UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Informatica

e dell'Automazione

Implementazione e analisi delle prestazioni di container Linux su router

Implementation and performance analysis of Linux containers on routers

Relatore:

Prof. Ennio Gambi

Correlatore:

Ing. Adelmo De Santis

Candidato:

Alessandro Sebastianelli

Anno Accademico 2024/2025

In memoria di mia madre

INDICE

Capitolo 1

Introduzione e nozioni fondamentali

L’obiettivo di questa tesi sperimentale è quello di implementare una “distro” ovvero distribuzione Linux all’interno di un router, in modo da estenderne le funzionalità. I dispositivi scelti per l’implementazione sono un router della Mikrotik con il sistema operativo RouterOS e un router della Aethra Telecommunications con il sistema operativo ATOS.

Una "distro Linux" è una distribuzione di Linux cioè una versione specifica del sistema operativo Linux contenente il kernel Linux e un insieme di software aggiuntivi come librerie di sistema, utility di gestione del sistema, ambienti desktop, programmi di utilità e applicazioni software preinstallate.

Le distribuzioni Linux sono create da comunità di sviluppatori e organizzazioni che uniscono il kernel Linux con una selezione di software aggiuntivi per creare un sistema operativo completo e funzionante. Esistono centinaia di distribuzioni Linux disponibili, ciascuna con le proprie caratteristiche, obiettivi e filosofie di sviluppo.

Alcune delle distribuzioni Linux più famose includono Ubuntu, Fedora, Debian, CentOS, Arch Linux, openSUSE e molti altri. Ognuna di queste distribuzioni può essere ottimizzata per una o più specifiche funzionalità come l'uso generico desktop, il server, l'embedded system, la sicurezza, l'anonimato online, la privacy, ecc... Gli utenti possono scegliere la distribuzione Linux che meglio si adatta alle proprie esigenze e obiettivi.

Il "kernel Linux" è il componente centrale del sistema operativo Linux. È il cuore del sistema, responsabile della gestione delle risorse hardware del computer e dell'esecuzione dei programmi. Il kernel Linux fornisce un'interfaccia tra il software delle applicazioni e l'hardware sottostante, consentendo alle applicazioni di comunicare con i dispositivi hardware senza dover conoscere i dettagli specifici di ciascun dispositivo.

Il kernel Linux gestisce diverse funzionalità come ad esempio la gestione della memoria ovvero assegna e gestisce la memoria del sistema per le applicazioni e i processi in esecuzione; la gestione dei processi ovvero crea, gestisce e termina i processi del sistema, consentendo loro di eseguire operazioni in modo efficiente; la gestione del sistema di file ovvero fornisce un'interfaccia per l'accesso e la gestione dei file sul disco; la gestione dei dispositivi ovvero fornisce driver per comunicare con dispositivi hardware come CPU, memoria, dispositivi di archiviazione, schede di rete e così via; la gestione della rete ovvero Gestisce le operazioni di rete, inclusa l'elaborazione dei pacchetti di rete e la gestione delle connessioni di rete.

Una "distro Linux over router" è una distribuzione Linux che è stata configurata e ottimizzata per essere utilizzata come router o gateway di rete. Questo significa che la distribuzione Linux

è stata adattata per eseguire funzionalità di routing, NAT (Network Address Translation), firewalling e altre operazioni di rete in modo efficace ed efficiente.

Le distribuzioni Linux over router sono spesso utilizzate in reti domestiche, aziendali o nelle infrastrutture di rete più complesse per fornire connettività Internet e gestire il traffico di rete. Possono essere configurate per funzionare su hardware dedicato, come router hardware o server, o possono essere virtualizzate su piattaforme di virtualizzazione.

Alcune delle distribuzioni Linux più comunemente utilizzate come router includono:

1. pfSense: Basato su FreeBSD, pfSense è una distribuzione popolare per router e firewall, noto per la sua facilità d'uso e potenti funzionalità di sicurezza.

2. OPNsense: Anch'esso basato su FreeBSD, OPNsense è un fork di pfSense con alcune caratteristiche aggiuntive e una comunità di sviluppo separata.

3. VyOS: VyOS è una distribuzione Linux basata su Debian che offre funzionalità di routing avanzate e un'interfaccia a riga di comando per la configurazione.

4. RouterOS (MikroTik): Sebbene non sia una distribuzione Linux, RouterOS di MikroTik è un sistema operativo basato su kernel Linux che viene implementato nei dispositivi di rete MikroTik, noti per la loro versatilità e potenza nelle reti.

Queste distribuzioni Linux over router offrono un'ampia gamma di funzionalità di rete e possono essere adattate alle esigenze specifiche della rete in cui vengono implementate.

RouterOS è un sistema operativo sviluppato da MikroTik, un'azienda lettone specializzata nella produzione di dispositivi di rete. Questo sistema operativo è progettato principalmente per essere eseguito su hardware MikroTik, che include router, switch, access point wireless e altri dispositivi di rete.

RouterOS è basato su un kernel Linux modificato e offre un'ampia gamma di funzionalità di routing, firewall, gestione della rete, sicurezza e altro ancora. È conosciuto per la sua flessibilità, robustezza e potenza, rendendolo una scelta popolare per reti di varie dimensioni e complessità.

Le caratteristiche principali di RouterOS includono:

1. Routing avanzato: RouterOS supporta protocolli di routing dinamico come OSPF, BGP, RIP, nonché il routing statico. Può essere configurato per gestire il traffico di rete in modo efficiente anche in reti complesse.

2. Firewall: RouterOS include un firewall potente e flessibile che consente agli amministratori di sistema di controllare il flusso del traffico di rete, applicare regole di sicurezza e proteggere la rete da intrusioni e attacchi informatici.

3. Gestione della banda: Consente la gestione della larghezza di banda e il controllo del traffico per ottimizzare le prestazioni della rete e garantire una distribuzione equa delle risorse di rete.

4. Wireless: RouterOS supporta funzionalità wireless complete, inclusa la configurazione di access point, bridge wireless, client e altri. Può essere utilizzato per creare reti wireless affidabili e ad alte prestazioni.

5. Hotspot: RouterOS include funzionalità di hotspot che consentono agli amministratori di sistema di creare aree Wi-Fi pubbliche o private con autenticazione e gestione degli utenti.

RouterOS è utilizzato in una vasta gamma di scenari, tra cui reti domestiche, aziendali, ISP (Internet Service Provider), reti di larga scala e molto altro ancora. MikroTik offre una serie di dispositivi hardware preinstallati con RouterOS, ma è anche possibile installare RouterOS su hardware x86 standard per personalizzare e adattare la propria soluzione di rete.

Ci sono diversi vantaggi nell'implementare una distribuzione Linux come sistema operativo per un router:

1. Flessibilità: Una distribuzione Linux offre un'ampia gamma di software e strumenti che possono essere utilizzati per personalizzare e adattare il router alle esigenze specifiche dell'ambiente di rete. Questo include la possibilità di installare e configurare servizi aggiuntivi come firewall, VPN, proxy, server DHCP, DNS e altro ancora.

2. Controllo completo: Con una distribuzione Linux, gli amministratori di rete hanno un controllo completo sul router e possono personalizzare ogni aspetto del suo funzionamento. Possono modificare il kernel, aggiungere moduli del kernel personalizzati, scrivere script di automazione e configurare le regole del firewall in modo dettagliato.

3. Sicurezza: Le distribuzioni Linux offrono un'ampia gamma di strumenti di sicurezza e protocolli che possono essere utilizzati per proteggere il router e la rete da intrusioni, attacchi informatici e minacce esterne. Gli amministratori possono configurare regole del firewall, monitorare il traffico di rete, implementare VPN sicure e altro ancora.

4. Costo: Molte distribuzioni Linux sono open source e gratuite da utilizzare, il che può ridurre i costi complessivi rispetto all'utilizzo di soluzioni proprietarie.

5. Aggiornamenti e supporto: Le distribuzioni Linux sono supportate da una vasta comunità di sviluppatori e utenti che forniscono aggiornamenti regolari, correzioni di bug e supporto tecnico tramite forum online, documentazione e altri canali di comunicazione.

6. Scalabilità: Le distribuzioni Linux sono progettate per essere altamente scalabili e possono essere utilizzate su una vasta gamma di hardware, da dispositivi embedded a server di grandi dimensioni. Ciò consente di adattare facilmente il router alle esigenze di crescita della rete.

L'implementazione di un "appliance su un router per concentrare le funzioni" si riferisce alla configurazione di un dispositivo di rete (il "router") in modo che funzioni anche come un "appliance" dedicato a compiti specifici, come ad esempio firewall, VPN, server DHCP, server DNS, sistema di monitoraggio del traffico di rete, e altro ancora. Questo approccio consente di consolidare diverse funzionalità di rete in un unico dispositivo, semplificando la gestione e ottimizzando l'uso delle risorse hardware disponibili.

Di solito, questa implementazione avviene utilizzando una distribuzione Linux o un sistema operativo specializzato progettato per funzionare su dispositivi di rete. Per esempio:

1. Distribuzione Linux: Una distribuzione Linux come pfSense, OPNsense, VyOS o simili può essere installata sul router per fornire una vasta gamma di funzionalità di rete. Queste distribuzioni sono progettate specificamente per fungere da appliance di rete e offrono strumenti per configurare e gestire servizi come firewall, VPN, proxy, server DHCP, server DNS e altro ancora.

2. Sistema operativo proprietario: Alcuni router possono essere equipaggiati con un sistema operativo proprietario che consente di abilitare funzionalità aggiuntive tramite moduli o plugin. Ad esempio, alcuni router Cisco utilizzano Cisco IOS o Cisco IOS XE, che consentono di attivare e configurare funzionalità specifiche come firewall, VPN e routing avanzato.

L'implementazione di un appliance su un router per concentrare le funzioni offre diversi vantaggi:

- Semplificazione della gestione: gestire un'unica piattaforma per diverse funzioni di rete è più semplice e richiede meno tempo e risorse rispetto a dover gestire dispositivi separati per ciascuna funzionalità.

- Ottimizzazione delle risorse: utilizzare un'unica piattaforma per diverse funzioni consente di ottimizzare l'uso delle risorse hardware disponibili, riducendo al minimo i costi e il consumo energetico.

- Integrazione e coerenza: avendo tutte le funzioni di rete integrate su un'unica piattaforma, è più facile garantire l'integrazione e la coerenza delle configurazioni e delle politiche di sicurezza.

- Flessibilità: è possibile aggiungere o rimuovere funzionalità in base alle esigenze di rete senza dover acquistare hardware aggiuntivo o sostituire dispositivi esistenti.

Un esempio di implementazione di un appliance su un router per concentrare le funzioni, focalizzandoci sull'implementazione di un firewall su un router:

1. Selezione del router: per iniziare è essenziale selezionare un router dotato di capacità hardware e supporto software idonei a operare come appliance di rete. Ad esempio un router che offre funzionalità avanzate di routing e firewall integrate

2. Scelta della distribuzione Linux o del sistema operativo: Se il router supporta l'installazione di software personalizzato, può essere scelta una distribuzione Linux specializzata per funzionare come firewall, come ad esempio pfSense o OPNsense. Queste distribuzioni offrono una vasta gamma di funzionalità di sicurezza e sono progettate per essere facili da configurare e gestire.

3. Installazione e configurazione del software del firewall: viene installata la distribuzione Linux scelta sul router e viene configurato il software del firewall per soddisfare le esigenze specifiche della rete. Vengono definite regole del firewall per controllare il traffico di rete in

entrata e in uscita, bloccare o consentire determinati tipi di traffico, implementare NAT (Network Address Translation) e altro ancora.

4. Integrazione con altre funzionalità di rete: Oltre al firewall, possono essere integrate altre funzionalità di rete sull'appliance, come ad esempio VPN, server DHCP e DNS, sistema di monitoraggio del traffico di rete e altro ancora. Questo consente di concentrare diverse funzioni di rete su un'unica piattaforma e semplificare la gestione della rete.

5. Test e monitoraggio: Una volta configurato l'appliance, è importante testare le configurazioni del firewall e monitorare il traffico di rete per garantire che il router stia svolgendo correttamente le funzioni di firewalling e che la rete stia operando come previsto.

6. Manutenzione e aggiornamenti: Infine, viene mantenuto l'appliance aggiornato con gli ultimi aggiornamenti di sicurezza e correzioni di bug per garantire la stabilità e la sicurezza della rete nel tempo.

Esempio di implementazione di un appliance su un router per concentrare le funzioni, in riferimento sull'implementazione di un centralino telefonico su un router:

1. Selezione del router: viene selezionato un router che abbia le capacità hardware e il supporto software necessari per funzionare come un appliance di rete con funzionalità di centralino telefonico. Potrebbe essere un router hardware dedicato con supporto per linee telefoniche e capacità di gestione delle chiamate, oppure un router che supporta l'installazione di software personalizzato.

2. Scelta della distribuzione Linux o del sistema operativo: Se il router supporta l'installazione di software personalizzato, viene scelta una distribuzione Linux specializzata per funzionare come centralino telefonico, come ad esempio Asterisk, FreeSWITCH o Kamailio. Queste distribuzioni offrono una vasta gamma di funzionalità per la gestione delle chiamate VoIP e sono ampiamente utilizzate per implementare centralini telefonici basati su IP.

3. Installazione e configurazione del software del centralino telefonico: viene installata la distribuzione Linux scelta sul router e configurato il software del centralino telefonico per soddisfare le esigenze specifiche della rete. Questo include la configurazione delle linee telefoniche, la definizione delle regole di instradamento delle chiamate, la configurazione delle funzionalità di segreteria telefonica, conferenza, trasferimento di chiamata, e altro ancora.

4. Integrazione con altre funzionalità di rete: Oltre al centralino telefonico, possono essere integrate altre funzionalità di rete sull'appliance, come ad esempio firewall, VPN, server DHCP e DNS, sistema di monitoraggio del traffico di rete e altro ancora. Questo consente di concentrare diverse funzioni di rete su un'unica piattaforma e semplificare la gestione della rete.

5. Test e monitoraggio: Una volta configurato l'appliance, è importante testare le configurazioni del centralino telefonico e monitorare il traffico telefonico per garantire che il

router stia svolgendo correttamente le funzioni di centralino e che la rete telefonica stia operando come previsto.

6. Manutenzione e aggiornamenti: Infine, assicurare di mantenere l'appliance aggiornato con gli ultimi aggiornamenti di sicurezza e correzioni di bug per garantire la stabilità e la sicurezza della rete telefonica nel tempo.

In questo modo, viene implementato con successo un centralino telefonico su un router per concentrare le funzioni, utilizzando una distribuzione Linux specializzata e sfruttando le capacità hardware del router per fornire una soluzione telefonica integrata e centralizzata.

Utilizzare una distribuzione Linux su un router può portare a risparmi significativi nelle architetture SMB in diversi modi:

1. Costo del software: Molte distribuzioni Linux sono open source e gratuite da utilizzare, il che significa che non è necessario pagare licenze software per ogni dispositivo o utente che si connette alla rete. Questo può ridurre notevolmente i costi di licenza rispetto all'uso di soluzioni proprietarie, come i server Windows.

2. Hardware meno costoso: Le distribuzioni Linux sono note per essere leggere e efficienti, il che significa che possono essere eseguite su hardware meno costoso rispetto a sistemi operativi proprietari come Windows Server. Questo permette di risparmiare sui costi hardware sia per i router che per i server di rete.

3. Flessibilità e personalizzazione: Le distribuzioni Linux offrono molti strumenti e software che possono essere utilizzati per configurare e gestire una rete SMB. Con una distribuzione Linux, è possibile personalizzare la configurazione del router e dei server di rete in base alle esigenze specifiche dell'organizzazione, senza dover fare affidamento su soluzioni preconfezionate e costose.

4. Supporto della comunità: Le distribuzioni Linux sono supportate da tanti sviluppatori e utenti che forniscono assistenza, risorse e aggiornamenti gratuiti. Questo può ridurre la dipendenza da costosi contratti di supporto tecnico e consentire alle organizzazioni di risparmiare sui costi di manutenzione e assistenza.

5. Sicurezza: Le distribuzioni Linux sono conosciute per la loro robustezza e sicurezza, e possono essere configurate per offrire un alto livello di protezione per la rete SMB. Con le distribuzioni Linux, è possibile implementare soluzioni di sicurezza avanzate come firewall, VPN, monitoraggio del traffico di rete e altro ancora, senza dover acquistare software di terze parti.

Per concludere, utilizzare una distribuzione Linux su un router può portare a risparmi significativi nelle architetture SMB, sia in termini di costi software che hardware, oltre a offrire maggiore flessibilità, personalizzazione e sicurezza rispetto alle soluzioni proprietarie.

Utilizzando una distribuzione Linux su un router, si ottiene inoltre un altro vantaggio come una maggiore scalabilità nelle architetture di rete per diverse ragioni:

1. Configurazione flessibile: Le distribuzioni Linux offrono una vasta gamma di opzioni di configurazione che consentono di adattare il router alle esigenze specifiche della rete. È possibile configurare il router per supportare nuove funzionalità o servizi aggiuntivi man mano che la rete cresce, senza dover cambiare l'hardware sottostante o acquistare nuovi dispositivi.

2. Supporto hardware ampio: Le distribuzioni Linux sono progettate per essere eseguite su tante gamme di hardware, dai dispositivi embedded a server di grandi dimensioni. Ciò significa che è possibile scegliere l'hardware più adatto alle esigenze di crescita della rete e sostituire o aggiornare il router in modo rapido ed efficiente quando necessario.

3. Scalabilità orizzontale: Utilizzando una distribuzione Linux su un router, è possibile implementare una scalabilità orizzontale, ovvero aggiungere nuovi router o nodi alla rete per aumentare la capacità e la copertura della rete. Questo approccio consente di distribuire il carico di lavoro su più dispositivi e di gestire più utenti e dispositivi senza compromettere le prestazioni.

4. Clustering e bilanciamento del carico: alcune distribuzioni Linux supportano funzionalità avanzate come il clustering e il bilanciamento del carico, che consentono di distribuire il carico di lavoro su più nodi di rete in modo equo ed efficiente. Questo può migliorare le prestazioni della rete e garantire una maggiore disponibilità dei servizi per gli utenti.

5. Virtualizzazione: Utilizzando una distribuzione Linux su un router virtualizzato, è possibile creare istanze multiple del router su un singolo hardware fisico utilizzando tecnologie di virtualizzazione come KVM, Xen o VMware. Questo permette di consolidare più funzioni di rete su un'unica piattaforma e di ridurre i costi e la complessità della gestione della rete.

Utilizzando una distribuzione Linux su un router, è possibile migliorare la ridondanza della rete in diversi modi:

1. Failover di rete: Utilizzando protocolli di routing dinamico come OSPF (Open Shortest Path First) o BGP (Border Gateway Protocol), è possibile configurare il router Linux per monitorare lo stato della rete e reindirizzare automaticamente il traffico attraverso percorsi alternativi in caso di guasto di un collegamento di rete o di un router. Questo assicura che la rete rimanga operativa anche in presenza di guasti hardware o di rete.

2. High Availability (HA): Utilizzando software di clustering e failover come Keepalived o Pacemaker, è possibile configurare il router Linux in un ambiente ad alta disponibilità in cui più router condividono un indirizzo IP virtuale. In caso di guasto di uno dei router, gli altri router prendono automaticamente il controllo del traffico per garantire la continuità del servizio.

3. Redundancy Protocol: diverse distribuzioni Linux supportano protocolli di ridondanza specifici come VRRP (Virtual Router Redundancy Protocol) o HSRP (Hot Standby Router Protocol), che consentono a più router di collaborare per fornire una connettività di rete ridondante. Questi protocolli consentono a un router di assumere il ruolo di gateway predefinito per la rete in caso di guasto del router primario.

4. Clustering e virtualizzazione: utilizzando tecnologie di clustering e virtualizzazione, è possibile creare cluster di router virtuali su più nodi di rete fisici. In caso di guasto di un nodo, il carico di lavoro viene distribuito automaticamente sui nodi rimanenti per garantire la continuità del servizio.

5. Backup e ripristino: utilizzando regolarmente backup delle configurazioni e delle impostazioni del router Linux, è possibile ripristinare rapidamente il router in caso di guasto hardware o di corruzione del software. Questo assicura che la configurazione del router possa essere rapidamente ripristinata e che la rete possa tornare online il prima possibile.

Utilizzare una distribuzione Linux su un router offre diverse opzioni per migliorare la ridondanza della rete, consentendo di configurare percorsi alternativi, implementare soluzioni di alta disponibilità e sfruttare tecnologie di clustering e virtualizzazione per garantire la continuità del servizio in caso di guasto.

In particolare l'implementazione di una distribuzione Linux over router ci permette di realizzare anche altri scenari più specifici sfruttando il POC (Proof of Concept) ovvero è un termine utilizzato per descrivere una dimostrazione pratica o un esperimento che ha lo scopo di verificare la fattibilità o la validità di un'idea, di un concetto o di un progetto. E' un'attività che mira a dimostrare che un determinato concetto o tecnologia può funzionare in pratica e può essere implementato con successo.

Quando un utente si connette alla rete Wi-Fi dell'ateneo, viene reindirizzato automaticamente a una pagina web speciale, chiamata "captive portal", anziché accedere direttamente a Internet. Questa pagina di destinazione può richiedere all'utente di effettuare l'accesso con le proprie credenziali (ad esempio, nome utente e password dell'account dell'ateneo) o di accettare determinate condizioni d'uso, come termini e condizioni, politiche sulla privacy o accordi di utilizzo della rete.

Una volta completata l'autenticazione o l'accettazione delle condizioni, l'utente viene reindirizzato nuovamente alla pagina iniziale che voleva visitare e può accedere normalmente a Internet tramite la rete Wi-Fi dell'ateneo.

I captive portal sono spesso utilizzati per garantire la sicurezza della rete e per controllare l'accesso degli utenti, oltre a consentire alle istituzioni di monitorare e registrare l'uso della rete per scopi di sicurezza e gestione delle risorse.

Implementare un captive portal di ateneo su una virtual machine (VM) Linux ospitata su un router può essere un'opzione efficace e flessibile. I passaggi per la configurazione sono:

1. Configurazione del router: Inizialmente, è necessario configurare il router per instradare correttamente il traffico verso la VM Linux che ospiterà il captive portal. Questo coinvolge la configurazione delle interfacce di rete del router e la definizione delle regole di instradamento per consentire al traffico di raggiungere la VM.

2. Installazione della VM Linux: Successivamente, è necessario installare una distribuzione Linux sulla VM. Puoi utilizzare una distribuzione Linux leggera e ottimizzata per le VM, come ad esempio Ubuntu Server o Debian.

3. Configurazione della rete della VM: Una volta installata la distribuzione Linux, è necessario configurare correttamente le interfacce di rete della VM in modo che possa comunicare con il router e con gli altri dispositivi di rete. E' importante assicurarsi di assegnare un indirizzo IP valido alla VM.

4. Installazione del software del captive portal: Successivamente, è necessario installare il software del captive portal sulla VM Linux. Viene utilizzato un software open-source come CoovaChilli, PacketFence, o altri, a seconda delle esigenze.

5. Configurazione del captive portal: Una volta installato il software del captive portal, è necessario configurarlo per soddisfare le esigenze specifiche dell' ambiente di rete e dell'ateneo. Ciò include la configurazione dell'autenticazione degli utenti, la personalizzazione della pagina di accesso, la definizione delle politiche di utilizzo della rete, e altro ancora.

6. Test e monitoraggio: Infine, è importante testare il captive portal per assicurarsi che funzioni correttamente e monitorare le prestazioni e l'utilizzo della rete una volta che è in funzione. Vengono effettuati test di accesso da dispositivi diversi e la verifica che gli utenti possano accedere correttamente alla rete dopo l'autenticazione tramite il captive portal.

Quindi Implementare un captive portal di ateneo su una VM Linux ospitata su un router offre flessibilità e scalabilità, consentendo di gestire in modo centralizzato l'accesso alla rete e fornire una soluzione di autenticazione sicura per gli utenti.

Quindi implementare una distribuzione Linux su un router consente un controllo di accesso a basso costo con software sviluppato internamente offrendo flessibilità, personalizzazione, sicurezza e supporto della comunità permettendo alle organizzazioni di gestire in modo efficace e conveniente l'accesso alla rete e di garantire la sicurezza e l'affidabilità delle loro infrastrutture di rete.

Installare una distribuzione Linux su un router permette di ottimizzare le prestazioni del dispositivo e di evitare la necessità di un server dedicato. Un server dedicato è un hardware riservato esclusivamente a un singolo cliente o a una specifica applicazione. In altre parole, con Linux su un router, è possibile gestire funzioni che altrimenti richiederebbero un server separato, semplificando così l'infrastruttura.

In sintesi, implementare una distribuzione Linux su un router consente di ottimizzare gli apparati riducendo la necessità di un server dedicato e consentendo di consolidare più funzioni di rete su un singolo dispositivo. Ciò porta a risparmi sui costi, semplificazione della gestione, miglioramento delle prestazioni e maggiore scalabilità dell'infrastruttura di rete.

1.1 Container

I container sono un tipo di tecnologia di virtualizzazione che consente di eseguire e isolare applicazioni e i relativi ambienti di esecuzione in modo efficiente e leggero. Piuttosto che virtualizzare un'intera macchina virtuale (VM) con un sistema operativo completo, i container virtualizzano solo il livello dell'applicazione, consentendo a più container di condividere il kernel del sistema operativo ospite.

Le caratteristiche principali dei container sono:

1. Isolamento delle risorse: i container forniscono un livello di isolamento delle risorse, come CPU, memoria, spazio di archiviazione e larghezza di banda di rete, per garantire che le applicazioni eseguite all'interno di un container non interferiscano con altre applicazioni o container sullo stesso host.

2. Portabilità: i container sono altamente portabili e possono essere eseguiti su qualsiasi sistema operativo o infrastruttura che supporta il motore di containerizzazione utilizzato, come Docker o Kubernetes. Questo consente agli sviluppatori di creare, testare e distribuire facilmente le proprie applicazioni su più ambienti, inclusi ambienti di sviluppo, test e produzione.

3. Leggerezza: i container sono leggeri e veloci da avviare, poiché condividono il kernel del sistema operativo ospite anziché virtualizzare un intero sistema operativo separato come fanno le macchine virtuali. Questo consente di risparmiare risorse di sistema e di migliorare le prestazioni complessive dell'ambiente di esecuzione.

4. Agilità: grazie alla loro leggerezza e portabilità, i container consentono di creare, distribuire e scalare facilmente le applicazioni in modo rapido ed efficiente. Gli sviluppatori possono implementare nuove versioni delle applicazioni, distribuire aggiornamenti e ridimensionare orizzontalmente o verticalmente le risorse in base alle esigenze senza interruzioni del servizio.

5. Gestione centralizzata: i container possono essere gestiti centralmente utilizzando strumenti di gestione dei container come Docker Swarm o Kubernetes. Questi strumenti consentono di orchestrare e coordinare l'esecuzione dei container su più host, monitorare le prestazioni, scalare automaticamente le risorse e gestire la distribuzione e l'aggiornamento delle applicazioni in modo efficiente.

Riassumendo i container offrono un'alternativa leggera e portatile alla virtualizzazione tradizionale delle macchine virtuali, consentendo agli sviluppatori di creare, distribuire e gestire facilmente le proprie applicazioni in ambienti distribuiti e complessi. Questa tecnologia è diventata sempre più popolare negli ultimi anni grazie alla sua agilità, efficienza e scalabilità.

1.2 Docker

In riferimento ai container è importante parlare di Docker che è una delle tecnologie più popolari per la loro gestione ed è ampiamente utilizzato nell'ambito dello sviluppo e del deployment delle applicazioni. I concetti fondamentali:

1. Immagini Docker: Docker utilizza immagini per creare e distribuire container. Un'immagine Docker è un pacchetto autonomo che contiene tutto il necessario per eseguire un'applicazione, inclusi il codice, le librerie, le dipendenze e le configurazioni. Le immagini Docker sono create utilizzando un file di definizione chiamato Dockerfile.

2. Container Docker: un container Docker è un'istanza in esecuzione di un'immagine Docker. I container sono eseguiti isolati dagli altri container e dall'ambiente di hosting, ma condividono il kernel del sistema operativo ospite. I container Docker possono essere avviati, arrestati, arrestati e gestiti in modo flessibile utilizzando il client Docker e i comandi Docker.

3. Docker Engine: è il motore di esecuzione di Docker che gestisce l'avvio e l'esecuzione dei container. È responsabile della gestione delle risorse del sistema, dell'isolamento dei container e della comunicazione con il sistema operativo ospite per l'esecuzione dei container. Docker Engine include anche il daemon Docker e il client Docker.

4. Docker Compose: è uno strumento per la definizione e la gestione di applicazioni multi-container in Docker. Consente agli sviluppatori di definire più servizi e le loro dipendenze in un file di composizione (docker-compose.yml) e di avviare e gestire facilmente l'intera applicazione con un singolo comando.

5. Docker Swarm: è uno strumento per la gestione di cluster di container Docker. Consente di creare e gestire un cluster di nodi Docker per distribuire, scalare e gestire applicazioni containerizzate su più host. Docker Swarm fornisce funzionalità di orchestrazione e gestione dei container, come ad esempio il bilanciamento del carico, il failover, il ripristino automatico e altro ancora.

Docker è una potente tecnologia per la gestione dei container che semplifica la creazione, la distribuzione e la gestione delle applicazioni containerizzate. Fornisce un'infrastruttura flessibile e scalabile per lo sviluppo e il deployment delle applicazioni, consentendo agli sviluppatori di concentrarsi sulla creazione di applicazioni di qualità e di alto valore.

Capitolo 2

Mikrotik

Per l'implementazione di una distribuzione Linux viene scelto il router della Mikrotik come apparato di rete. Un router della Mikrotik ha le seguenti caratteristiche:

1. RouterOS: il sistema operativo principale utilizzato sui dispositivi MikroTik è chiamato RouterOS. RouterOS è un sistema operativo basato su Linux che fornisce funzionalità avanzate di routing, firewalling, wireless, VPN, gestione di rete e altro ancora. È noto per la sua flessibilità e potenza, consentendo agli utenti di configurare e gestire reti complesse e scalabili.

2. Hardware: MikroTik offre una vasta gamma di hardware di rete, tra cui router, switch, access point wireless e dispositivi per l'installazione all'aperto. I dispositivi MikroTik sono conosciuti per le loro prestazioni affidabili, la robustezza e la capacità di gestire carichi di lavoro intensi.

3. Interfacce di gestione: MikroTik offre diverse interfacce di gestione per configurare e monitorare i dispositivi di rete. Queste includono un'interfaccia web basata su browser chiamata Webfig, un'interfaccia grafica utente chiamata Winbox e un'interfaccia a riga di comando (CLI) accessibile tramite SSH o Telnet.

4. Funzionalità avanzate: RouterOS offre una vasta gamma di funzionalità avanzate, tra cui routing statico e dinamico, supporto per protocolli di routing come OSPF e BGP, firewalling avanzato con supporto per NAT, filtraggio del traffico, controllo degli accessi e VPN.

5. Community attiva: MikroTik ha una comunità attiva di utenti, sviluppatori e professionisti di rete che condividono conoscenze, esperienze e risorse su forum online, gruppi di discussione e siti web dedicati. Questa comunità è una risorsa preziosa per ottenere supporto, consigli e suggerimenti relativi all'uso e alla configurazione dei dispositivi MikroTik.

In particolare è stato utilizzato il router MikroTik RB5009UG+S+IN composto da una scheda con 9 porte in rame e una USB 3.0 di dimensioni standard, 7 porte sono Gigabit Ethernet, un'altra è 2.5 Gigabit Ethernet e l'ultima è una gabbia SFP+ da 10G. Tutte le porte sono collegate a un potente switch-chip della famiglia Marvell Amethyst con una linea full-duplex da 10 Gbps che porta alla CPU Marvell Armada quad-core ARMv8 da 1,4 GHz. Sia la cpu che il chip di commutazione si trovano sul fondo della scheda, quindi il case funge da enorme dissipatore di calore. Può essere alimentato in tre modi diversi ovvero con PoE -in dalla porta ethernet 1, con il jack CC con una tensione di ingresso di 27-54 V o tramite terminale a 2 pin sul lato Le sempre con una tensione di ingresso di 27-54 V. Le schede sono dotate di 1 GB di RAM DDR4 e 1 GB di memoria NAND. Questa combinazione di porte e componenti fornisce quasi il doppio delle prestazioni con carichi pesanti della cpu. Altre specifiche di questo router sono: l'architettura a braccio 64 bit, un processore 88F7040, 4 core della cpu, una frequenza nominale della cpu a 350-1400 MHz, 1Gb di dimensione ram, 1Gb di spazio di

archiviazione, MTBF circa 200.000 ore a 25 C, temperatura ambiente testata da -40 C a 60 C, una porta USB 3.0 A con corrente massima 1.5.

La virtualizzazione delle funzioni di rete (NFV) astrae le funzioni di rete, consentendo loro di essere installate, controllate e manipolate da software in esecuzione su nodi di calcolo standardizzati. La NFV incorpora tecnologie di virtualizzazione e cloud per favorire lo sviluppo rapido di nuovi servizi di rete con scala e automazione elastiche. Queste tecnologie sono spesso raggruppate come NFV e SDN (software-defined networking).

Il desiderio di automatizzare l'orchestrazione e la gestione della rete, dello storage e delle risorse di calcolo è un fattore chiave di sviluppo nell'ambito delle tecnologie NFV e SDN. Ad esempio, uno scenario che include un server fisico con 10 VM o centinaia di container. Non scalerebbe mai, se fossero necessarie operazioni manuali. Grazie all'automazione, è possibile accelerare o distruggere rapidamente le funzioni di rete virtualizzate (VNF), come VM, container, router, firewall e sistemi di prevenzione delle intrusioni (IPS), al fine di scalare in modo elastico le funzioni di rete per soddisfare la richiesta dinamica.

La tecnologia NFV si contraddistingue per la sua agilità nel fornire servizi di rete con efficienza crescente, eliminando i colli di bottiglia imposti dai processi manuali e consentendo di implementare nuovi servizi su richiesta. Consente inoltre ai fornitori di servizi di offrire servizi più velocemente e in modo economicamente vantaggioso e permette di sfruttare l'automazione in modo da potersi adattare alle esigenze dei clienti per quanto riguarda scalabilità e agilità

L'architettura modulare NFV è ciò che consente ai fornitori di servizi di automatizzare a ogni livello. I principali componenti dell'architettura includono:

- Modulo di costruzione dell'infrastruttura NFV (NFVI): fornisce il livello di virtualizzazione (hypervisor o sistemi di gestione dei container come Kubernetes) e i componenti di elaborazione fisica, storage e networking che ospitano le VNF. La tecnologia NFVI è gestita attraverso il gestore dell'infrastruttura NFVI (VIM), il quale controlla l'allocazione delle

risorse per le VNF. OpenStack è un esempio di VIM open source che controlla le risorse fisiche e virtuali. La piattaforma Red Hat OpenStack è un esempio di VIM commerciale.

- VNF: applicazioni basate su software che forniscono uno o più servizi di rete. Le VNF utilizzano l'infrastruttura virtualizzata fornita da NFVI per connettersi alla rete e forniscono servizi di rete programmabili e scalabili. I gestori VFN supportano il ciclo di vita delle istanze VNF e la gestione di un software VNF.
- Gestione e orchestrazione (MANO): fornisce la gestione e l'orchestrazione VNF complessiva nell'architettura NFV. La soluzione MANO istanzia i servizi di rete attraverso l'automazione, la fornitura e il coordinamento dei flussi di lavoro verso i gestori VIM e VFN, i quali istanziano, a loro volta, le VNF e le catene di servizio di overlay networking. MANO connette l'architettura NFV con l'OSS/BSS esistente.

2.1 Configurazione router Mikrotik

Come primo step iniziale, viene creata l'immagine sotto Linux con Docker file utilizzando i seguenti comandi:

- Docker file per compilare richiede: apt install qemu-user-static binfmt-support utilizzato su sistemi basati su Debian o Ubuntu per installare due pacchetti importanti relativi all'esecuzione di binari di architetture diverse sulla tua macchina. Spiegazione dei due pacchetti:

1. **qemu-user-static:**
    - QEMU (Quick EMUlator) è uno strumento che fornisce un ambiente di emulazione di macchine virtuali. In questo contesto, **qemu-user-static** si concentra sulla parte di emulazione utente di QEMU. Questo pacchetto consente di eseguire binari destinati a un'architettura diversa (come ARM su un sistema x86) sulla tua macchina host senza la necessità di una macchina virtuale completa.
2. **binfmt-support:**
    - Questo pacchetto fornisce il supporto per i "binfmt handlers" nel kernel Linux. Binfmt (binary format) è un framework nel kernel che consente di registrare handler per formati di file binari specifici. In questo caso, viene utilizzato per associare i binari di architetture diverse a QEMU, consentendo loro di essere eseguiti sulla macchina host senza dover configurare una macchina virtuale separata.

Il commando "docker buildx build --platform arm64 -t debian_ssh_ale " è utilizzato per costruire un'immagine Docker utilizzando Buildx, una versione estesa del comando **docker build** che supporta la costruzione di immagini multi-architettura. Spiegazione più dettagliata:

1. **docker buildx build:**
    - Questo comando indica a Docker di utilizzare Buildx per la costruzione dell'immagine. Buildx è un'estensione di Docker Build che consente di costruire immagini Docker per diverse architetture in modo più flessibile e efficiente.
2. **--platform arm64:**
    - Questa opzione specifica la piattaforma di destinazione per la quale verrà costruita l'immagine. In questo caso, l'architettura di destinazione è **arm64** (architettura a 64 bit per ARM). Questo è utile quando si vuole creare un'immagine Docker specifica per un'architettura diversa da quella della macchina host.
3. **-t debian_ssh_ale:**
    - Questa opzione imposta il nome dell'immagine Docker risultante. Nel tuo caso, l'immagine verrà chiamata **debian_ssh_ale**.

Il comando "docker save debian_ssh_ale > debian.tar" viene utilizzato per esportare un'immagine Docker in un file di archivio tarball. Andando nello specifico:

1. **docker save debian_ssh_ale:**
    - **docker save** è il comando che consente di esportare un'immagine Docker in un formato di archivio. In questo caso, **debian_ssh_ale** è il nome dell'immagine Docker che si desidera esportare.
2. **> debian.tar:**
    - L'operatore **>** è utilizzato per redirigere l'output del comando precedente verso un file anziché visualizzarlo a schermo. In questo caso, l'output (l'archivio tarball dell'immagine Docker) viene salvato nel file chiamato **debian.tar**.

Il comando “ftp admin@192.168.188.159” (viene caricata l'immagine nello spazio file della MKT) viene utilizzato per stabilire una connessione FTP con un server remoto. Nel dettaglio:

1. **ftp:**
    - **ftp** è il comando per avviare un client FTP da linea di comando. Questo comando consente agli utenti di trasferire file tra il loro sistema locale e un server remoto utilizzando il protocollo FTP.
2. **admin@192.168.188.159:**
    - Questa parte specifica l'indirizzo IP del server FTP a cui ci si sta connettendo e l'utente con cui si desidera effettuare l'accesso. Nello specifico, l'indirizzo IP è **192.168.188.159**, e l'utente è **admin**. Quando si esegue questo comando, il client FTP cercherà di stabilire una connessione con il server FTP su quell'indirizzo IP usando le credenziali dell'utente specificato.

Una volta eseguito il comando, il client FTP chiederà la password per l'utente **admin**. Se la password è corretta, il client FTP stabilirà una connessione con il server FTP e mostrerà un prompt da cui è possibile impartire comandi FTP per navigare nel server remoto, visualizzare e trasferire file tra il sistema locale e il server. I comandi comuni includono **ls** per elencare i file, **get** per scaricare un file, **put** per caricare un file e altri comandi specifici del protocollo FTP

Considerando invece il router Mikrotik viene creato il container tramite il docker file nella sezione container del router Mikrotik tramite i seguenti passaggi:

Spiegazione “interface: vethx Root Dir” (sul disco esterno):

1. **vethx (Virtual Ethernet Pair):**
    - Come accennato in precedenza, le interfacce veth sono comunemente utilizzate in contesti di containerizzazione. Ogni container Docker viene solitamente fornito con una propria interfaccia di rete virtuale, spesso identificata con il prefisso "veth". Questa interfaccia è collegata al bridge della rete Docker e consente al container di comunicare con il resto della rete.
2. **Root Dir (Directory Radice):**
    - In Docker, il "Root Dir" si riferisce alla directory all'interno del filesystem del container che funge da radice per tutte le operazioni di sistema all'interno del container. È in questa directory che risiedono tutti i file e le directory che costituiscono il filesystem del container.

Quindi il container è stato configurato con un'interfaccia veth e che è stata assegnata una directory radice nel suo filesystem.

Spiegazione del comando “mounts debian3_root -> /usb1-part1/rootdir3” : è una rappresentazione testuale della configurazione dei mount (montaggi) in un sistema Debian, dove si sta montando il contenuto della directory debian3_root nel percorso /usb1-part1/rootdir3. Andando più nel dettaglio:

1. **mounts:**
    - Il termine "mounts" si riferisce alle operazioni di montaggio dei filesystem. In un sistema operativo Unix-like, il comando **mount** viene utilizzato per associare un filesystem a un determinato punto di montaggio nel sistema di file.
2. **debian3_root:**
    - Questo potrebbe essere il nome o l'identificatore di un filesystem o di una directory che si desidera montare. Potrebbe rappresentare la radice del filesystem di un'istanza di Debian (ad esempio, il filesystem di una particolare installazione Debian).
3. **->:**
    - La freccia (**->**) indica che il contenuto di **debian3_root** viene montato in **/usb1-part1/rootdir3**.
4. **/usb1-part1/rootdir3:**
    - Questa è la destinazione o il punto di montaggio nel sistema di file dove si desidera collegare il contenuto di **debian3_root**. Nel contesto dei mount, "/usb1-part1/rootdir3" rappresenta il percorso nel quale il contenuto di **debian3_root** sarà accessibile.

Il commando “envs debian_ennvs key TZ value Europe/Rome” viene utilizzato per configurare le variabili ambiente in un sistema Debian. Spiegazione nel dettaglio:

1. **envs:**
    - Il termine "envs" è utilizzato per gestire o configurare le variabili d'ambiente.
2. **debian_ennvs:**
    - è il nome o l'identificatore di un contesto o di un'istanza specifica di Debian in cui si desidera configurare le variabili d'ambiente.
3. **key TZ value Europe/Rome:**
    - Questa parte del comando indica la configurazione di una variabile d'ambiente specifica. In particolare:
        - **key:** rappresenta il nome della variabile d'ambiente. In questo caso, sembra essere "TZ".
        - **value:** Rappresenta il valore assegnato alla variabile d'ambiente, che è "Europe/Rome". Nell'ambito delle variabili d'ambiente, "TZ" spesso si riferisce al fuso orario (Time Zone).

I seguenti comandi fanno riferimento alla descrizione del Dockerfile:

il comando “FROM debian:stable” è un’istruzione di base utilizzata nei Dockerfile. Andando nel dettaglio:

1. **FROM:**

- L'istruzione **FROM** è una delle prime istruzioni che compare in un Dockerfile e specifica l'immagine di base da cui costruire la nuova immagine.

2. **debian:stable:**
    - Questa parte del comando specifica l'immagine di base che deve essere utilizzata. Nello specifico, **debian:stable** indica di utilizzare l'immagine ufficiale di Debian con la sua versione "stable" (stabile). Questo significa che l'immagine di base sarà la versione stabile più recente di Debian disponibile al momento della costruzione dell'immagine Docker.

Quindi, quando si utilizza **FROM debian:stable** in un Dockerfile, si sta effettivamente dicendo a Docker di iniziare la costruzione dell'immagine partendo da un'immagine base Debian ufficiale che rappresenta la versione stabile di Debian.

Il comando “RUN apt-get update” è un’istruzione all’interno di Dockerfile:

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questa istruzione consente di installare software, aggiornare pacchetti, eseguire configurazioni, ecc.
2. **apt-get update:**
    - Questo comando fa parte del sistema di gestione dei pacchetti di Debian e Ubuntu. In particolare, **apt-get update** è utilizzato per scaricare l'elenco più recente dei pacchetti disponibili dai repository configurati nel sistema.

Quando viene eseguita questa istruzione in un Dockerfile, sta effettivamente aggiornando la cache dei pacchetti del sistema all'interno dell'immagine Docker. Questo è spesso il primo passo nei Dockerfile che coinvolgono l'installazione di pacchetti, perché garantisce che il sistema sia consapevole delle versioni più recenti dei pacchetti disponibili prima di procedere con l'installazione effettiva.

Il comando “RUN apt install openssh-server -y” viene utilizzata per installare il pacchetto OpenSSH Server all’interno di un’immagine Docker.

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere l'installazione di pacchetti, configurazioni di sistema, copia di file, ecc.
2. **apt install openssh-server:**
    - Questa parte del comando utilizza il sistema di gestione dei pacchetti di Debian per installare il pacchetto **openssh-server**. Il comando **apt** è uno strumento di gestione dei pacchetti che facilita l'installazione, l'aggiornamento e la rimozione di software su sistemi Debian e Ubuntu.
3. **-y:**
    - L'opzione **-y** indica di accettare automaticamente tutte le conferme richieste durante l'installazione. In pratica, questa opzione rende non interattiva l'installazione, impedendo che venga richiesta la conferma dell'utente durante il processo di installazione.

Quindi, quando viene eseguita questa istruzione in un Dockerfile, il sistema all'interno dell'immagine Docker installerà automaticamente il pacchetto **openssh-server**. Questo è comune quando si costruiscono immagini Docker che devono fornire un servizio SSH all'interno dei container, ad esempio per scopi di sviluppo o per configurazioni specifiche di un'applicazione.

Il commando "RUN apt install net-tools iputils-ping traceroute -y" viene utilizzata per installare tre pacchetti specifici (net-tools, iputils-ping, traceroute) all'interno di un'immagine Docker.

1. **apt install net-tools iputils-ping traceroute:**
    - Questa parte del comando utilizza il sistema di gestione dei pacchetti di Debian per installare tre pacchetti specifici:
        - **net-tools:** fornisce un insieme di strumenti di rete, tra cui **ifconfig** e altri comandi utili per la configurazione e il monitoraggio della rete.
        - **iputils-ping:** fornisce il comando **ping** che è utilizzato per testare la connettività di rete.
        - **traceroute:** fornisce il comando **traceroute** che è utilizzato per tracciare il percorso di un pacchetto attraverso una rete.
2. **-y:**
    - L'opzione **-y** indica di accettare automaticamente tutte le conferme richieste durante l'installazione. In questo modo, il processo di installazione non richiede l'interazione dell'utente e procede automaticamente.

Il comando RUN "apt install vim nano iproute2 iptables build-essential -y" viene utilizzato per installare una serie di pacchetti all'interno di un'immagine Docker.

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere l'installazione di pacchetti, configurazioni di sistema, copia di file, ecc.
2. **apt install vim nano iproute2 iptables build-essential:**
    - Questa parte del comando utilizza il sistema di gestione dei pacchetti di Debian per installare cinque pacchetti specifici:
        - **vim e nano:** Sono editori di testo che sono spesso preferiti dagli utenti per la modifica dei file di configurazione o di testo all'interno di un sistema.
        - **iproute2:** Fornisce una suite di comandi per interagire con le funzionalità di routing del kernel e con altri aspetti del networking.
        - **iptables:** È utilizzato per configurare regole del firewall iptables.
        - **build-essential:** Contiene pacchetti essenziali necessari per la compilazione di software, compreso il compilatore gcc e altri strumenti.
3. **-y:**
    - L'opzione **-y** indica di accettare automaticamente tutte le conferme richieste durante l'installazione, rendendo il processo non interattivo.

Il comando “RUN apt install isc-dhcp-client isc-dhcp-server -y” viene utilizzato per installare i pacchetti isc-dhcp-client e isc-dhcp-server all’interno di un’immagine Docker:

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere l'installazione di pacchetti, configurazioni di sistema, copia di file, ecc.
2. **apt install isc-dhcp-client isc-dhcp-server:**
    - Questa parte del comando utilizza il sistema di gestione dei pacchetti di Debian per installare due pacchetti:
        - **isc-dhcp-client:** È un client DHCP (Dynamic Host Configuration Protocol) che consente a un dispositivo di ottenere dinamicamente la configurazione di rete, inclusi indirizzo IP, gateway e server DNS, da un server DHCP.
        - **isc-dhcp-server:** È un server DHCP che fornisce configurazioni di rete dinamiche agli altri dispositivi della rete che richiedono tali informazioni.
3. **-y:**
    - L'opzione **-y** indica di accettare automaticamente tutte le conferme richieste durante l'installazione, rendendo il processo non interattivo.

Il comando “RUN apt install lshw ethtool vlan kmod -y” viene utilizzato per installare i pacchetti lshw, ethtool, vlan e kmod all’interno di un’immagine Docker:

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere l'installazione di pacchetti, configurazioni di sistema, copia di file, ecc.
2. **apt install lshw ethtool vlan kmod:**
    - Questa parte del comando utilizza il sistema di gestione dei pacchetti di Debian per installare quattro pacchetti:
        - **lshw:** Fornisce informazioni dettagliate sull'hardware del sistema.
        - **ethtool:** È uno strumento che consente di visualizzare e modificare le impostazioni di una scheda di rete.
        - **vlan:** È utilizzato per la configurazione delle VLAN (Virtual LAN), che consentono la suddivisione di una rete fisica in più reti virtuali.
        - **kmod:** È il sistema di gestione dei moduli del kernel di Linux, utile per installare e rimuovere moduli del kernel.
3. **-y:**
    - L'opzione **-y** indica di accettare automaticamente tutte le conferme richieste durante l'installazione, rendendo il processo non interattivo.

Il comando “RUN useradd -rm -d /home/alessandro -s /bin/bash -g root -G sudo alessandro” viene utilizzato per aggiungere un nuovo utente denominato “alessandro” all’interno dell’immagine Docker:

1. **RUN:**

- L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere la configurazione di utenti, l'installazione di pacchetti, configurazioni di sistema, ecc.

2. **useradd -rm -d /home/alessandro -s /bin/bash -g root -G sudo alessandro:**
    - Questa parte del comando utilizza il comando **useradd** per aggiungere un nuovo utente chiamato "alessandro" all'interno dell'immagine Docker con le seguenti opzioni:
        - **-rm:** Rimuove la home directory predefinita (**/home/alessandro**) dell'utente se esistente. Questo aiuta a garantire che la home directory sia pulita o vuota quando l'utente viene creato.
        - **-d /home/alessandro:** Specifica il percorso della home directory dell'utente. In questo caso, è impostato su **/home/alessandro**.
        - **-s /bin/bash:** Specifica il shell predefinito dell'utente. Qui è impostato su Bash (**/bin/bash**).
        - **-g root:** Specifica il gruppo principale dell'utente, che è impostato su "root".
        - **-G sudo:** Aggiunge l'utente al gruppo "sudo". Gli utenti nel gruppo "sudo" hanno privilegi di amministratore e possono eseguire comandi con privilegi elevati utilizzando **sudo**.
        - **alessandro:** È il nome dell'utente che si sta aggiungendo.

Il comando “RUN echo 'alessandro::*********' | chpasswd” viene utilizzato per impostare la password dell’utente all’interno dell’immagine Docker (la password per motivi di sicurezza viene oscurata):

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere la configurazione di utenti, l'installazione di pacchetti, configurazioni di sistema, ecc.
2. **echo 'alessandro::**********' | chpasswd:**
    - Questa parte del comando utilizza la combinazione di comandi **echo** e **chpasswd** per impostare la password dell'utente "alessandro". La stringa **'alessandro::*********'** è una rappresentazione cifrata della password. Questa stringa segue il formato **username:password_hash**, dove **password_hash** è la password cifrata.
    - L'utilizzo di una password cifrata in questo modo è una pratica comune nei Dockerfile per garantire che le password non vengano esposte in chiaro nei file di configurazione.
    - Nella stringa fornita, la password è vuota (**::**), e ********** è un hash di password specifico o un segnaposto. Nella pratica, dovresti generare una password hash sicura in modo appropriato, ad esempio utilizzando lo strumento **openssl passwd**.

Il comando “RUN rmmod pcspkr ; echo "blacklist pcspkr" >>/etc/modprobe.d/blacklist.conf” esegue due operazioni relative ai moduli del kernel Linux:

1. **rmmod pcspkr:**

- **rmmod** è un comando di sistema in Linux utilizzato per rimuovere un modulo del kernel in modo dinamico. Nel comando fornito, **pcspkr** è il nome del modulo del kernel che viene rimosso. Il modulo **pcspkr** è associato al speaker PC (beeper) e la sua rimozione potrebbe disabilitare il suono del beep del sistema.

2. **;:**
    - Il punto e virgola **;** è un separatore di comandi in una singola istruzione **RUN**. Permette di eseguire più comandi consecutivamente.
3. **echo "blacklist pcspkr" >>/etc/modprobe.d/blacklist.conf:**
    - Questa parte del comando aggiunge la stringa "blacklist pcspkr" alla fine del file **/etc/modprobe.d/blacklist.conf**. Il file **blacklist.conf** viene utilizzato per specificare i moduli del kernel che non devono essere caricati automaticamente durante l'avvio del sistema. La voce "blacklist pcspkr" indica al sistema di non caricare automaticamente il modulo **pcspkr**.

Il comando “RUN service ssh start” viene utilizzato per avviare il servizio SSH all’interno di un’immagine Docker durante la fase di costruzione:

1. **RUN:**
    - L'istruzione **RUN** è utilizzata in un Dockerfile per eseguire comandi durante la creazione dell'immagine Docker. Questo può includere l'installazione di pacchetti, configurazioni di sistema, copia di file, ecc.
2. **service ssh start:**
    - Questa parte del comando avvia il servizio SSH all'interno del sistema operativo dell'immagine Docker. Il comando **service** è spesso utilizzato in sistemi basati su init o Systemd per gestire i servizi.
    - Nell'ambito di questo comando, **ssh** è il nome del servizio che si sta avviando, e **start** è l'azione richiesta, che indica di avviare il servizio.
    - L'avvio del servizio SSH all'interno dell'immagine Docker può essere utile se si desidera abilitare l'accesso SSH al container durante la fase di sviluppo o per altre configurazioni specifiche.

Il comando “CMD ["/usr/sbin/sshd","-D"]” definisce il comando di default che viene eseguito quando si avvia un container basato sull’immagine Docker:

1. **CMD:**
    - L'istruzione **CMD** è utilizzata in un Dockerfile per specificare il comando di default che deve essere eseguito quando un container basato sull'immagine viene avviato.
2. **["/usr/sbin/sshd", "-D"]:**
    - Questa parte del comando specifica l'eseguibile e gli eventuali argomenti che devono essere passati al momento dell'avvio del container.
    - **/usr/sbin/sshd**: Indica l'eseguibile del server SSH. Questo è il processo principale del server SSH che gestisce le connessioni SSH al container.
    - **-D**: Specifica l'opzione di esecuzione in modalità daemon. In questo modo, il server SSH viene eseguito in primo piano (foreground) anziché in modalità di

background. Questo è utile quando si esegue un container in modo interattivo o quando si desidera vedere direttamente l'output del server SSH nei log del container.

Vengono installati tutti i software aggiuntivi necessari per il corretto funzionamento del sistema. In particolare aggiunto il kmod per la gestione dei moduli del kernel e vlan per la gestione delle vlan.

kmod è un sistema di gestione dei moduli del kernel in ambienti Linux. Un modulo del kernel è una porzione di codice che può essere caricata dinamicamente nel kernel del sistema operativo per estendere le funzionalità del kernel o aggiungere supporto per nuovi dispositivi hardware. Il modulo del kernel può essere caricato o scaricato in esecuzione senza dover riavviare il sistema, offrendo una maggiore flessibilità e dinamicità.

2.2 Configurazione delle vlans nel router Mikrotik

Le VLAN (Virtual Local Area Network) sono una tecnologia di rete che consente di segmentare una rete fisica in più reti logiche isolate tra loro. Questo permette a dispositivi fisicamente collegati alla stessa rete di essere separati in gruppi distinti, migliorando la gestione del traffico, la sicurezza e l'efficienza della rete. Le VLAN sono configurate su switch di rete, dove ciascuna VLAN è identificata da un ID univoco. Attraverso l'uso di VLAN, è possibile controllare il traffico di rete, ridurre la congestione e limitare la diffusione dei dati a specifici segmenti della rete, aumentando la sicurezza isolando diversi gruppi di dispositivi, come dipartimenti aziendali o tipi di traffico (ad esempio, dati aziendali e voce).

Nel dispositivo MikroTik, è stata creata una VLAN identificata con l'ID 20 per gestire la rete LAN interna, consentendo di separare e organizzare il traffico di rete locale in modo efficiente e sicuro. La VLAN 20 isola i dispositivi della LAN, migliorando la sicurezza e la gestione del traffico. Per quanto riguarda la comunicazione sulla WAN, questa avviene direttamente sulla VLAN predefinita, la VLAN 1, che viene utilizzata per il traffico esterno. Questa configurazione permette di mantenere separati i traffici della rete interna (LAN) e della rete esterna (WAN), assicurando che il traffico interno rimanga isolato e gestito attraverso la VLAN 20, mentre il traffico verso e dalla WAN utilizza la VLAN 1, facilitando la gestione della rete e migliorando la sicurezza complessiva.

Il bridge nel dispositivo MikroTik è configurato con VLAN-Filtering, PVID 1 e l'opzione "admit all". Questo significa che il bridge è in grado di filtrare i pacchetti basandosi sulle VLAN, con la VLAN 1 impostata come VLAN predefinita (PVID). I frame appartenenti alla VLAN 1 sono considerati "nativi", il che significa che i frame in arrivo senza tag di VLAN vengono automaticamente associati alla VLAN 1. Inoltre, con l'opzione "admit all", il bridge è configurato per accettare tutti i tipi di frame in ingresso, sia taggati con un ID di VLAN specifico che non taggati. Questa configurazione permette una grande flessibilità, poiché consente di gestire e instradare correttamente il traffico sia delle VLAN configurate che del traffico non VLAN, assicurando al contempo che i dispositivi non configurati per utilizzare VLAN specifiche possano comunicare senza problemi all'interno della rete.

L'interfaccia eth2 del dispositivo MikroTik è configurata per il lato WAN con PVID 1 e l'opzione "admit all", il che significa che accetta sia frame taggati che non taggati, associando automaticamente i frame non taggati alla VLAN 1. Questa configurazione facilita l'accesso alla VLAN 1. Tuttavia, per il test da effettuare, si deve modificare l'impostazione dell'interfaccia eth2 a PVID 1 con l'opzione "admit only untagged" e abilitare l'ingress filtering. Questo cambiamento fa sì che l'interfaccia accetti solo frame non taggati, rifiutando quelli taggati, e applica il filtraggio all'ingresso per migliorare la sicurezza e l'efficienza del traffico di rete. Il test garantirà che solo il traffico non taggato, che viene automaticamente associato alla VLAN 1, sia accettato dall'interfaccia eth2, mentre tutto il traffico taggato sarà filtrato e bloccato, verificando così la corretta configurazione e isolamento del traffico WAN.

L'interfaccia eth8 del dispositivo MikroTik è configurata per il lato LAN con PVID 20, impostata come porta di accesso con le opzioni "admit only untagged" e "ingress filtering". Questa configurazione significa che l'interfaccia eth8 accetta solo frame non taggati, associandoli automaticamente alla VLAN 20 grazie al PVID 20. Inoltre, con l'opzione "admit only untagged", l'interfaccia rifiuta tutti i frame taggati con un ID VLAN, garantendo che solo il traffico destinato alla VLAN 20 entri tramite questa porta. L'abilitazione dell'ingress filtering

aggiunge un ulteriore livello di controllo, assicurando che solo i frame non taggati siano accettati e che il traffico non conforme sia filtrato all'ingresso.

Il container può utilizzare una sola interfaccia di rete (chiamata veth0). Per gestire sia il traffico LAN che quello WAN sulla stessa interfaccia, è stato necessario ricorrere alle VLAN (Virtual LAN), che permettono di separare e identificare distintamente i due tipi di traffico sulla medesima interfaccia fisica.

L'interfaccia virtuale veth2 è configurata come trunk, che permette il trasporto di traffico appartenente a diverse VLAN. La configurazione di veth2 include PVID 1 come VLAN nativa, l'opzione "admit all", e l'ingress filtering. Con PVID 1 come VLAN nativa, i frame non taggati che entrano nell'interfaccia vengono automaticamente associati alla VLAN 1. L'opzione "admit all" consente a veth2 di accettare sia frame taggati che non taggati, permettendo il passaggio di traffico di diverse VLAN attraverso questa interfaccia. L'ingress filtering è abilitato per garantire che solo il traffico corretto entri nell'interfaccia, migliorando la sicurezza e l'efficienza della rete.

La configurazione delle VLAN nel dispositivo MikroTik utilizza la VLAN 20 per la rete LAN, con l'interfaccia fisica ether8 configurata come porta di accesso untagged e l'interfaccia virtuale veth2 configurata come trunk tagged. L'interfaccia ether8 accetta solo frame non taggati, associandoli automaticamente alla VLAN 20, permettendo ai dispositivi connessi di comunicare senza necessità di configurazioni VLAN specifiche. D'altra parte, veth2, configurata come trunk con PVID 1 e opzione "admit all", trasporta sia frame taggati che non taggati, gestendo il traffico della VLAN 20 in forma taggata verso un container o un ambiente virtualizzato.

La configurazione della VLAN 1 nel dispositivo MikroTik è destinata alla rete WAN, con le interfacce bridge, veth2, e ether2 impostate come untagged. Questo significa che il traffico non taggato su queste interfacce viene automaticamente associato alla VLAN 1. Il bridge aggrega il traffico delle varie interfacce, facilitando la gestione unificata del traffico. L'interfaccia virtuale veth2 permette di estendere la connettività della VLAN 1 ad ambienti virtualizzati o containerizzati, mentre l'interfaccia fisica ether2 fornisce la connettività diretta alla WAN

L'interfaccia di livello 3 denominata WAN_PORT_MGMT è configurata per la gestione del traffico di rete con l'indirizzo IP 192.168.188.159/24. Questa configurazione specifica include l'abilitazione del protocollo ARP (Address Resolution Protocol), che consente la risoluzione degli indirizzi IP in indirizzi MAC, essenziale per la comunicazione in rete. Inoltre, l'interfaccia è associata alla VLAN ID 1, il che significa che gestisce il traffico appartenente a questa VLAN. Infine, WAN_PORT_MGMT è configurata sulla interface bridge1, che aggrega più interfacce fisiche e virtuali, permettendo una gestione centralizzata e facilitando il routing e la trasmissione del traffico di rete attraverso la VLAN 1.

L'interfaccia di livello 3 denominata LAN_PORT_V20 è configurata per la gestione del traffico di rete nella LAN con l'indirizzo IP 192.168.27.1/24. Questa configurazione include l'abilitazione del protocollo ARP (Address Resolution Protocol), che consente la risoluzione degli indirizzi IP in indirizzi MAC, essenziale per la comunicazione in rete. L'interfaccia è associata alla VLAN ID 20, indicando che gestisce il traffico appartenente a questa VLAN. È configurata sulla interface bridge1, che aggrega più interfacce fisiche e virtuali. È cruciale che

LAN_PORT_V20 sia in ascolto su bridge1, poiché altrimenti non sarebbe in grado di funzionare correttamente.

Il sistema Linux dispone di due interfacce di rete configurate per gestire la comunicazione attraverso VLAN utilizzando il comando ip. La configurazione prevede i seguenti passaggi:

1)Creazione della VLAN 20 su eth0:

ip link add link eth0 name eth0.20 type vlan id 20

Questo comando crea un'interfaccia VLAN denominata eth0.20 collegata all'interfaccia fisica eth0, specificando l'ID della VLAN 20.

2)Assegnazione degli indirizzi IP:

ip addr add 192.168.188.160/24 dev eth0

ip addr add 192.168.27.2/24 dev eth0.20

Il primo comando assegna l'indirizzo IP 192.168.188.160/24 all'interfaccia eth0, utilizzata per la comunicazione sulla rete principale. Il secondo comando assegna l'indirizzo IP 192.168.27.2/24 all'interfaccia VLAN eth0.20, utilizzata per la comunicazione sulla VLAN 20.

3)Attivazione dell'interfaccia VLAN:

ip link set eth0.20 up

Questo comando attiva l'interfaccia eth0.20, rendendola operativa per il traffico di rete.

4)Impostazione del gateway predefinito:

ip route add default via 192.168.188.1

Questo comando imposta il gateway predefinito del sistema a 192.168.188.1, che sarà utilizzato per instradare il traffico verso altre reti.

Collegando un calcolatore alla porta ether8, che è configurata per la VLAN 20, il calcolatore ottiene un indirizzo IP dalla rete 192.168.27.0/24. Questo dimostra che la configurazione delle interfacce di rete sul sistema Linux è corretta e funziona come previsto, permettendo al calcolatore di comunicare attraverso la VLAN 20 e ottenere un indirizzo IP appropriato. La configurazione garantisce che il traffico sia correttamente instradato attraverso le reti specificate, utilizzando l'interfaccia VLAN per la separazione del traffico e l'indirizzo IP corretto per la comunicazione all'interno della VLAN.

2.3 Criticità di configurazione con Mikrotik

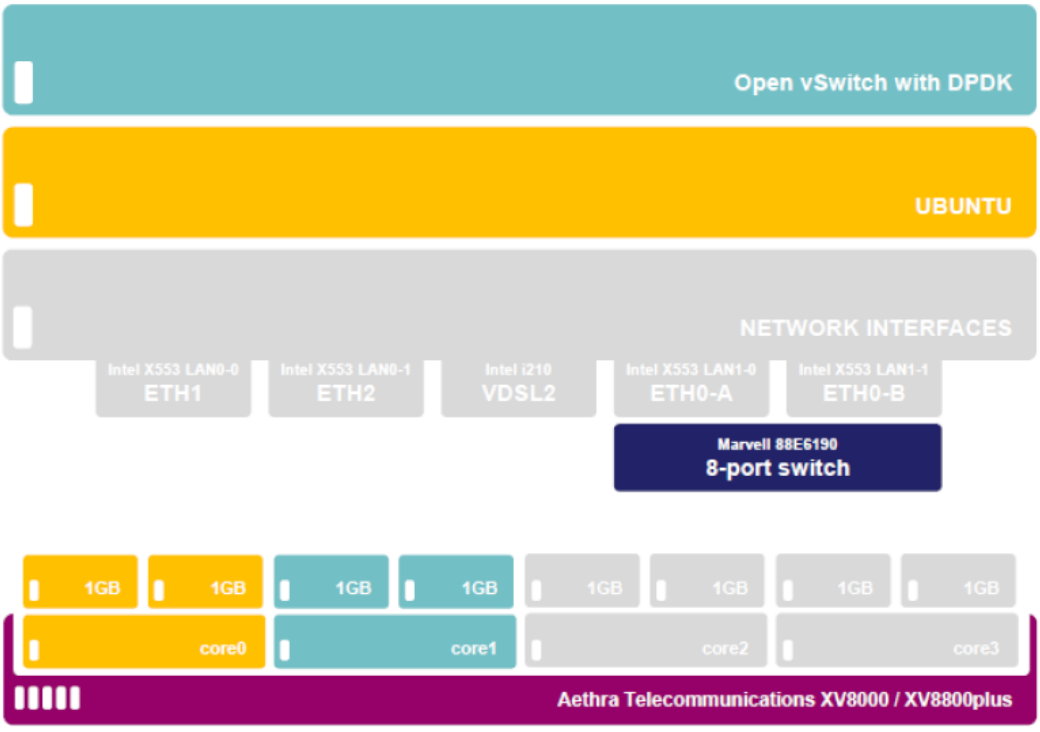
La problematica riguarda il container: a causa delle politiche di sviluppo software adottate da Mikrotik, non si hanno privilegi sufficienti per implementare il NAT all'interno di un container Linux. Questo impedisce di confrontare le prestazioni tra un router fisico e uno virtuale.
Un'altra criticità è che al container Mikrotik può essere associata solo una singola interfaccia di rete, anziché più di una.

Capitolo 3

Configurazione su dispositivo XV8800 Aethra Telecommunications

Nella documentazione Aethra Telecommunications, viene riportato un POC in cui l'immagine software di un router Huawei viene eseguita nella piattaforma di virtualizzazione di un router XV8800. I motivi di tale test sono diversificare l'infrastruttura di rete e facilitare la migrazione delle configurazioni tra diversi dispositivi. Come ad esempio considerare un laboratorio con un router Huawei e proporre di migrare l'architettura di rete su un altro router. Il router Huawei, ha già una configurazione pronta e ottimizzata. Invece di ricreare questa configurazione da zero su un altro router, può semplicemente trasferita in una macchina virtuale all'interno del router Aethra. In questo modo, la configurazione continua a funzionare come previsto, ma su un'infrastruttura hardware diversa.

Nel mio lavoro di tesi, il router Huawei è stato sostituito da un router realizzato attraverso kernel Linux, distribuzione Ubuntu.



La figura descrive l'architettura a livelli astratti del sistema:

1. **Parte Fisica**: L'hardware del router formata dalle interfacce di rete ETH1, ETH2, VDSL2, ETH0-A, ETH0-B, uno switch a 8 porte e vari blocchi che rappresentano la memoria e i core della cpu.
2. **Layer di Ubuntu**: Il sistema operativo su cui gira la macchina virtuale.

3. **Open vSwitch (OVS)**: Una realizzazione virtuale di uno switch pronto per SDN (Software-Defined Networking).

È possibile dedicare specifici core e slot di memoria ai singoli servizi. In particolare, si può decidere quanti core allocare alla funzione di switching e quanti core e RAM destinare alla macchina virtuale. Un core è sempre riservato al sistema come supervisor, mentre i core rimanenti possono essere allocati in modo dinamico o manuale. Ogni core, nel contesto delle funzioni di rete, si riferisce al networking generale e non a funzioni specifiche come il NAT. Per la macchina virtuale, ad esempio, è possibile allocare i core 5 e 6 per la virtualizzazione, permettendo una gestione più mirata delle risorse.

```
XV8800 with 8 CPU: Assignment of cores to the accelerator

set dpa on
set dpa cores 1 2 3
```

La figura mostra il comando per assegnare i core alla cpu:

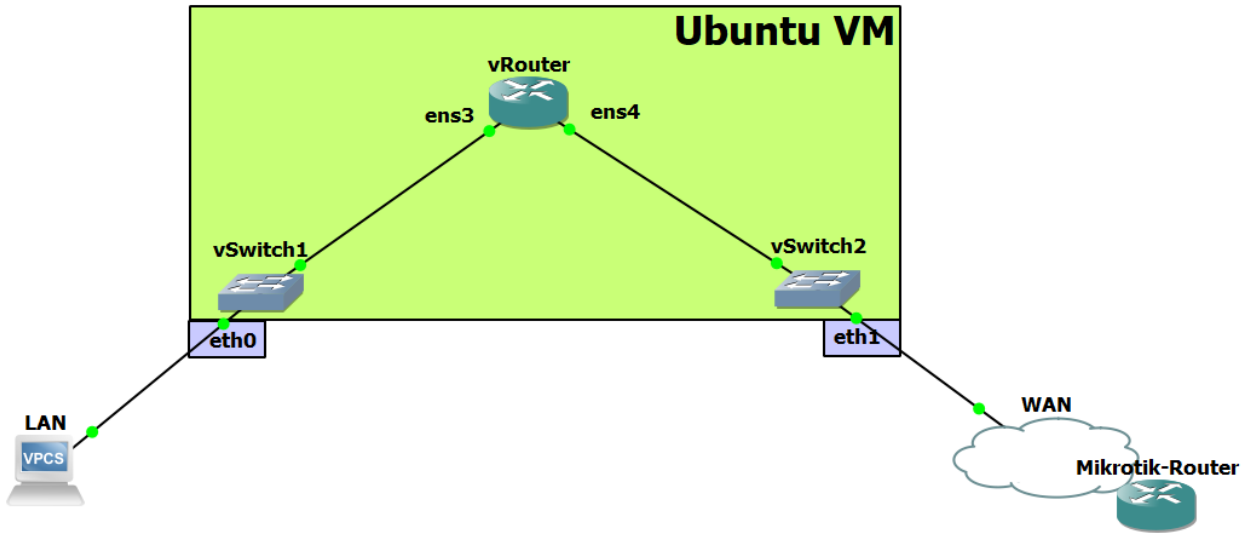
Set dpa on e set dpa core 1 2 3 sono i due comandi per assegnare 3 core all'accelerazione hardware dei flussi di rete.

```
ATOSNT>>show system statistics

Show statistics of ATOSNT system
Dynamic memory usage (KBytes)
    Total   RamDisk  RamDisk(%)      Free Free(%)
 16514780    757344         4.6   8685200    52.6

Average CPU usage in last 2 seconds
   CPU       Assignment Idle(%) User(%) Kernel(%) Waste(%) HWIrq(%) SWIrq(%) IRQ/sec
  CPUs                -    62.2    37.8       0.0       0.0       0.0       0.0  1446.0
  CPU0              SYS    98.0     2.0       0.0       0.0       0.0       0.0   147.0    <- "1"
  CPU1              DPA     0.0   100.0       0.0       0.0       0.0       0.0   100.0    <- "2"
  CPU2              DPA     0.0   100.0       0.0       0.0       0.0       0.0   100.0    <- "3"
  CPU3              DPA     0.0   100.0       0.0       0.0       0.0       0.0   100.0    <- "4"
  CPU4              SYS   100.0     0.0       0.0       0.0       0.0       0.0   325.5
  CPU5              SYS   100.0     0.0       0.0       0.0       0.0       0.0   323.5
  CPU6              SYS   100.0     0.0       0.0       0.0       0.0       0.0   141.5
  CPU7              SYS   100.0     0.0       0.0       0.0       0.0       0.0   208.5
```

Tramite l'uso del comando show system statistic mostrato in figura, vengono mostrate le statistiche del sistema, inclusi l'utilizzo della memoria e la media di utilizzo della CPU, considerando l'esempio e il comando precedente si può osservare che effettivamente le cpu 1,2,3 sono state allocate.

3.1 Topologia di rete

La figura rappresenta la topologia creata per lo sviluppo di questa tesi sperimentale.



In questo scenario, è stato scelto di utilizzare un router Aethra all'interno del quale è stata installata una macchina virtuale con sistema operativo Ubuntu. Sono state configurate due interfacce, ens3 ed ens4. Queste interfacce, pur essendo logiche dal punto di vista dell'utente, sono trattate come fisiche dal sistema operativo. Poiché si tratta di un contesto di full virtualization, il sistema operativo vede queste interfacce come fisiche e le nomina di conseguenza, utilizzando il prefisso "ens", come farebbe per le interfacce fisiche reali. Formalmente, queste interfacce sono logiche poiché non corrispondono ad alcun hardware fisico, ma per il sistema operativo appaiono come interfacce fisiche poiché sono le uniche visibili. Le interfacce ens3 ed ens4 sono state configurate rispettivamente indirizzo ip 192.168.1.10/24 e indirizzo ip acquisito tramite dhcp, sono connesse due vSwitch. Il vSwitch (virtual switch) è un componente fondamentale che deve essere abilitato per poter connettere le interfacce di rete logiche e fisiche. Una volta abilitato, il vSwitch permette alle macchine virtuali di comunicare sia con l'esterno (ad esempio, altre reti o Internet) sia con l'interno (come altre macchine virtuali all'interno della stessa LAN). In pratica, il vSwitch funziona come uno switch fisico, gestendo il traffico di rete tra le interfacce collegate, ma opera a livello virtuale all'interno dell'infrastruttura virtualizzata. Questa configurazione è essenziale per garantire la connettività e la comunicazione efficace delle macchine virtuali. Le interfacce esn3 ed esn4 sono collegate ai due vswitches rispettivamente tramite le interfacce ublan e ubwan. Il vswitch denominato vswlan è collegato alla porta ethernet 0 del router Aethra tramite ip statico 192.168.1.254/24 mentre il vswitch denominato vswwan è collegato alla porta ethernet 1 del router Aethra. A loro volta l’interfaccia ethernet 0 è collegata a un calcolatore che funge da host mentre l’interfaccia ethernet 1 è collegata al router Mikrotik di frontiera.

3.2 Configurazione topologia

add interfaces IFC eth0 eth0

add interfaces IFC eth1 eth1

Questi due comandi permettono di configurare le interfacce del sistema operativo. L'interfaccia eth0 lato Lan e l'interfaccia eth1 lato Wan

add vswitches VSWITCH vswlan

add vswitches VSWITCH vswwan

Questi comandi permettono la creazione del vSwitch tramite "add" e l'abilitazione tramite "set".

add vswitches vswlan PORT eth0

add vswitches vswwan PORT eth1

Questi comandi permettono l'aggiunta delle porte fisiche agli switch virtuali.

add interface ifc tap ublan

add interface ifc tap ubwan

add vswitch vswlan port ublan

add vswitch vswwan port ubwan

Per creare le porte logiche su cui verrà collegat9a la macchina virtuale, è necessario aggiungere interfacce IFC di tipo tap. Le interfacce tap operano a livello 2 del modello OSI e consentono la creazione di bridge, facilitando la connessione tra le porte logiche e la macchina virtuale. Questo processo permette alla macchina virtuale di interagire con la rete come se fosse collegata a interfacce fisiche, garantendo una comunicazione fluida e efficiente all'interno della topologia di rete.

add vswitches vswlan FLOW eth0 tap-ublan

add vswitches vswwlan FLOW eth1 tap-ubwan

Il flusso di traffico viene impostato utilizzando i comandi illustrati precedentemente. Questi comandi garantiscono che tutto il traffico in ingresso nell'interfaccia eth1 venga inoltrato a tap-vdev1 e viceversa. Queste interfacce sono collegate tra loro a livello di NFV (Network Functions Virtualization) tramite il flusso configurato nel vSwitch.

Il flusso permette di stabilire un circuito virtuale all'interno dello switch, forzando il percorso del traffico. Ad esempio, considerando un vSwitch senza CPU, tutti i frame in arrivo vengono inviati a un controller che applica regole per assicurare che ogni frame con un determinato indirizzo MAC di destinazione (DMAC) venga inoltrato a una specifica porta. Questo processo

viene ripetuto per ogni frame in ingresso. I flussi possono operare a livello 3, permettendo il matching anche a livello IP.

In questo modo, la distinzione tra switch e router diventa irrilevante: utilizzando un flusso di livello 2, il dispositivo opera come uno switch, mentre con un flusso di livello 3, opera come un router. L'hardware utilizzato non è rilevante per la funzionalità svolta.

Un ulteriore esempio riguarda le tabelle ARP, gestite dal controller, eliminando la necessità di tabelle ARP tradizionali. Lo switch diventa così un insieme di transceiver per la commutazione ad alta velocità dei pacchetti, trasformandosi da dispositivo intelligente a semplice matrice di commutazione. In questo scenario, la "mente" risiede nel controller, mentre lo switch funge solo da "braccio".

In data center, ad esempio, gli switch Huawei di dimensioni considerevoli, con lame orizzontali per il collegamento dei cavi, non sono intelligenti. Questi dispositivi, noti come SFU (Switch Fabric Unit), sono semplici commutatori senza intelligenza intrinseca, poiché l'intelligenza risiede nei due controller. Se uno dei controller si guasta, l'altro prende il suo posto, garantendo la continuità operativa dello switch. Inoltre, il problema dei loop di STP (Spanning Tree Protocol) viene gestito direttamente dal controller, semplificando ulteriormente la rete.

3.3 NAT

Per assicurarsi che la topologia di rete funzioni correttamente e che i pacchetti raggiungano la loro destinazione, viene configurato il NAT (Network Address Translation) tra la rete LAN e la rete WAN. Questa configurazione consente ai dispositivi sulla rete LAN di comunicare con dispositivi sulla rete WAN, traducendo gli indirizzi IP interni in indirizzi IP pubblici e viceversa, garantendo così un corretto instradamento e consegna dei pacchetti.

Network Address Translation (NAT) è una tecnologia utilizzata nelle reti informatiche per consentire a più dispositivi di una rete locale di condividere un singolo indirizzo IP pubblico. Questa tecnologia è utile in contesti dove gli indirizzi IP pubblici sono limitati o costosi. Il funzionamento del Nat si può riassumere in questi passaggi:

1. **Indirizzi Privati e Pubblici**: In una rete locale (LAN), ogni dispositivo ha un indirizzo IP privato unico. Però, per comunicare con l'esterno (Internet), questi dispositivi devono utilizzare un indirizzo IP pubblico.
2. **Traduzione degli Indirizzi**: Quando un dispositivo della rete locale invia un pacchetto dati verso Internet, il router che implementa il NAT modifica l'indirizzo IP sorgente del pacchetto, sostituendolo con l'indirizzo IP pubblico del router stesso. Inoltre, il router tiene traccia della connessione in una tabella di NAT, registrando l'indirizzo IP privato e la porta di origine.
3. **Ritorno dei Pacchetti**: Quando una risposta arriva dall'esterno, il router utilizza la tabella di NAT per tradurre l'indirizzo IP di destinazione pubblico e la porta di ritorno all'indirizzo IP privato e alla porta originale del dispositivo interno. Il pacchetto viene quindi inviato al dispositivo appropriato nella rete locale.

Esistono diversi tipi di NAT:

1. **NAT Dinamico**: Mappa automaticamente un indirizzo IP privato a uno pubblico disponibile in modo dinamico. Questo tipo di NAT è utile quando si hanno più dispositivi che accedono a Internet contemporaneamente.
2. **NAT Statico**: Mappa un indirizzo IP privato a uno specifico indirizzo IP pubblico. È utilizzato quando si desidera che un dispositivo interno sia sempre raggiungibile da un indirizzo IP pubblico specifico.
3. **PAT (Port Address Translation)**: Conosciuto anche come "NAT sovraccarico", mappa più indirizzi IP privati a un singolo indirizzo IP pubblico utilizzando porte differenti. Questo è il tipo di NAT più comune nei router domestici.

Il NAT offre diversi vantaggi come

1. **Conservazione degli Indirizzi IP**: Riduce il numero di indirizzi IP pubblici necessari, permettendo a più dispositivi di condividere un singolo indirizzo IP pubblico.
2. **Sicurezza**: Aggiunge un livello di sicurezza nascondendo gli indirizzi IP privati dalla rete esterna, rendendo più difficile per gli attaccanti identificare e attaccare dispositivi specifici all'interno della rete locale.
3. **Flessibilità di Rete**: Facilita la riorganizzazione e la gestione degli indirizzi IP all'interno di una rete locale senza dover cambiare la configurazione degli indirizzi IP pubblici.

Mentre gli svantaggi sono:

1. **Compatibilità con Applicazioni**: Alcune applicazioni, specialmente quelle che richiedono connessioni peer-to-peer, possono avere problemi con il NAT, poiché richiedono la conoscenza degli indirizzi IP e delle porte esatte per funzionare correttamente.
2. **Ritardi**: La traduzione degli indirizzi può introdurre piccoli ritardi nella trasmissione dei pacchetti dati.

La configurazione del Nat viene eseguita sul sistema operativo Ubuntu, dove l'interfaccia esn3 è collegata alla rete LAN e l'interfaccia esn4 è collegata alla rete WAN. Con questa configurazione, tutti i pacchetti che escono dall'interfaccia esn4 vengono "nattati" utilizzando la tecnica del masquerade. Questo processo modifica i pacchetti in modo che sembrino provenire dall'indirizzo IP dell'interfaccia esn4, permettendo una comunicazione fluida con la rete WAN e nascondendo gli indirizzi IP interni della rete LAN.

Tramite il comando " sudo iptables -t nat -A POSTROUTING -o esn4 -j MASQUERADE" i pacchetti in uscita da esn4 vengono "mascherati"

Tramite i comandi "sudo iptables -A FORWARD -i esn3 -o esn4 -j ACCEPT" e "sudo iptables -A FORWARD -i esn4 -o esn3 -m state --state RELATED,ESTABLISHED -j ACCEPT" vengono aggiunte le regole per il forwarding tra le interfacce esn3 ed esn4.

Tramite i comandi "sudo apt-get install iptables-persistent" e "sudo netfilter-persistent save" vengono salvate le regole per le distribuzioni basate su ubuntu.

Tramite il comando "iptables -t nat -L -n -v" vengono elencate le regole nella tabella NAT (-t nat) con opzioni per non risolvere i nomi (-n) e mostrare informazioni dettagliate (-v).

```
root@ubuntu2310:~# iptables -t nat -L -n -v
Chain PREROUTING (policy ACCEPT 2560 packets, 231K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain INPUT (policy ACCEPT 75 packets, 25187 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 29 packets, 1867 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 25 packets, 1585 bytes)
 pkts bytes target     prot opt in     out     source               destination
  112 17861 MASQUERADE  0    --  *      ens4    0.0.0.0/0            0.0.0.0/0
root@ubuntu2310:~# iptables -L FORWARD -n -v
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
 7174 5519K ACCEPT     0    --  *      *       0.0.0.0/0            192.168.1.0/24
 4893  943K ACCEPT     0    --  *      *       192.168.1.0/24       0.0.0.0/0
    0     0 ACCEPT     0    --  eth0   eth1    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 ACCEPT     0    --  esn3   esn4    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     0    --  esn4   esn3    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
```

In riferimento alla figura, considerando le catene di POSTROUTING e di FORWARD:

La catena POSTROUTING si occupa di manipolare i pacchetti dopo che il routing è stato determinato, ma prima che escano dall'interfaccia di rete. È comunemente utilizzata per il Source NAT (SNAT) e il masquerading. Vedendo la catena nel dettaglio:

- **Policy**: ACCEPT (Accetta tutti i pacchetti che non corrispondono a nessuna regola specifica).
- **Regole**:

-MASQUERADE: Applica il masquerading ai pacchetti in uscita sull'interfaccia ens4. Il masquerading è una forma di SNAT che modifica l'indirizzo IP di origine dei pacchetti con l'indirizzo IP pubblico dell'interfaccia di uscita.

-Condizioni: Tutti i pacchetti (all) che attraversano qualsiasi interfaccia di ingresso (*) e che escono dall'interfaccia ens4 (uscita), indipendentemente dalla loro sorgente o destinazione (0.0.0.0/0).

La catena FORWARD gestisce i pacchetti che devono essere inoltrati attraverso il router, cioè pacchetti che arrivano su un'interfaccia di rete e devono essere trasmessi su un'altra interfaccia senza essere processati localmente. Vedendo la catena nel dettaglio:

- **Policy**: ACCEPT (Accetta tutti i pacchetti che non corrispondono a nessuna regola specifica).
- **Regole**:

  - Regola 1:

  - **pkts/bytes**: 7174 pacchetti, 5519K byte.
  - **Descrizione**: Accetta tutti i pacchetti (all) provenienti da qualsiasi interfaccia di ingresso (*) e destinati all'interfaccia di uscita eth1, se la sorgente è la rete 192.168.1.0/24 e la destinazione è qualsiasi (0.0.0.0/0).

  -Regola 2:

  - **pkts/bytes**: 4893 pacchetti, 943K byte.
  - **Descrizione**: Accetta tutti i pacchetti (all) provenienti dall'interfaccia di ingresso eth0 e destinati all'interfaccia di uscita eth1, indipendentemente dalla sorgente e dalla destinazione, se lo stato della connessione è RELATED, ESTABLISHED.
  - **State**: RELATED, ESTABLISHED indica che il pacchetto fa parte di una connessione già stabilita o correlata a una connessione esistente.
- **Regola 3**:
  - **pkts/bytes**: 0 pacchetti, 0 byte.
  - **Descrizione**: Accetta tutti i pacchetti (all) provenienti dall'interfaccia di ingresso esn3 e destinati all'interfaccia di uscita ens4, indipendentemente dalla sorgente e dalla destinazione.
- **Regola 4**:
  - **pkts/bytes**: 0 pacchetti, 0 byte.
  - **Descrizione**: Accetta tutti i pacchetti (all) provenienti dall'interfaccia di ingresso ens4 e destinati all'interfaccia di uscita esn3, indipendentemente dalla sorgente e dalla destinazione.

Inoltre, nella catena di FORWARD sono state inserite due regole che consentono al router Ubuntu di inoltrare i pacchetti provenienti dalla rete 192.168.1.0/24 verso il resto del mondo e viceversa. Queste regole possono essere verificate con il comando iptables -L FORWARD -n -v. È importante notare che un sistema Linux si comporta come un router solo quando il bit ip_forward è impostato a 1 nel file /proc/sys/net/ipv4/ip_forward. L'attivazione di questo bit può essere verificata tramite il comando mostrato nella figura seguente:

```
root@ubuntu2310:~# cat /proc/sys/net/ipv4/ip_forward
1
```

Per far sì che il sistema Linux funzioni come un router, il contenuto del file ip_forward deve essere impostato a 1. Al contrario, se il file è impostato a 0, il sistema Linux si comporterà come un server e non inoltrerà i pacchetti da un'interfaccia all'altra. Nella topologia in esame, ciò significa che i pacchetti che entrano tramite l'interfaccia ens3 non verranno inoltrati a ens4 senza il routing abilitato. Impostando ip_forward a 1 e aggiungendo le regole appropriate nella catena FORWARD di iptables, il sistema inizierà a inoltrare i pacchetti correttamente. Di conseguenza, un pacchetto proveniente da un dispositivo con indirizzo IP 192.168.1.1 raggiungerà l'interfaccia 192.168.1.10, e il meccanismo di routing garantirà che i pacchetti siano instradati correttamente.

```
C:\Users\alese>ping 10.100.0.115

Esecuzione di Ping 10.100.0.115 con 32 byte di dati:
Risposta da 10.100.0.115: byte=32 durata=1ms TTL=64
Risposta da 10.100.0.115: byte=32 durata<1ms TTL=64

Statistiche Ping per 10.100.0.115:
    Pacchetti: Trasmessi = 2, Ricevuti = 2,
    Persi = 0 (0% persi),
Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 0ms, Massimo =  1ms, Medio =  0ms
```

Nella figura è mostrato un esempio del comando "ping" utilizzato per verificare la raggiungibilità dell'indirizzo 10.100.0.115 dell'interfaccia vswan.

Il processo di routing complessivo della topologia, partendo dal calcolatore che funge da host:

Quando il calcolatore con indirizzo IP 192.168.1.1 invia un ping all'indirizzo 10.100.0.1, il quale non fa parte del suo spazio degli indirizzi, la richiesta viene inoltrata al gateway predefinito, che è 192.168.1.10, l'indirizzo IP dell'interfaccia esn3 del sistema Ubuntu. Questa interfaccia riceve il frame contenente il pacchetto destinato a 10.100.0.1, lo decapsula e legge l'indirizzo IP di destinazione. Poiché l'indirizzo IP di destinazione è 10.100.0.1, il sistema, consultando la tabella di routing, determina che il pacchetto deve essere inoltrato all'interfaccia ens4.

Successivamente, iptables, operante nella catena di POSTROUTING, rileva che il pacchetto deve uscire dall'interfaccia esn4 e applica la tecnica del masquerading. Questo processo comporta la sostituzione dell'indirizzo IP sorgente del pacchetto con quello assegnato all'interfaccia esn4, che è dinamico. Di conseguenza, il masquerading riscrive l'indirizzo IP sorgente del pacchetto con il valore dell'IP assegnato all'interfaccia in quel momento.

La distinzione tra il masquerading e il Source NAT (SNAT) risiede nella natura dell'indirizzo IP: il SNAT è statico, mentre il masquerading viene utilizzato in scenari con indirizzi IP dinamici. Analogamente, nel contesto delle tecnologie Huawei, il masquerading è paragonabile a Easy IP, una soluzione di rete che semplifica la configurazione e la gestione delle funzioni di indirizzamento IP e NAT sui dispositivi di rete Huawei, come router e switch. Questo servizio è

progettato per facilitare l'accesso a Internet dei dispositivi nella rete locale senza la necessità di configurazioni complesse.

3.4 Configurazione Nat sul router Aethra

Viene configurato il Nat anche sul router Aethra attraverso i comandi:

ATOSNT>>conf

top

set system name ATOSNT

set system console-baudrate 115200

set dpa on

set dpa cores 1

Il primo gruppo di comandi configurano il nome del dispositivo, la velocità della console per la gestione, e attivano una funzionalità avanzata di accelerazione della rete per migliorare le prestazioni complessive del dispositivo come ad esempio il comando che abilita il Data Plane Acceleration che migliora le prestazioni di elaborazione dei pacchetti di rete.

add vswitches VSWITCH vswlan

add vswitches VSWITCH vswwan

set vswitches vswlan on

set vswitches vswwan on

Il secondo gruppo di comandi configurano due switch virtuali nel dispositivo di rete. Il comando add vswitches VSWITCH vswlan crea un virtual switch (vSwitch) denominato vswlan, e add vswitches VSWITCH vswwan crea un altro vSwitch denominato vswwan. Successivamente, i comandi set vswitches vswlan on e set vswitches vswwan on attivano rispettivamente i vSwitch vswlan e vswwan, rendendoli operativi per gestire il traffico di rete virtualizzato.

add interfaces IFC eth1 eth1

add interfaces IFC eth0 eth0

add interfaces IFC tap ublan

add interfaces IFC tap ubwan

add interfaces IFC vswlan vswlan

add interfaces IFC vswwan vswwan

Il terzo gruppo di comandi aggiungono diverse interfacce di rete al dispositivo. Le interfacce fisiche eth1 e eth0 vengono aggiunte con i comandi add interfaces IFC eth1 eth1 e add interfaces IFC eth0 eth0. Inoltre, vengono aggiunte due interfacce TAP, ublan e ubwan,

utilizzate per connessioni virtuali, con i comandi add interfaces IFC tap ublan e add interfaces IFC tap ubwan. Infine, vengono aggiunte due interfacce virtuali, vswlan e vswwan, con i comandi add interfaces IFC vswlan vswlan e add interfaces IFC vswwan vswwan, per gestire il traffico attraverso gli switch virtuali precedentemente creati.

add vswitches vswlan PORT eth0

add vswitches vswlan PORT ublan

add vswitches vswlan FLOW 10 normal

add vswitches vswwan PORT eth1

add vswitches vswwan PORT ubwan

add vswitches vswwan FLOW 10 normal

Il quarto gruppo di comandi configurano le porte e i flussi per i virtual switch (vSwitch) vswlan e vswwan. Il comando add vswitches vswlan PORT eth0 collega la porta fisica eth0 al vSwitch vswlan, e add vswitches vswlan PORT ublan collega l'interfaccia TAP ublan allo stesso vSwitch. Con add vswitches vswlan FLOW 10 normal, viene impostato un flusso normale con priorità 10 per vswlan. Allo stesso modo, add vswitches vswwan PORT eth1 collega la porta fisica eth1 al vSwitch vswwan, e add vswitches vswwan PORT ubwan collega l'interfaccia TAP ubwan a vswwan. Infine, add vswitches vswwan FLOW 10 normal imposta un flusso normale con priorità 10 per il vSwitch vswwan. Queste configurazioni permettono al traffico di rete di essere gestito e instradato correttamente attraverso i vSwitch e le rispettive porte collegate.

set interfaces vswlan ip address 192.168.1.254/24

set interfaces vswwan ip address 10.100.0.114/24

Il quinto gruppo di comandi configurano gli indirizzi IP per le interfacce virtuali vswlan e vswwan. Il comando set interfaces vswlan ip address 192.168.1.254/24 imposta l'indirizzo IP 192.168.1.254 con una maschera di sottorete 255.255.255.0 sull'interfaccia virtuale vswlan, configurandola per la rete locale. Allo stesso modo, set interfaces vswwan ip address 10.100.0.114/24 imposta l'indirizzo IP 10.100.0.114 con una maschera di sottorete 255.255.255.0 sull'interfaccia virtuale vswwan, configurandola per la rete WAN. Queste configurazioni permettono al dispositivo di comunicare correttamente sia nella rete locale che nella rete esterna.

add ip route 0.0.0.0 0.0.0.0 10.100.0.1 1

add napt IFC vswwan

set napt vswwan on

Il sesto gruppo di comandi configurano il routing e il Network Address Port Translation (NAPT) sul dispositivo. Il comando add ip route 0.0.0.0 0.0.0.0 10.100.0.1 1 aggiunge una rotta predefinita (default route) che instrada tutto il traffico non specificato attraverso il gateway 10.100.0.1 con priorità 1. Il comando add napt IFC vswwan abilita il NAPT sull'interfaccia

vswwan, permettendo la traduzione degli indirizzi e delle porte per i pacchetti che passano attraverso questa interfaccia. Infine, set napt vswwan on attiva effettivamente il NAPT sull'interfaccia vswwan, abilitando la funzionalità di traduzione per il traffico in uscita. Queste configurazioni consentono al dispositivo di instradare correttamente il traffico verso Internet e di gestire la traduzione degli indirizzi IP e delle porte per i dispositivi della rete locale.

add nfv VM ubuntu

add nfv ubuntu DRIVE drive0

add nfv ubuntu IFC ulan

add nfv ubuntu IFC uwan

set nfv ubuntu on

set nfv ubuntu vcpus-number 2

set nfv ubuntu startup-ram 4096

set nfv ubuntu cpu-pinning 2 3

set nfv ubuntu vnc-port 5901

set nfv ubuntu drive0 location e:virt_images7ubu2310.qcow2

set nfv ubuntu ulan attached-to ublan

set nfv ubuntu uwan attached-to ubwan

set nfv ubuntu uwan vm-mac-address 00-AA-BB-CC-00-01

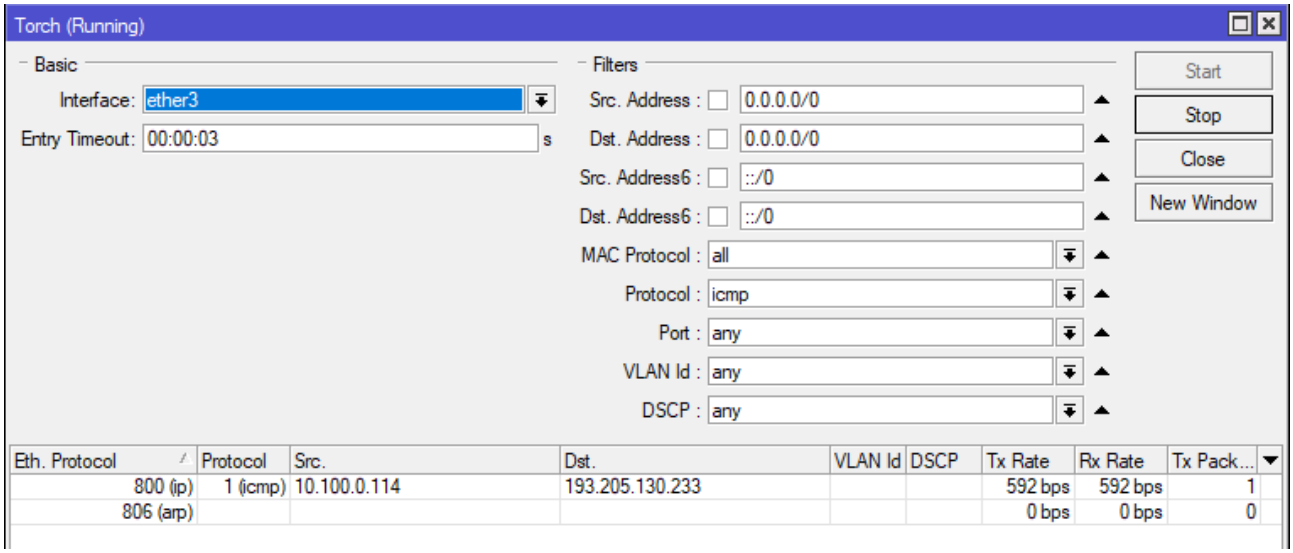
Il settimo gruppo di comandi configurano e avviano una macchina virtuale (VM) chiamata ubuntu all'interno del dispositivo di rete. Il comando add nfv VM ubuntu crea la VM ubuntu, mentre add nfv ubuntu DRIVE drive0 aggiunge un disco di archiviazione denominato drive0 alla VM. Le interfacce di rete ulan e uwan vengono aggiunte alla VM con i comandi add nfv ubuntu IFC ulan e add nfv ubuntu IFC uwan. La VM viene attivata con set nfv ubuntu on. Successivamente, set nfv ubuntu vcpus-number 2 assegna 2 CPU virtuali alla VM e set nfv ubuntu startup-ram 4096 assegna 4096 MB di RAM. Il comando set nfv ubuntu cpu-pinning 2 3 associa fisicamente i core 2 e 3 della CPU alla VM. Per l'accesso remoto, set nfv ubuntu vnc-port 5901 imposta la porta VNC a 5901. Il percorso dell'immagine del disco per la VM viene specificato con set nfv ubuntu drive0 location e:virt_images7ubu2310.qcow2. Le interfacce di rete vengono ulteriormente configurate con set nfv ubuntu ulan attached-to ublan e set nfv ubuntu uwan attached-to ubwan, collegandole rispettivamente alle reti ublan e ubwan. Infine, set nfv ubuntu uwan vm-mac-address 00-AA-BB-CC-00-01 imposta l'indirizzo MAC 00-AA-BB-CC-00-01 per l'interfaccia uwan. Queste configurazioni permettono alla VM ubuntu di operare con risorse hardware allocate, connettività di rete e accesso remoto configurato.

CAPITOLO 4

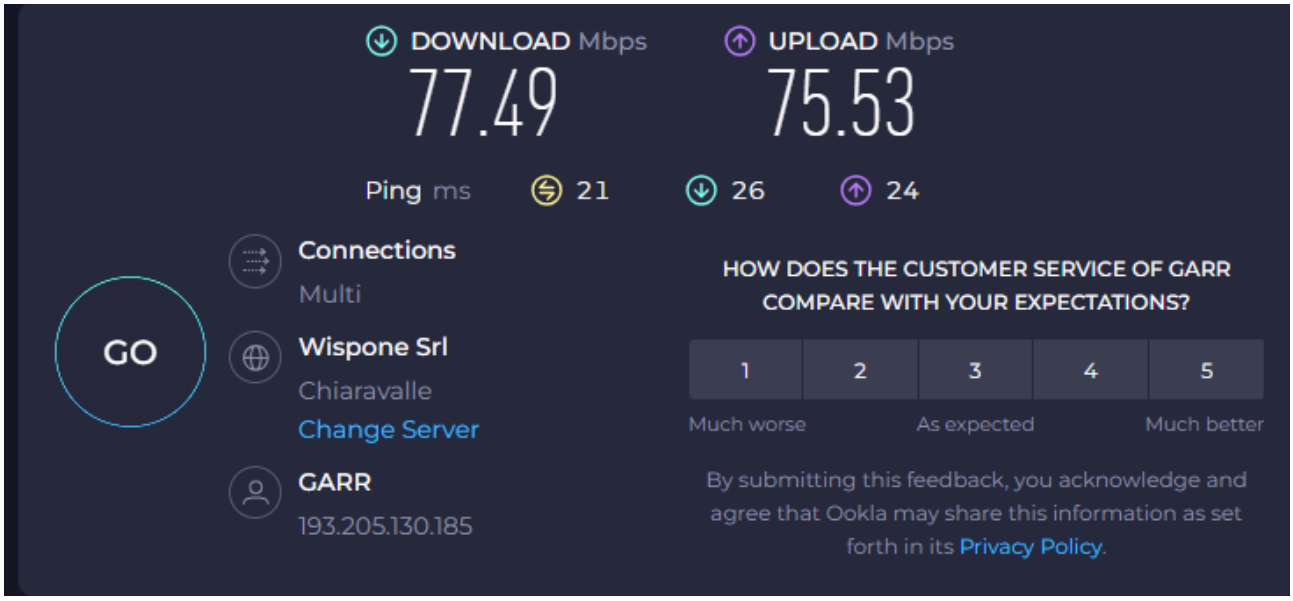
Risultati e conclusioni

Dopo aver configurato il NAT, il passo successivo è verificare la qualità delle prestazioni della rete, sia utilizzando il router fisico Aethra che la macchina virtuale Ubuntu.

Considerando l'implementazione del NAT sul router Aethra, dove l'indirizzo LAN 192.168.1.254 è associato alla VLAN vSwlan e l'indirizzo WAN 10.100.0.114 è associato alla VLAN vSwwan, e un computer con indirizzo IP 192.168.1.22 e gateway 192.168.1.254, è stato verificato il corretto instradamento dei pacchetti attraverso il router Mikrotik di frontiera. Questo processo assicura che i pacchetti dal computer siano correttamente indirizzati attraverso il router Mikrotik, passando per le interfacce configurate sul router Aethra.

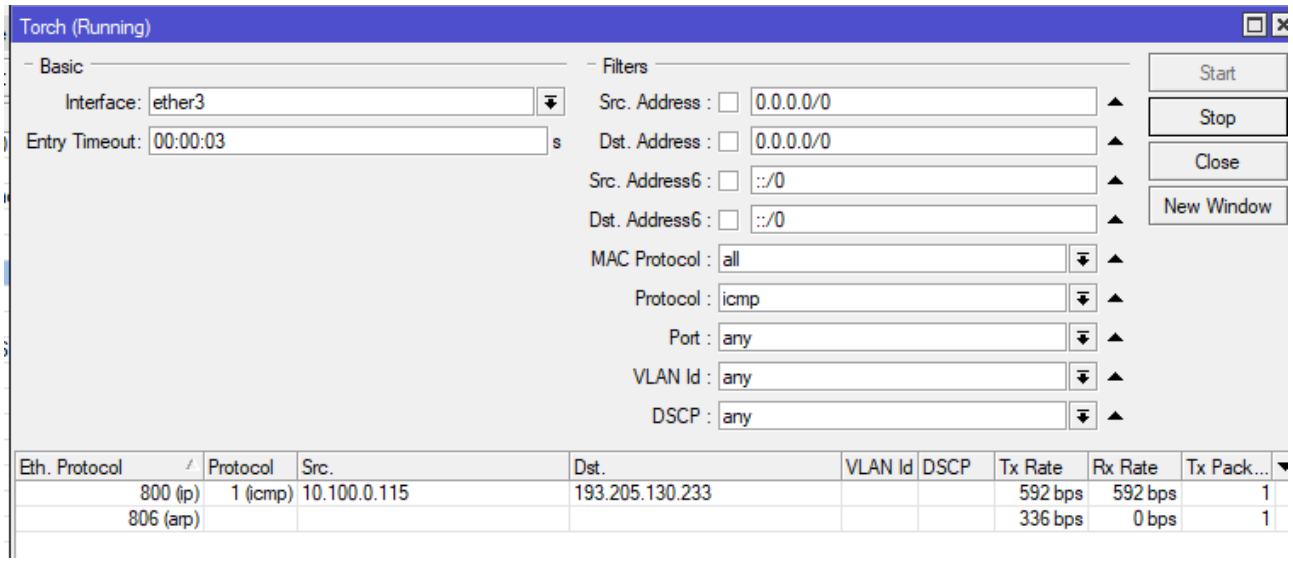


L'immagine mostra l'uso del tool Torch su un dispositivo MikroTik per monitorare il traffico di rete sull'interfaccia ether3. Torch è configurato per catturare il traffico ICMP, utilizzato per i ping, con un timeout di 3 secondi per le voci catturate. I filtri impostati permettono di catturare traffico da qualsiasi indirizzo IP di origine e destinazione, sia IPv4 che IPv6, senza restrizioni su MAC protocol, porte, VLAN ID o DSCP. L'analisi in tempo reale mostra due voci di traffico: una per il protocollo ICMP con indirizzo IP di origine 10.100.0.114 e destinazione 193.205.130.233, che trasmette e riceve a una velocità di 592 bps con un pacchetto trasmesso, e una per ARP, senza pacchetti attivi. Torch consente di avviare o fermare la cattura del traffico e aprire nuove finestre di monitoraggio, fornendo un efficace strumento per diagnosticare problemi di rete e monitorare il traffico in tempo reale su specifiche interfacce.

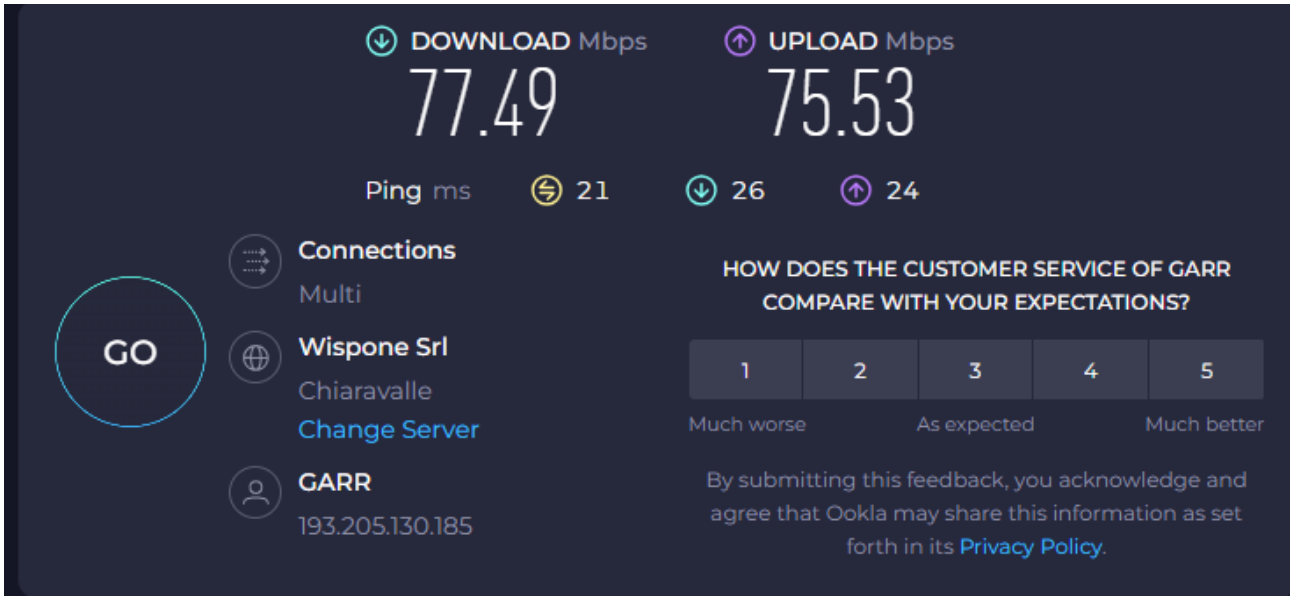


L'immagine mostra i risultati di un test di velocità della connessione Internet eseguito con Speedtest di Ookla. Il test rivela una velocità di download di 77.49 Mbps e una velocità di upload di 75.53 Mbps, entrambe indicative di una connessione molto rapida, adatta per attività come streaming di video in alta definizione, download di file di grandi dimensioni e giochi online. La latenza della connessione, o ping, è di 21 ms, un valore basso che indica una connessione reattiva ideale per applicazioni che richiedono tempi di risposta immediati come giochi online e videochiamate. Il test è stato eseguito utilizzando connessioni multiple simultaneamente per garantire un'accurata misurazione delle velocità. Il server utilizzato per il test è gestito da Wispone Srl a Chiaravalle, con l'IP 193.205.130.185 del server GARR. Inoltre, è presente una scala di valutazione per il servizio clienti di GARR, da 1 a 5, per raccogliere feedback sull'esperienza. Complessivamente, i risultati mostrano una connessione Internet molto performante, adatta per attività online intensive con un'ottima qualità di servizio percepita.

Considerando l'implementazione del NAT su un sistema operativo Ubuntu, con l'indirizzo LAN 192.168.1.0, l'indirizzo WAN 10.100.0.115, e un computer con indirizzo IP 192.168.1.22 e gateway 192.168.1.10, è stata verificata la corretta instradamento dei pacchetti dal router Mikrotik di frontiera.



L'immagine mostra la schermata del tool "Torch" in esecuzione su un dispositivo MikroTik, utilizzato per monitorare il traffico di rete sull'interfaccia ether3, configurato per catturare il traffico ICMP, con un timeout di 3 secondi. I filtri permettono di catturare traffico da qualsiasi indirizzo IP di origine e destinazione, sia IPv4 che IPv6, senza restrizioni su MAC protocol, porte, VLAN ID o DSCP. Nella tabella dei risultati, due voci di traffico sono visibili: un traffico ICMP proveniente dall'indirizzo IP 10.100.0.115 verso 193.205.130.233 con una velocità di trasmissione e ricezione di 592 bps e un pacchetto trasmesso, e un traffico ARP con una velocità di trasmissione di 336 bps ma senza pacchetti ricevuti. Torch permette di avviare o fermare la cattura del traffico e aprire nuove finestre di monitoraggio, fornendo un efficace strumento per diagnosticare problemi di rete e monitorare il traffico in tempo reale su specifiche interfacce.



L'immagine mostra i risultati di un test di velocità della connessione Internet eseguito con Speedtest di Ookla. La velocità di download misurata è di 77.49 Mbps, mentre la velocità di upload è di 75.53 Mbps, entrambe indicanti una connessione molto rapida e affidabile, adatta per attività ad alta intensità di dati come streaming, giochi online e trasferimenti di file di grandi dimensioni. Il ping è di 21 ms, un valore basso che indica una latenza minima, essenziale per applicazioni che richiedono tempi di risposta immediati, come i giochi online e le videochiamate.

In conclusione, è stato dimostrato che le prestazioni sono identiche: la piattaforma di virtualizzazione sviluppata funziona correttamente e offre le stesse prestazioni di un router fisico (senza virtualizzazione).