DROP
        ebtables -t filter -A FORWARD -p IPv4 -d XX:XX:XX:XX:XX:XX -j DROP
        ebtables -t filter -A FORWARD -p IPv4 --ip-src 10.10.10.4 -j DROP
        ebtables -t filter -A FORWARD -p IPv4 --ip-dst 10.10.10.4 -j DROP
}
```

Listing 2 Fragment działania skryptu, odpowiedzialny za odrzucanie ruchu do VM easyRSA

Adres MAC maszyny wirtualnej easyRSA został intencjonalnie zanonimizowany. Domyślnie negowany jest ruch zarówno po adresie MAC wirtualnego interfejsu sieciowego (na

łańcuchu forward do maszyny easyRSA), jak i jego adresie statycznym IPv4 (to już w ramach L3, czyli maszyny Firewall).

Funkcja drop jest uruchamiana przez program cron w dwóch wypadkach - przy włączeniu maszyny fizycznej, oraz w ustalonych, czasowych interwałach. Ta druga to ochrona przed ludzkim błędem, umożliwiająca maksymalnie 5-minutowe okno dostępowe. Jeśli użytkownik zapomniałby ręcznie zablokować komunikację z root CA, wykona to za niego program cron.

```
root@masteratom:~# crontab -l
#Reload drop on VM easyRSA starting XX:00 and every 5 minutes so on
@reboot /etc/ebtables/ebtables.process drop
*/5 * * * * /etc/ebtables/ebtables.process drop
```

Listing 3 Uruchomione zadania programu cron środowiska Proxmox

W przypadku chęci odblokowania dostępu, aby serwer OpenVPN mógł się połączyć z urzędem certyfikacji w celu podpisania CSR, możliwe jest zniesienie ograniczeń między urządzeniami, poprzez użycie funkcji allow.

Ruch na jakimkolwiek z łańcuchów jest blokowany w programie iptables na maszynie Firewall w kierunku VM easyRSA, ale dzięki wyszczególnieniu w warstwie drugiej adresacji IP, możliwe jest nawiązanie połączenia pomiędzy serwerem OpenVPN a root CA.

3.2.2. Urząd certyfikacji infrastruktury klucza publicznego - VM easyRSA

Tab. 2 Wykaz parametrów VM 102

Pamięć RAM	512 MB
Wirtualizowany procesor [typ]	kvm64
Liczba gniazd/rdzeni	1/1
Port usługi SSH	22

Infrastruktura PKI jest zbudowana dzięki programowi easy-rsa, który korzysta z bibliotek openSSL, a jego katalog dostępny jest w przestrzeni /etc/easy-rsa. Na maszynie wirtualnej nr 102 ujęte są odpowiednie certyfikaty i elementy infrastruktury PKI:

- klucz prywatny root CA, służący do podpisywania CSR klientów oraz serwerów. Opisany jest jako ca.key;
- klucz publiczny root CA, stosowany do potwierdzania tożsamości głównego urzędu certyfikacji przy wykorzystywaniu przez hosty, np. serwera OpenVPN. Oznaczony jest jako ca.crt;
- CSR oraz podpisane ich certyfikaty przez root CA. Takie zaświadczenia mają już uzupełnione parametry wedle standardu X.509, jak nazwę zatwierdzającego urzędu certyfikacji, datę ważności, własności wykorzystania klucza (różne dla serwera i klienta) itd.,
- numery seryjne wydanych certyfikatów.

Aby możliwe było działanie jednostek bazujących na zbudowanej infrastrukturze, należy wysłać CSR w bezpieczny sposób na maszynę easyRSA z serwera OpenVPN. Zrealizowane to zostanie poprzez komendę ‘scp’, czyli *secure copy*. Po przesłaniu pliku, wykorzystując skrypt Sign_req.sh, sprawdzane jest czy nie istnieje już certyfikat o jednakowej nazwie, a w przypadku niepowodzenia, zostanie on utworzony. Funkcja ta jest ujęta w Dodatku B w załączniku 1.

Skrypt, jak większość innych (na pozostałych VM), używanych w powłoce bash w ramach projektu, posiada kontrolę liczby wprowadzanych parametrów. Ma to uchronić plik wynikowy przed utworzeniem z nieprawidłowymi wartościami, zdejmując ten obowiązek z użytkownika. Realizowane jest to przez funkcję warunkową „if [$# -ne 1]”. W pozostałej części programu, ujętego w Dodatku B w załączniku 1 następuje sprawdzenie, czy plik nie został już utworzony - tę rolę pełni kombinacja poleceń find oraz grep, która po sprawdzeniu warunku zwraca wartość logiczną do programu.

Plik nie jest natychmiastowo odsyłany do serwera, gdyż bezpieczniej jest podjąć drugą, przemyślaną decyzję przez administratora, aniżeli dopiero po zrozumieniu błędu wykonać przymusowe unieważnienie wydanego certyfikatu. Ten następny krok realizuje skrypt Send_cert.sh, który przesyła plik do określonego miejsca na serwerze OpenVPN. Został on umieszczony w projekcie jako Dodatek C w załączniku 1. Wykorzystywany jest wspomniany

program scp. Po wymaganych czynnościach, jeśli użytkownik zapomniałby zablokować specjalnie przyznawany dostęp, to wciąż do 5 minut zostanie wyłączona możliwość komunikacji z maszyną wirtualną easyRSA, dzięki programowi cron w node masteratom.

Dlaczego Root CA pełni funkcję jedynego urzędu certyfikacji, zarówno dla serwera, jak i klientów? Można by stworzyć sub-ca (czyli pośredniczący w komunikacji z głównym) na wspólnej maszynie co serwer OpenVPN, który przejąłby rolę podmiotu do podpisywania otrzymywanych żądań. Certyfikat wydawany dla sub-ca ma domyślne uprawnienia standardu X.509 "Key Usage: Certificate Sign, CRL Sign", natomiast serwer X.509 „Digital Signature, Key Encipherment". Oznacza to, że obiekt z parametrami sub-ca, może co najwyżej podpisać certyfikat na tej samej maszynie. Oprócz tego, taki serwer nie będzie mógł potwierdzać tożsamości klientów przy próbie nawiązania połączenia do niego, co wymuszałoby utworzenie kolejnej, nadmiarowej instancji. Dodatkowo, taka jednostka będzie słabiej odseparowana od pozostałych urządzeń w sieci, niż aktualne rozwiązania to zapewniają (np. wirtualizacja).

3.2.3. Serwer wirtualnej sieci prywatnej - VM openVPN

Tab. 3 Wykaz parametrów VM 101

Pamięć RAM	512 MB
Wirtualizowany procesor [typ]	kvm64
Liczba gniazd/rdzeni	2/1
Port usługi SSH	8823
Port usługi serwera OVPN	8312

Nasłuchiwanie serwera VPN uruchomione jest na porcie 7312. Ruch jest przesyłany z wykorzystaniem protokołu UDP, gdyż zapewnia większą ochronę przeciwko skanowaniu

portów oraz przed atakami DoS, niż TCP. Wirtualnym interfejsem sieciowym jest urządzenie tun, tworzące routowany tunel IP.

Serwer OpenVPN oprócz obsługiwania klientów, czy to nowych (żądających utworzenia CSR), czy obecnych (podłączających się do sieci lub przy przerzucaniu ruchu), musi porozumiewać się z root CA. Komunikacja ta jest umożliwiona przez zestaw kolejnych skryptów, które automatycznie sprawdzają i wykonują czynności za użytkownika (administratora).

W celu zwiększenia prostoty, dostępny jest poradnik postępowania przy tworzeniu nowego klienta, wyświetlany komendą „OVPN-help". Podzielony jest na część bardziej szczegółową (opisową), jak i wyłącznie do uruchomienia z poziomu CLI, gdzie wspomniane programy są jedynie wypisane skrótowo. Dostępny jest jako Dodatek D w załączniku 1.

Chcąc wygenerować żądanie podpisania certyfikatu, należy skorzystać ze skryptu Gen_req.sh. Wysyła on plik na root CA, jeśli tylko komunikacja jest dozwolona. Skrypt opisany jest w zakładce Dodatek E w załączniku 1. Katalog, do którego dane są przekazane na VM 102, jest ściśle określony, gdyż wspomniane w poprzednim punkcie programy tamtej maszyny wirtualnej także z niego korzystają. Przez to, proces tworzenia nowych klientów przebiega sprawnie, będąc w niewielkim stopniu niezautomatyzowanym.

Po otrzymaniu podpisanego certyfikatu, pomijając kwestie kontroli wersji, zostaje utworzony plik konfiguracyjny .ovpn, zawierający wszelkie wymagane składowe. Są nimi: certyfikaty root CA oraz zainteresowanego, klucz prywatny klienta oraz TLS, co zostało wskazane na Rys. 13. Funkcję tę realizuje skrypt Create_client_config.sh, dostępny w Dodatku F w załączniku 1.

```
cat ${DEFAULT_CLIENT_CONFIG} \
    <(echo '<ca>') \
    ${KEYS}/ca.crt \
    <(echo -e '</ca>\n<cert>') \
    ${KEYS}/${1}.crt \
    <(echo -e '</cert>\n<key>') \
    ${PRIVATE_KEYS}/${1}.key \
    <(echo -e '</key>\n<tls-auth>') \
    ${KEYS}/ta.key \
    <(echo '</tls-auth>') \
    > ${CLIENT_OVPN_FILE}/${1}.ovpn
```

Rys. 13 Fragment skryptu Create_client_config.sh

Uruchomiony serwer OpenVPN pozwala autoryzować klientów, a także oferuje kilka funkcjonalności:

- push "redirect-gateway def1 bypass-dhcp"

Opcja ta pozwala przekierować cały ruch podłączonych hostów przez bramę domyślną serwera OpenVPN. Dzięki temu, klienci są widoczni w Internecie jakby posiadali publiczny adres IPv4 sieci serwera OpenVPN, aniżeli np. swojego dostawcy. Wspólnie z tą opcją, należy na maszynie wirtualnej uruchomić możliwość przekazywania ruchu w ramach łańcucha forward, wpisując logiczną wartość „prawda” do parametru ustawień jądra: echo 1 > /proc/sys/net/ipv4/ip_forward

Domyślna trasa wychodząca maszyny openVPN (default via 10.10.10.1 dev ens18 onlink) wskazuje na adres IP interfejsu maszyny Firewall, i to właśnie ona zostanie „wstrzyknięta” klientom serwera OpenVPN. Gdyby ta opcja nie była włączona, to klient uzyskałby dostęp do wszelkich zasobów serwera, jednak nie mógłby przedstawiać się innym adresem publicznym w sieci Internet.

- ifconfig-pool-persist /var/log/openvpn/ipp.txt

Każdy z klientów posiada przydzielony mu dynamicznie adres IPv4 z serwerowej puli DCHP (jednak na serwerze występuje rezerwacja statyczna adresu dla klienta). W przypadku rozłączenia serwera, jesteśmy w stanie odnowić część niezamkniętych połączeń przy ponownym uruchomieniu procesu. Taka informacja umożliwia także logowanie stanu połączeń

podłączonych klientów, co obrazuje Rys. 14. Klient „Kacper”, połączony z adresu IP 10.10.10.6 (czyli podsieci LAN), zestawił połączenie na dynamicznym porcie 59708. Został mu tym samym przydzielony wirtualny adres IP 10.8.0.6, którego bramą jest adres IP 10.8.0.1, przypisany interfejsowi tun0 na maszynie wirtualnej openVPN.

```
Common Name,Real Address,Bytes Received,Bytes Sent,Connected Since
Kacper,10.10.10.6:59708,3292,5576,2024-08-29 20:00:24
ROUTING TABLE
Virtual Address,Common Name,Real Address,Last Ref
10.8.0.6,Kacper,10.10.10.6:59708,2024-08-29 20:00:24
```

Rys. 14 Fragment pliku logującego podłączonych użytkowników serwera OpenVPN

- tls-auth client/keys/ta.key 0

Przy próbie połączenia, następuje zestawienie tunelu w oparciu o kryptografię asymetryczną, aby po nawiązaniu bezpiecznego powiązania, przejść na rozwiązanie kluczy symetrycznych HMAC (ang. (ang. *keyed-Hash Message Authentication Code*). Powodem jest dużo większa szybkość przesyłu danych, w przeciwieństwie do wykonywania dużo bardziej obciążających obliczeń technik asymetrycznych, np. RSA. Klucz ten będzie używany przez klienta, gdyż jest umieszczony w generowanym pliku konfiguracyjnym .ovpn.

3.2.4. Zapora sieciowa kontrolująca przesył pakietów - VM Firewall

Tab.2. Wykaz parametrów VM 100

Pamięć RAM	512 MB
Wirtualizowany procesor [typ]	kvm64
Liczba gniazd/rdzeni	2/1
Port usługi SSH	8822

Maszyna Firewall jest najważniejszym elementem sieciowym przedstawionego rozwiązania, biorąc pod uwagę osiągalność klientów i serwera, jak i ich bezpieczeństwo. Reguły przedstawione w tym punkcie, są na maszynie wirtualnej wczytywane ze specjalnego pliku /etc/iptables/iptables.rules przy każdej zmianie statusu interfejsu. Czynność jest wykonywana przez krótki skrypt w katalogu /etc/network/if-pre-up.d:

```
#!/bin/sh
/sbin/iptables-restore < /etc/iptables/iptables.rules
```

Listing 4 Skrypt ładujący konfigurację programu iptables

Konfiguracja maszyny zapewnia kontrolę pakietów warstwy 3, domyślnie odrzucając ruch niezależnie od łańcucha tabeli filter, co zostało wskazane w Listingu 5. Rozwiązanie w iptables jest fuzją zapory sieciowej typu stateful oraz stateless.

```
-F
-P OUTPUT DROP
-P INPUT DROP
-P FORWARD DROP
```

Listing 5 Domyślne reguły iptables całkowicie blokujące ruch

Kolejne powtórzenia reguł odrzucających są zamierzone, gdyż obejmują specyficzne przypadki, które nie warto lekceważyć. Dotyczą:

- ruchu na interfejsie loopback, gdyż używany jest przez wiele usług systemu operacyjnego gościa. Loopback nie może zostać wyłączony w prosty sposób, np. poprzez odłączenie kabla, jak w przypadku interfejsów fizycznych;
- transmisji na portach 137-139, czyli usługa Netbios, gdzie maszyny wirtualne Windows mogłyby ogłaszać się światu;
- komunikacji w L3 z VM easyRSA.

Następne z reguł będą odwoływać się do szczegółowych rozwiązań:

- -A INPUT -m state --state ESTABLISHED -j ACCEPT

Wpis umożliwia instalowanie aktualizacji na maszynie Firewall. State ESTABLISHED dotyczy połączeń, które zanotowały ruch z obu stron, np. wskutek wychodzących pakietów z maszyny wirtualnej w stronę Internetu.

- -A FORWARD -m state --state ESTABLISHED -j ACCEPT

Odrzucenie stanu RELATED dla łańcucha FORWARD wynika z większego bezpieczeństwa na „tylko" akceptowanie połączeń nawiązanych z obu stron, aniżeli skojarzonych z już istniejącym. Stan RELATED może czasem zezwolić na ruch w obrębie usług, które zostały jawnie odrzucone, np. odpowiedzi ICMP.

- -A INPUT -s 10.10.10.0/28 -p tcp --dport 8822 -j ACCEPT

Port SSH do maszyny został zmieniony ze standardowego 22. Dodatkowo, ograniczona jest możliwość nawiązania połączenia do sieci lokalnej. Port SSH w teorii mógłby być otwarty do Internetu dzięki rozwiązaniu Fail2ban. Lepszym podejściem będzie jednak umożliwienie zestawienia sesji dla w pełni zatwierdzonych klientów, łączących się do serwera OpenVPN, niż wyłapywanie dziesiątek prób połączeń do maszyny przez krążące po Internecie automaty.

- -A INPUT -p udp -d 10.10.10.3 --dport 7312 -j ACCEPT
  -A FORWARD -p udp -d 10.10.10.3 --dport 7312 -j ACCEPT

Reguły zezwalają na połączenie na wystawionym porcie usługi serwera OpenVPN. Wpis ten jest komplementarny z przedstawionym poniżej z tablicy nat do prawidłowego zestawiania tuneli.

Tablica NAT:

- -A PREROUTING -p udp --dport 7312 -j DNAT --to-destination 10.10.10.3:7312

Reguła umożliwia przekierowanie portu ruchu przychodzącego ze świata, czyli skierowanego w NAT (stąd DNAT - destination NAT). Jak zostało wspomniane w Rozdziale 2.5, w łańcuchu PREROUTING jest dokonywana początkowa analiza otrzymanego pakietu, jeszcze przed przeprowadzeniem pierwszych decyzji trasowania.

- -A POSTROUTING -s 10.10.10.0/28 -m iprange ! --src-range 10.10.10.4 -j MASQUERADE

Aby maszyny wirtualne serwera fizycznego, jak i klienci podłączeni do serwera OpenVPN, mogły osiągać inne hosty w ramach działania Internetu, należy zmodyfikować pakiet na każdym z routerów. Według planu, TP-Link w podsieci WAN, czyli 172.20.20.0/24, nie zna trasy do podsieci 10.10.10.0/28, więc teoretycznie otrzymując pakiet od 10.10.10.3, odrzuci go. Maskarada w łańcuchu POSTROUTING pozwala na zrealizowanie tego warunku poprzez podmienienie źródłowego adresu IP pakietu na znany routerowi, czyli 172.20.20.20 (adres IPv4 na interfejsie WAN maszyny Firewall) z 10.10.10.1, czyli adresu lokalnej bramy. Taka relacja zostaje zapisana w tablicy połączeń urządzenia (L4), o określonym czasie życia pakietu, więc w przypadku otrzymania danych z Internetu, router wie na który z interfejsów ma przekierować datagram.

Dodatkowo, usługa SSH maszyny Firewall chroniona jest przez zewnętrzne oprogramowanie Fail2ban. W przypadku próby ataku metodą brute force, zdefiniowanego jako 5 nieudanych logowań w czasie 5 minut, adres IP napastnika zostaje umieszczony w specjalnym łańcuchu zapory sieciowej na 2 godziny, co przedstawione na Rys. 15.

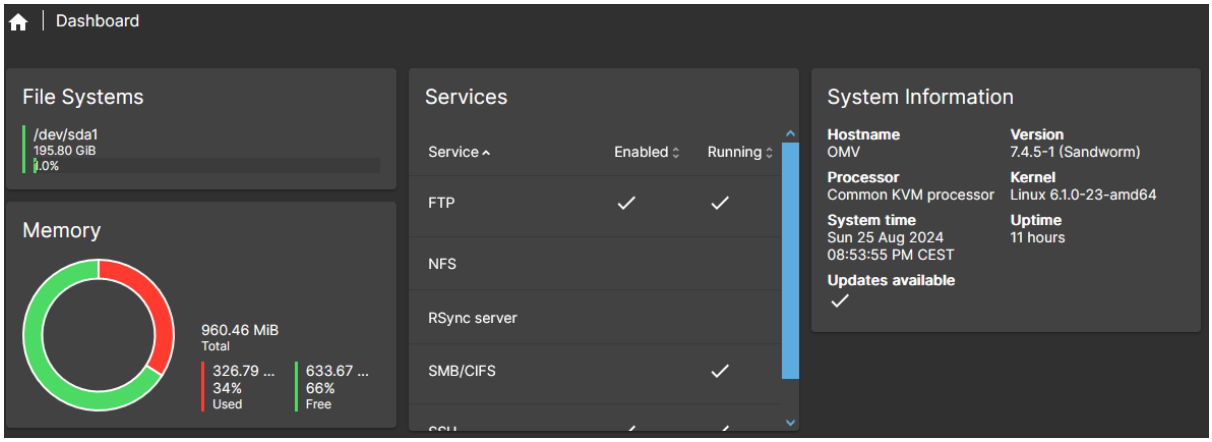
Rys. 15 Specjalny łańcuch programu iptables, widoczny po zablokowaniu atakującego

3.2.5. Serwer usługi SFTP - VM OMV (OpenMediaVault)

Tab. 4 Wykaz parametrów VM 103

Pamięć RAM	1 GB
Wirtualizowany procesor [typ]	kvm64
Liczba gniazd/rdzeni	1/2
Port usługi SSH	8825
Port usługi serwera SFTP	9925

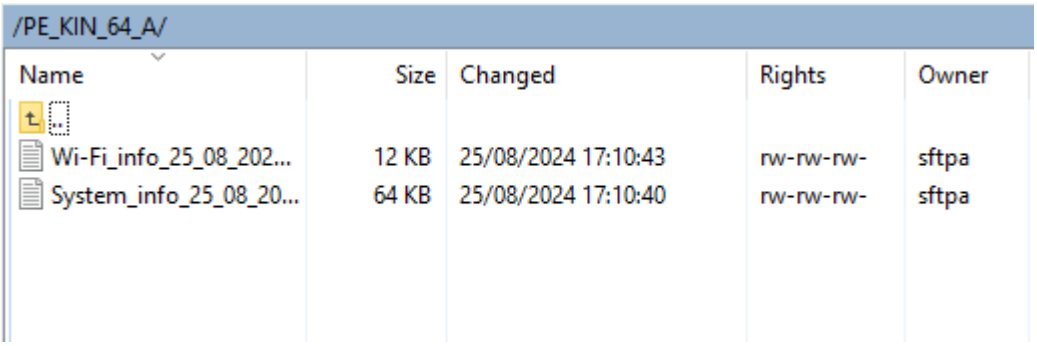
Ostatnia z maszyn wirtualnych o nr 103, czyli serwer SFTP. Nie licząc prostych ustawień konfiguracyjnych, które można w pełni wykonać z poziomu CLI, OpenMediaVault jest w pełni zarządzalny przez GUI. Na Rys. 16 wskazano Dashboard serwera.



Rys. 16 Prosty dashboard w GUI

Serwer SFTP, o przydzielonej pamięci masowej 200 GB (ide1), obsługuje klientów na niestandardowym porcie - 9925. W wersji bieżącej uruchomione jest wyłącznie jedno konto użytkownika z uprawnieniami pozwalającymi na korzystanie z zasobów dyskowych. Klient loguje się poprzez podanie nazwy użytkownika oraz hasła dostępowego. Dzięki protokołowi SSH, bezpieczne jest połączenie np. w ramach sieci lokalnej niż w przypadku korzystania z protokołu FTP.

Do serwera można zestawić połączenie z klienta FTP, np. FileZilla lub WinSCP. Serwer dostępny jest w sieci lokalnej pod adresem 10.10.10.5/28. Katalog użytkownika to PE_KIN_64_A (oznaczający pendrive Kingston o pojemności 64 GB i identyfikatorze własnym A), odnoszący się do proponowanego sposobu połączenia poprzez wspomniany nośnik, co wskazano na Rys. 17.



Rys. 17 Zasoby dyskowe serwera SFTP widoczne z aplikacji FileZilla

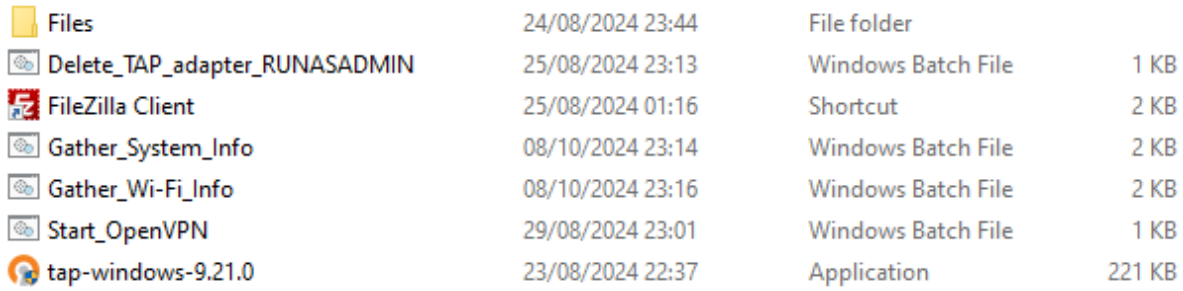
3.2.6. Połączenie się z serwerem SFTP

Do łatwego i szybkiego podłączenia się do serwera SFTP z Internetu został przygotowany nośnik zewnętrzny. Pendrive ten posiada wyłącznie niezbędne pliki konfiguracyjne, jak i skrypty do wykorzystania na OS Windows, co zostało przedstawione na Rys. 18. Programy znajdujące się na nim to:

- OpenVPN, zawiera niezbędne pliki z biblioteki DLL oraz wyekstrahowaną z serwera OpenVPN konfiguracja klienta;
- FileZilla Portable, czyli prosty klient FTP.

Na nośniku danych umieszczone zostały także skrypty powłoki Windows:

- Start_OpenVPN.bat, czyli aplikacja zestawiająca tunel do serwera wirtualnej sieci prywatnej. Do jej działania niezbędne jest zainstalowanie sterownika TAP-windows, umieszczonego na urządzeniu pendrive;
- Gather_System_Info.bat, prosty program zapisujący do pliku przeróżne ustawienia systemowe bieżącego urządzenia;
- Gather_Wi-Fi_Info.bat, aplikacja wyciągająca dane znanych komputerowi połączeń z sieciami bezprzewodowymi;
- Delete_TAP_adapter_RUNASADMIN.bat, skrypt usuwający wcześniej zainstalowany sterownik TAP-Windows.



Rys. 18 Narzędzia umieszczone na pendrive

Aby połączyć się do VPN, należy wyposażyć komputer w interfejs sieciowy TAP-Windows. Nie można zakładać jego istnienia na systemie Windows, stąd posiadając program instalacyjny, możliwe jest jego natychmiastowe użycie. Po nawiązaniu połączenia z siecią serwera OpenVPN, oprócz przesyłania plików, możliwe jest zebranie odpowiednich danych diagnostycznych. Wszystkie skrypty wykorzystują literę dysku urządzenia, z którego zostały uruchomione, poprzez zastosowanie zmiennej %~dp0.

Poszczególne skrypty:

- Start_OpenVPN.bat, przedstawiony w Listing 6.

```
cd /d %~dp0Files\OpenVPN\bin
openvpn.exe --config ..\config\PE_KIN_64_A\PE_KIN_64_A.ovpn
```

Listing 6 Skrypt odpowiadający za zestawienie połączenia do sieci VPN

- Gather_System_Info.bat

Skrypt został przedstawiony w całości jako Dodatek G w załączniku 1. Plik wyjściowy zawiera w tytule datę oraz godzinę przeprowadzenia operacji. Format zmiennej date oraz time można sprawdzić uruchamiając je z poziomu CMD systemu operacyjnego Windows.

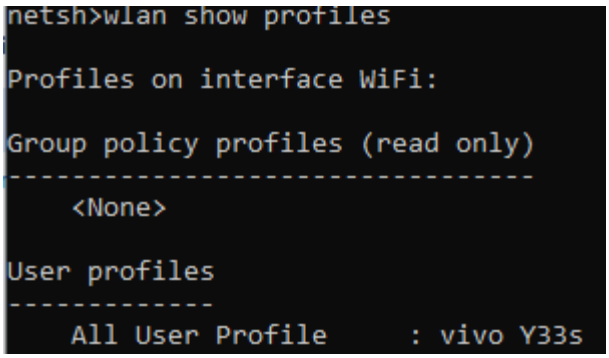
Skrypt zapisuje do pliku najbardziej podstawowe dane diagnostyczne, jak systeminfo (nazwa komputera, dane pamięci RAM, wersja oprogramowania), odczytuje także aktualnie włączone procesy (tasklist) czy zestawione połączenia sieciowe (netstat -an), co widać na Rys. 19.

```
echo === Get processor info: === >> %output_file%
wmic cpu get name, NumberOfCores, NumberOfLogicalProcessors, MaxClockSpeed >> %output_file%

echo === Get processor load: === >> %output_file%
wmic cpu get loadpercentage >> %output_file%

echo === Get BIOS info: === >> %output_file%
wmic bios get manufacturer, smbiosbiosversion, version >> %output_file%
echo. >> %output_file%

echo === Get drivers: === >> %output_file%
driverquery >> %output_file%
echo. >> %output_file%
```

Rys. 19 Fragment skryptu Gather_System_Info

- Gather_Wi-Fi_Info.bat

Skrypt jest ujęty jako Dodatek H w załączniku 1. Wykorzystana została tutaj prosta komenda netsh wlan show profiles, a poprzez odpowiedni podział pliku tekstowego, możliwe jest odczytanie informacji o każdej z kiedykolwiek połączonych sieci Wi-Fi. W skrypcie użyto zmiennych opóźnionych, bez których bezbłędne wykonanie pętli i odczytanie danych jest niemożliwe.



Rys. 20 Format tekstowy wywołania programu netsh wlan show profiles

Każdy z odczytanych profili jest poprzedzony dwukropkiem oraz spacją (czyli jednym znakiem), co widać na Rys. 20. Można podzielić taki ciąg znaków na dwie, odseparowane wartości. Uzyskano wtedy pierwszy człon, czyli „All User Profile” oraz drugi - „ vivo Y33s”. Wystarczy teraz zmienną opóźnioną przekształcić, usuwając żądaną liczbę znaków drugiego ciągu - „:~1’’ oznacza pominięcie pierwszego symbolu, czyli znaku spacji. Wynikiem jest

prawidłowa nazwa profilu sieci bezprzewodowych, której użyto jako argument do odczytania wszelkich danych diagnostycznych nt. nawiązanego połączenia. Plik wynikowy także zapisywany jest do odpowiedniego miejsca na nośniku - <litera_dysku>:\Files\Batch_results.

- Delete_TAP_adapter_RUNASADMIN.bat

Do uruchomienia skryptu niezbędne jest przydzielenie uprawnienia administratora. W innym wypadku, nie jest możliwe usunięcie sterownika.

Program, szczegółowo przedstawiony w Dodatku I w załączniku 1, opiera się o działanie komendy pnputil /enum-devices, która zawiera informacje o aktualnie zainstalowanych sterownikach w systemie. Wykorzystując wzorzec dotyczący adaptera, wyekstrahowano jego prawidłową nazwę. Posiadając jego prawidłowy format, możliwe jest usunięcie go z listy sterowników, wykorzystując komendę: pnputil /delete-driver %TAP_interface% /uninstall /force. Numer sterownika może różnić się w zależności od sprzętu, stąd skrypt opiera się na jedynej, niezmiennej wartości.

3.3. Model serwerowej przestrzeni dyskowej

W celu weryfikacji przedstawionej koncepcji budowy infrastruktury serwerowej, został opracowany model do przeprowadzenia badań testowych. Składa się na niego:
- telefon komórkowy,
- komputer stacjonarny z zainstalowanym systemem operacyjnym Windows 10,
- dostęp do Internetu,
- uruchomione skrypty usług, zapisujące dane na wybranych maszynach wirtualnych.

Telefon komórkowy posiada zainstalowane oprogramowanie klienta OpenVPN, a także możliwość przeglądania stron internetowych. Na komputerze stacjonarnym zalogowany jest użytkownik z uprawnieniami administratora, co umożliwia na dostęp do konsoli Windows PowerShell, a także innych, chronionych obiektów (np. sterowników oprogramowania). Dodatkowo, zainstalowane są programy jak klient serwera OpenVPN czy Wireshark. Oba urządzenia są w stanie połączyć się do Internetu zarówno poprzez operatora komórkowego (w technologii LTE), jak i stacjonarnego (poprzez wykorzystanie infrastruktury

światłowodowej). Część danych zostanie także pobrana bezpośrednio z maszyn wirtualnych, które zapisują informacje diagnostyczne o przetwarzanych usługach.

3.3.1. Analiza zestawiania tunelu w programach do obserwowania ruchu sieciowego

Przypadek będzie omawiany przy zestawianiu połączenia w sieci lokalnej, do adresu IP 10.10.10.3, czyli na interfejsie LAN maszyny wirtualnej openVPN.

Już od samego początku procedury, szyfrowany zostaje ruch przesyłany w ramach TLS handshake w ramach protokołu TLS, co przedstawiono na Rys. 21. Widoczne są wyłącznie pakiety przesyłane w ramach protokołu UDP.

```
root@openVPN:~# tcpdump udp port 7312
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on ens18, link-type EN10MB (Ethernet), snapshot length 262144 bytes
23:30:16.645692 IP 10.10.10.6.63759 > openVPN.7312: UDP, length 42
23:30:16.645961 IP openVPN.7312 > 10.10.10.6.63759: UDP, length 54
23:30:16.646851 IP 10.10.10.6.63759 > openVPN.7312: UDP, length 337
23:30:16.649886 IP openVPN.7312 > 10.10.10.6.63759: UDP, length 1222
```

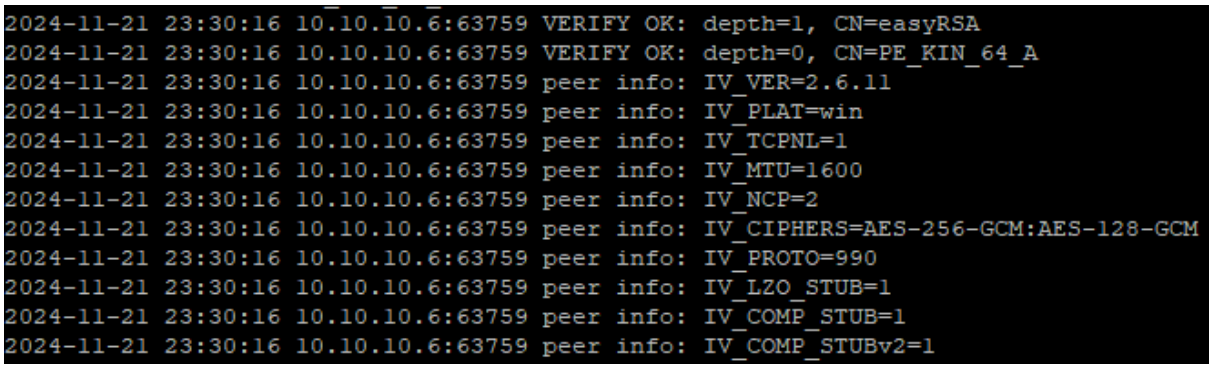
Rys. 21 Odczyt programu tcpdump podczas zestawiania tunelu

Na urządzeniu klienckim możliwe jest wyłącznie podejrzenie komunikacji z serwerem DHCP serwera OpenVPN (Rys. 22), a także wymiany pakietów w ramach dołączenia do grup multicastowych.

```
0.0.0.0         255.255.255.255    DHCP     342 DHCP Discover - Transaction ID 0x2a155261
10.8.0.9        255.255.255.255    DHCP     314 DHCP Offer    - Transaction ID 0x2a155261
0.0.0.0         255.255.255.255    DHCP     370 DHCP Request  - Transaction ID 0x2a155261
10.8.0.9        255.255.255.255    DHCP     314 DHCP ACK      - Transaction ID 0x2a155261
```

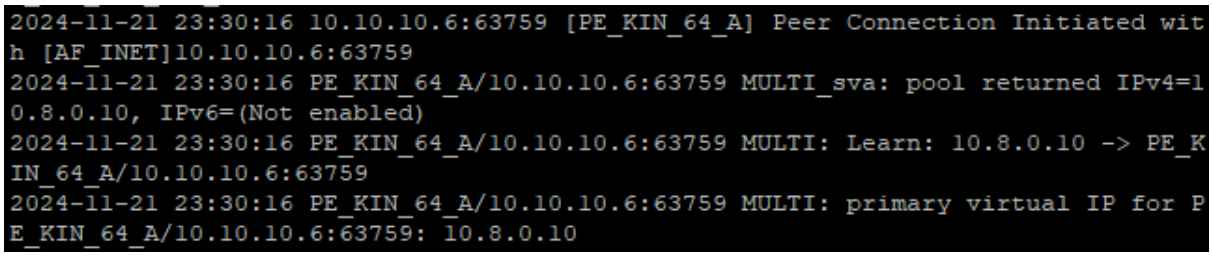
Rys. 22 Przechwycone pakiety przydzielania adresu IP od serwera DHCP do klienta w programie Wireshark

Jak widać, potencjalny atakujący nie jest w stanie przechwycić żadnych szczegółów transmisji. Jedynym miejscem, gdzie takie dane są gromadzone, jest plik logujący na serwerze OpenVPN. Na Rys. 23, przedstawiony został pierwszy etap wspomnianej procedury TLS handshake. Klient prezentuje serwerowi wspierane algorytmy kryptograficzne (szyfrowania, funkcji skrótu), wersję protokołu TLS czy parametry związane z kompresją danych.



Rys. 23 Fragment logów dotyczący połączenia klienta z serwerem OpenVPN

W dalszej części zanotowanych wpisów, na Rys. 24 można zauważyć komunikację hosta z serwer DHCP serwera OpenVPN, która została wspomiana na Rys. 24.



Rys. 24 Fragment logów opisujący wymianę ramek w ramach protokołu DHCP

Ostatecznie, najbardziej wartościowe informacje są ujęte na Rys. 25. Jest to zbiór wszelkich danych o uzgodnionych parametrach połączenia. Rozbijając treść na części składowe, ustalono używany protokół TLS w wersji 1.3. Następnie, z zestawu szyfrów (ang. *cipher suite*) TLS_AES_256_GCM_SHA384 wiadomo o:

- używanym algorytmie szyfrowania AES_256_GCM (symetryczny szyfr blokowy o długości klucza 256 bitów o trybie pracy Galois/Counter Mode),
- algorytmie skrótu SHA384, związanego z GCM.

Kolejne własności informują o stosowanym certyfikacie klucza publicznego o długości 2048 bitów, opartym o algorytm RSA. Finalnie, do weryfikacji autentyczności serwera służy algorytm podpisu cyfrowego RSA-SHA256.

```
2024-11-21 23:30:16 10.10.10.6:63759 Control Channel: TLSv1.3, cipher TLSv1.3 TL
S_AES_256_GCM_SHA384, peer certificate: 2048 bit RSA, signature: RSA-SHA256
```

Rys. 25 Uzgodnione parametry bezpiecznego połączenia w oparciu o protokół TLS

3.3.2. Zestawienie tunelu do serwera OpenVPN

Połączenie jest realizowane po publicznej sieci Internet z wykorzystaniem usług od dwóch różnych operatorów. Prezentacja działania została zaprezentowana na telefonie komórkowym z systemem Android. Adres IPv4 z Rys. 26 należy do operatora sieci GSM, co potwierdza informacja z bazy danych organizacji RIPE, widoczna na Rys. 27.

Na Rys. 28 widoczny jest adres IPv4 stacjonarnego dostawcy Internetu, którym identyfikuje się serwer VPN. Rys. 29, czyli wypis z bazy RIPE, wskazuje już na innego ISP. Przedstawiono statystyki z aktywnego połączenia do serwera VPN (Rys. 30). Potwierdzeniem prawidłowego zestawienia tunelu jest symbol klucza na pasku powiadomień urządzenia.

Rys. 26 Adres publiczny IPv4 przed podłączeniem się do sieci VPN

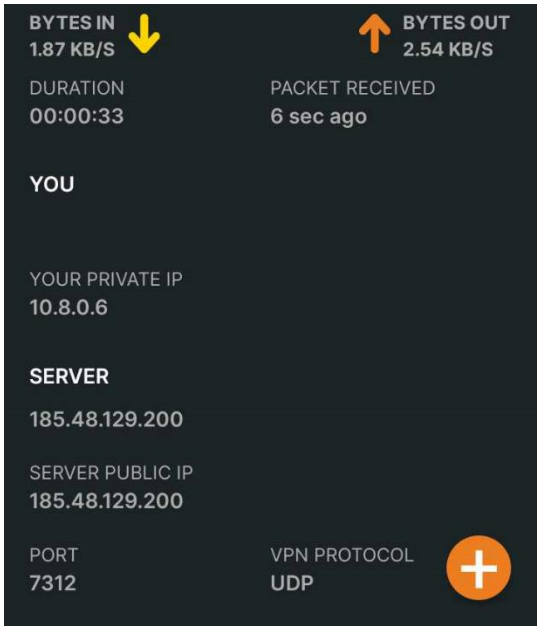
```
inetnum:        37.47.0.0 - 37.47.127.255
netname:        PL-IDEA-MOBILE
descr:          Orange Mobile
```

Rys. 27 Wyciąg z bazy organizacji RIPE dot. adresu IP należącego do Orange

Rys. 28 Adres IPv4 publiczny po przyłączeniu się do sieci VPN

```
inetnum:        185.48.128.0 - 185.48.131.255
netname:        PL-CYBERNETWMW4-20140220
```

Rys. 29 Wyciąg z bazy organizacji RIPE dot. adresu IP należącego do operatora stacjonarnego



Rys. 30 Statystyki połączenia z serwerem OpenVPN widoczne w aplikacji na telefon

3.3.3. Materiał wideo przedstawiający komunikację z serwerem SFTP po przyłączeniu się do sieci VPN

Na dołączonym filmie pt. *„Prezentacja_działania.mkv”* jako załącznik 2, zostało przedstawione działanie całej infrastruktury u klienta z systemem operacyjnym Windows 10. Komputer podłączony bezprzewodowo do hotspotu realizowanego przez telefon komórkowy, zestawia tunel do wirtualnej sieci prywatnej. To połączenie jest realizowane przez klienta serwera OpenVPN pod nazwą „PE-KIN_64_A”. Jednocześnie, telefon komórkowy także operuje jako klient serwera OpenVPN jako „Kacper”.

Na początku zaznaczono, poprzez wywołanie komendy w programie Windows PowerShell:

- Get-NetAdapter, że sterownik TAP-Windows, odpowiedzialny za zestawienie tunelu do serwera OpenVPN, nie jest zainstalowany;
- Invoke-WebRequest ifconfig.me/ip, że publiczny adres IP, którym PC identyfikuje się w sieci Internet, to 37.47.111.202;
- curl “http://myexternalip.com/raw”, iż publiczny adres IP zmienił się na oznaczony na Rys. 24, czyli związany z siecią VPN.

Po osadzeniu sterownika w systemie, widoczna jest kolejna karta sieciowa o nazwie Ethernet 4, która umożliwia zestawienie tunelu. Następnie, następuje wywołanie skryptów odpowiedzialnych za gromadzenie danych diagnostycznych.

Przy próbie przesłania pliku o większych gabarytach widać, że przepustowość wysyłania ograniczona jest połączeniem albo komputera stacjonarnego z telefonem, albo komórki z operatorem zewnętrznym w technologii LTE.

Serwer OpenVPN w czasie trwania materiału wyświetla aktualnie podłączonych klientów - są nimi wspomniany komputer stacjonarny oraz telefon. Po zakończeniu prac, zostaje wykorzystany skrypt do usunięcia sterownika TAP-Windows.

4. Podsumowanie wyników pracy

Dzięki zastosowaniu technologii wirtualizacji, został zrealizowany w pełni działający serwer SFTP. Umożliwiła ona uruchomienie kilku maszyn wirtualnych na jednym, fizycznym urządzeniu, przez co możliwe było zaoszczędzenie czasu (wygodny dostęp i tworzenie kolejnych VM), zasobów (podzespołów komputerów), energii elektrycznej, jak i znacząco uprościła zarządzanie infrastrukturą (brak nadmiarowego miejsca w szafach rackowych czy dodatkowych łączników; wszystkie usługi na jednym dysku). Każda z maszyn wirtualnych pełni ściśle określoną rolę, będąc skonfigurowaną dokładnie pod swoje zastosowanie, co ostatecznie umożliwia na dostęp do serwera SFTP z publicznej sieci Internet po zestawieniu tunelu do wirtualnej sieci prywatnej. Maszyny wirtualne są chronione mniej lub bardziej rygorystycznie, w zależności od przeznaczenia. Duże znaczenie w zarządzaniu całą infrastrukturą sieciową także pełnią skrypty, zarówno na maszynach wirtualnych, jak i przygotowane na nośniku pendrive. Dzięki nim użytkowanie środowiska jest bardziej intuicyjne, wygodne, a także zmniejsza ryzyko popełnienia błędu.

Główną trudność, w odczuciu Autora, przy realizowaniu projektu sprawiło konfigurowanie serwera VPN, ze względu na przestarzałą dokumentację oprogramowania. Wiele ustawień czy nawet modeli założeniowych działania, trzeba było dopasowywać z innych rozwiązań dostępnych na rynku.

Praca dyplomowa, ze względu na dużą liczbę użytych programów, może być rozszerzona o: dokładniejsze i bardziej szczegółowe filtry na zaporze sieciowej, wykorzystanie protokołu FTPS zamiast SFTP wraz z certyfikatami generowanymi z maszyny wirtualnej easyRSA, większą automatyzację i tak już ułatwionego procesu dodawania nowego klienta serwera OpenVPN.

5. Bibliografia

[1] *Address Allocation for Private Internets*, https://datatracker.ietf.org/doc/html/rfc1918, dostęp w dniu 22.11.2024

[2] *Proxmox Virtual Environment*, https://www.proxmox.com/en/proxmox-virtual-environment/overview, dostęp w dniu 06.01.2025

[3] *Dokumentacja Supermicro X8SIL-F*, https://www.supermicro.com/manuals/motherboard/3420/MNL-1130.pdf, dostęp w dniu 19.11.2024

[4] *History of Atlas Computer*, https://ethw.org/Milestones:Atlas_Computer_and_the_Invention_of_Virtual_Memory,_1957-1962, dostęp w dniu 23.10.2024

[5] A. Tanenbaum, H. Bos, *Modern Operating Systems, 5th Edition*, 2023, s. 209 – 215

[6] G. Popek, R. Goldberg, *Formal Requirements for Virtualizable Third Generation Architectures*, 1974, https://www.cs.cornell.edu/courses/cs6411/2018sp/papers/popek-goldberg.pdf, dostęp w dniu 22.10.2024

[7] A. Tanenbaum, H. Bos, *Modern Operating Systems, 5th Edition*, 2023, s. 472 - 495

[8] A. Tanenbaum, H. Bos, *Modern Operating Systems, 5th Edition*, Location of type 1 and type 2 hypervisors, 2023, s. 478

[9] *Architektura Proxmox*, https://community.veeam.com/blogs-and-podcasts-57/proxmox-virtual-environment-architecture-services-and-user-tools-8067, dostęp w dniu 23.11.2024

[10] Oskar Andreasson, *What is an IP filter*, 2006. https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#TCPIPLAYERS, dostęp w dniu 20.11.2024

[11] Oskar Andreasson, *Traversing of tables and chains*, 2006, https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#TCPIPLAYERS, dostęp w dniu 23.11.2024

[12] Oskar Andreasson, *User-land states*, 2006, https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#TCPIPLAYERS, dostęp w dniu 20.11.2024

[13] Eric Rescorla, *TLS 1.3 - RFC 8446*, https://datatracker.ietf.org/doc/html/rfc8446, dostęp w dniu 20.11.2024

[14] *SSL Handshake*, https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake, dostęp w dniu 23.11.2024
[15] *Podatność opcji 121 serwera DHCP w sieciach VPN, CVE-2024-3661*, https://nvd.nist.gov/vuln/detail/CVE-2024-3661, dostęp w dniu 20.11.2024
[16] *Infrastruktura PKI*, https://www.keyfactor.com/education-center/what-is-pki, dostęp w dniu 23.11.2024
[17] *Easy-RSA 3*, https://easy-rsa.readthedocs.io/en/latest/, dostęp w dniu 20.11.2024
[18] *SSH File Transfer Protocol draft*, https://datatracker.ietf.org/doc/html/draft-ietf-secsh-filexfer-13, dostęp w dniu 20.11.2024
[19] *Protokół SFTP*, https://certera.com/blog/https-vs-sftp-know-the-difference/, dostęp w dniu 07.01.2025

6. Spis załączników

Załącznik 1 - *skrypty powłoki Bash, Sh oraz Windows wykorzystane w projekcie opisane jako Dodatki*, Skrypty.pdf

Dodatek A - skrypt do ebtables, używany na hipernadzorcy Proxmox

Dodatek B - skrypt do podpisywania CSR, używany na VM 102

Dodatek C - skrypt do wysyłania certyfikatów, używany na VM 102

Dodatek D - instrukcja dodawania klienta serwera OpenVPN, używana na VM 101

Dodatek E - skrypt do tworzenia CSR, używany na VM 101

Dodatek F - skrypt do tworzenia pliku .OVPN, używany na VM 101

Dodatek G - skrypt do gromadzenia informacji o systemie, używany na pendrive

Dodatek H - skrypt do gromadzenia informacji o sieciach Wi-Fi, używany na pendrive

Dodatek I - skrypt do usuwania sterownika OpenVPN, używany na pendrive

Załącznik 2 - *wideo prezentujące podłączenie do systemu VPN oraz transfer plików w ramach protokołu SFTP*, Prezentacja_działania.mkv