



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Dipartimento di Informatica - Scienza e Ingegneria

Corso di Laurea in Ingegneria e Scienze Informatiche

**Analisi e Mitigazione delle
Vulnerabilità nel Protocollo BGP:
Un Approccio al Rafforzamento della
Sicurezza nelle Reti di
Telecomunicazioni**

Relatore:
Chiar.mo Prof.
Andrea Piroddi

Presentata da:
Roberto Pisu

Sessione Ottobre 2025
Anno Accademico 2024/2025

(DA FARE ALLA FINE)

5 parole chiave per caratterizzare il contenuto della dissertazione:
(se non ti piacciono così sparse puoi anche semplicemente scriverle su una riga sola)

parola 5

parola 4

parola 3

parola 2

Parola 1

*Alla mia famiglia
ai miei amici e a chi mi è stato accanto.
Che questo traguardo sia solo l'inizio.*

Abstract

Abstract qui (ti consiglio di farlo alla fine)

Indice

0	INTRODUZIONE	1
1	Autonomous System	5
1.1	Il ruolo dell'AS in Internet	5
1.1.1	Cos'è il Regional Internet Registry	5
1.1.2	Struttura ASN	7
1.2	Classificazione AS	7
2	Protocollo di routing BGP 4.0	9
2.1	Cos'è il routing	10
2.2	Nascita del protocollo	10
2.2.1	Sostituzione EGP	10
2.3	Tipologia di protocollo (livello, a che altri protocolli di rete si appoggia)	10
2.4	Funzionamento BGP	10
2.4.1	Path vector	10
2.4.2	Attributi BGP (a cosa serve ognuno)	10
2.4.3	Formato dei messaggi BGP	10
2.4.4	Sessioni BGP (eBGP, iBGP)	10
2.5	Differenze tra le varie versioni fino all'attuale 4.0	10
3	Metodologie di attacco e prevenzione del protocollo BGP	11
3.1	BGP prefix hijacking	12
3.1.1	funzionamento	12
3.1.2	conseguenze	12
3.1.3	che vulnerabilità sfrutta	12
3.2	BGP route leaking	12
3.2.1	funzionamento	12
3.2.2	conseguenze	12
3.2.3	che vulnerabilità sfrutta	12
3.3	BGP session hijacking	12

3.3.1	funzionamento	12
3.3.2	conseguenze	12
3.3.3	che vulnerabilità sfrutta	12
3.4	BGPsec	12
3.4.1	Come funziona BGPsec	12
3.4.2	Vulnerabilità risolte	12
3.4.3	Firma digitale con curve ellittiche (ECDSA)	12
3.5	BGP RPKI	13
3.5.1	Come funziona BGP RPKI	13
3.5.2	Vulnerabilità risolte	13
3.6	Monitoraggio e rilevamento anomalie	13
3.6.1	BGPStream (CAIDA)	13
3.6.2	CAIDA	13
3.6.3	RIPE RIS	13
3.6.4	Route Views	13
4	Implicazioni degli attacchi BGP nel mondo	15
4.1	BGP hijacking Pakistan Telecom 2008	15
4.2	BGP hijacking di China Telecom	15
5	Tecnologie utilizzate	17
5.1	Ambiente di virtualizzazione	17
5.1.1	VMware Workstation Pro	17
5.1.2	LXC	20
5.2	FRRouting	23
5.3	GoBGP	24
5.4	Topologia rete	25
5.4.1	Struttura rete generale (topologia a maglia parziale tra 7 AS)	26
5.4.2	Struttura singolo AS	27
6	Sviluppo e implementazione	29
6.1	Creazione di un AS	29
6.1.1	Creazione VM con ubuntu server	29
6.1.2	Configurazione ubuntu server	34
6.1.3	Installazioni preliminari sulla VM host	37
6.1.4	Installazione FRR sui container	42
6.1.5	Installazione GoBGP sui container dei Border Router	43
6.1.6	Configurazione OSPF nei router	44
7	Prospettive future: SDN e BGP	47
7.1	Cos'è la SDN	47

7.1.1	Architettura e principio di separazione control/data plane . .	47
7.1.2	Vantaggi principali: flessibilità, programmazione, automazione	47
7.2	Integrazione tra SDN e BGP	47
7.2.1	Routing interdominio gestito centralmente	47
7.2.2	Esempi di progetti o framework (es. SDX, BGP-SDN)	47
7.3	Prospettive evolutive	47
7.3.1	Reti programmabili e scenari futuri	47
7.3.2	Possibili impatti su sicurezza e gestione globale	47

Elenco delle tabelle

5.1	Confronto tra LXC e Docker ^[6]	22
-----	---	----

Elenco delle figure

1.1	Mappa dei RIR ^[8]	6
5.1	OS virtualization VS Hardware virtualization	19
5.2	bare-metal vs hosted	19
5.3	para vs full virtualization	20
5.4	Topologia della rete completa	26
5.5	Topologia dell'AS	27
6.1	Schermata di avvio di workstation pro 17	30
6.2	Selezione file iso	31
6.3	Nome e percorso VM	32
6.4	Impostazioni VM	33
6.5	VM completata	34
6.6	Selezione lingua	35
6.7	Ubuntu minimized	35
6.8	Configurazione mirror ubuntu	36
6.9	Configurazione storage	36
6.10	Configurazione bridge virtuali	38
6.11	Configurazione interfacce	39
6.12	Configurazione rete VM host	41

Capitolo 0

INTRODUZIONE

Al giorno d’oggi, Internet rappresenta una delle infrastrutture più critiche e pervasive della nostra società, in quanto viene utilizzata costantemente in molteplici ambiti della vita quotidiana: dalla comunicazione personale e professionale che ci tiene connessi a livello globale, all’accesso immediato a una quantità crescente di informazioni e contenuti multimediali, fino alla gestione di transazioni economiche, servizi bancari, amministrazione pubblica e logistica internazionale.

Semplificando, Internet può essere definita come la “rete delle reti”: una struttura complessa e gerarchica che consente il collegamento tra milioni di reti locali eterogenee, distribuite in tutto il mondo. A livello architetturale, Internet è composta da un numero elevato, ma finito, di Autonomous System (AS), ciascuno dei quali corrisponde a una rete di proprietà e gestione unificata — come ad esempio un Internet Service Provider (ISP), un’azienda, un’università o un ente governativo — caratterizzata da una propria politica di routing indipendente.

(Si stima l’esistenza di più di 90.000 AS)

Per “politica di routing” si intende l’insieme di regole, preferenze e vincoli che determinano in che modo il traffico di rete viene instradato all’interno dell’AS e verso gli altri sistemi autonomi. Il protocollo di routing incaricato di gestire la propagazione delle informazioni tra AS è il Border Gateway Protocol (BGP), attualmente considerato lo standard de facto per l’interconnessione a livello globale. Il suo com-

pito è annunciare e apprendere rotte ¹, determinando così i percorsi lungo i quali i pacchetti viaggiano da un'estremità all'altra del mondo.

Nonostante la sua centralità e longevità, il protocollo BGP presenta una serie di limiti strutturali, legati al fatto che fu progettato in un'epoca in cui la sicurezza informatica non era ancora una priorità. BGP si basa infatti su un modello di fiducia implicita tra gli operatori di rete e non prevede, nella sua implementazione standard, meccanismi di autenticazione, integrità o verifica delle informazioni propagate. Questa mancanza di sicurezza nativa rende il protocollo vulnerabile a diversi tipi di attacco, tra cui il *prefix hijacking*, il *route leaking* e il *session hijacking*, che possono compromettere seriamente l'affidabilità e la sicurezza della rete Internet.

Alla luce di queste considerazioni, risulta fondamentale analizzare in dettaglio il funzionamento di BGP e le sue vulnerabilità, al fine di individuare le possibili contromisure per prevenire o limitare gli effetti di un eventuale attacco. In un mondo sempre più dipendente dall'utilizzo di Internet, garantire la resilienza e la sicurezza del protocollo di routing interdominio è una priorità non solo tecnica, ma anche strategica e geopolitica.

La tesi si compone di sette capitoli suddivisi come segue:

- **Primo Capitolo - Autonomous System** Nel primo capitolo viene approfondito il ruolo dell'AS, le sue caratteristiche, a cosa serve, come vengono classificati e altre informazioni utili a comprendere il funzionamento del protocollo BGP.
- **Secondo Capitolo - Protocollo di routing BGP 4.0** In questo capitolo viene analizzato in dettaglio il funzionamento del protocollo BGP (Border Gateway Protocol), attualmente lo standard principale per il routing tra sistemi autonomi su Internet. Dopo una panoramica introduttiva sui concetti fondamentali di routing, viene descritto il contesto storico e tecnico che ha portato all'introduzione di BGP, in particolare come evoluzione del precedente Exterior Gateway Protocol (EGP). Viene poi approfondita la natura del protocollo, la sua collocazione nei livelli del modello di rete e le dipendenze da altri protocolli sottostanti. Una sezione centrale del capitolo è dedicata

¹Una rotta è il percorso scelto dal protocollo di routing per raggiungere una rete specifica.

al funzionamento interno di BGP: vengono spiegati il meccanismo di routing basato su path vector, l'applicazione delle politiche di importazione ed esportazione delle rotte, gli attributi utilizzati per determinare il miglior percorso e le strategie per evitare la formazione di cicli. Il capitolo si conclude con una descrizione del formato dei messaggi BGP e delle sessioni di peering tra router (iBGP ed eBGP), per poi presentare un confronto tra le versioni storiche del protocollo e le novità introdotte nell'attuale versione 4.0.

- **Terzo Capitolo - Metodologie di attacco e prevenzione del protocollo BGP** Nel terzo capitolo vengono approfondite le principali vulnerabilità del protocollo BGP nella sua ultima versione 4.0, ne vengono analizzate le cause, le modalità di esecuzione dell'attacco e le possibili conseguenze. Vengono trattati tre scenari d'attacco particolarmente rilevanti: il *prefix hijacking*, in cui un AS annuncia indebitamente prefissi IP altrui; il *route leaking*, che comporta la diffusione impropria di rotte apprese da altri peer; e il *session hijacking*, in cui un attore malevolo intercetta o falsifica la sessione BGP tra due router. Per ciascun attacco vengono analizzati nel dettaglio il funzionamento tecnico, le ripercussioni sul traffico e le vulnerabilità specifiche che vengono sfruttate. Successivamente, il capitolo introduce due tecnologie progettate per aumentare la sicurezza di BGP: Border Gateway Protocol Security (BGPsec), che fornisce autenticazione crittografica delle informazioni di routing tramite firme digitali basate su curve ellittiche (ECDSA), e Resource Public Key Infrastructure (RPKI), un'infrastruttura a chiave pubblica che permette di validare l'autorità di un AS ad annunciare un determinato prefisso IP. Vengono evidenziate le differenze tra le due soluzioni e i rispettivi ambiti di applicazione.

Nella parte finale, viene trattato il tema del monitoraggio delle anomalie BGP attraverso piattaforme pubbliche come *BGPStream* (di Cooperative Association for Internet Data Analysis (CAIDA)), *RIPE RIS* e *Route Views*, strumenti fondamentali per l'analisi forense, la diagnosi di incidenti e l'early warning² nel routing globale.

- **Quarto Capitolo - Implicazioni degli attacchi BGP nel mondo** Questo

²L'early warning è la capacità di rilevare tempestivamente anomalie o potenziali attacchi alla rete.

capitolo analizza due noti casi reali di attacchi al protocollo BGP, con l'obiettivo di mostrare le conseguenze concrete che vulnerabilità teoriche possono avere su scala globale. Il primo caso è quello del BGP hijacking da parte di Pakistan Telecom nel 2008, che causò l'inaccessibilità mondiale di YouTube per diverse ore. Il secondo riguarda China Telecom, coinvolta in episodi di deviazione di traffico internazionale tra il 2010 e il 2018.

- **Quinto Capitolo - Tecnologie utilizzate** Nel quinto capitolo entriamo nel vivo della simulazione, vengono infatti presentate le principali tecnologie adottate per costruire l'ambiente virtuale necessario alla simulazione degli attacchi e delle contromisure al protocollo BGP. L'infrastruttura è basata su Virtual Machine (VM) Ubuntu Server create con VMware Workstation Pro, che sfrutta virtualizzazione hardware di tipo 2 (hosted) con supporto alla full virtualization assistita da hardware.

All'interno di ogni VM vengono eseguiti container LXC, impiegati per simulare i router interni (Internal Router (IR)) in modo efficiente. Per la gestione del routing interno è stato utilizzato Free Range Routing (FRR), mentre per i router di confine (Border Router (BR)) è stato adottato GoBGP, particolarmente adatto alla configurazione delle sessioni BGP e alla sperimentazione di meccanismi come RPKI e BGPsec.

La rete complessiva è strutturata secondo una topologia a maglia parziale tra sette AS, ciascuno dotato di una struttura interna realistica composta da router, switch e VLAN. Tale configurazione permette di testare scenari di routing complessi in un ambiente controllato e riproducibile.

- **Sesto Capitolo - Sviluppo e implementazione**
- **Settimo Capitolo - Prospettive future: SDN e BGP**

Capitolo 1

Autonomous System

In questo capitolo, andiamo ad analizzare cos'è un AS, il suo ruolo nel routing globale e la loro classificazione.

1.1 Il ruolo dell'AS in Internet

Un AS è definito come un insieme di indirizzi IP e router sotto il controllo di una singola entità amministrativa, che adotta una politica di routing uniforme e coerente verso l'esterno. Secondo l'*RFC 1930* dell'IETF, un AS è necessario ogniqualvolta un'organizzazione desidera definire delle regole di instradamento proprie e differenziate rispetto ad altri domini di routing, oppure quando intrattiene relazioni di peering con più fornitori di connettività a Internet. Ogni AS è identificato univocamente dal Autonomous System Number (ASN), assegnato da uno dei cinque Regional Internet Registry (RIR).^{[5][7]}

1.1.1 Cos'è il Regional Internet Registry

I RIR sono organizzazioni responsabili dell'assegnazione e della registrazione delle risorse numeriche di Internet all'interno di specifiche regioni geografiche del mondo. Le risorse che i RIR assegnano e registrano sono:

- Indirizzi IP sia IPv4 che IPv6.

- ASN usati per identificare gli AS.

Le organizzazioni (come gli ISP, grandi aziende, università, enti governativi, ecc.) che desiderano connettersi a Internet in modo indipendente (ovvero, operare il proprio AS) e/o avere un blocco di indirizzi IP pubblico da gestire direttamente, devono richiedere queste risorse al RIR competente per la loro area geografica.

Attualmente, esistono 5 RIR a livello globale:

- AfriNIC (Africa)
- ARIN (Nord America)
- LACNIC (America Latina e Caraibi)
- APNIC (Asia e Oceania)
- RIPE NCC (Europa, Medio Oriente e parti dell'Asia Centrale)

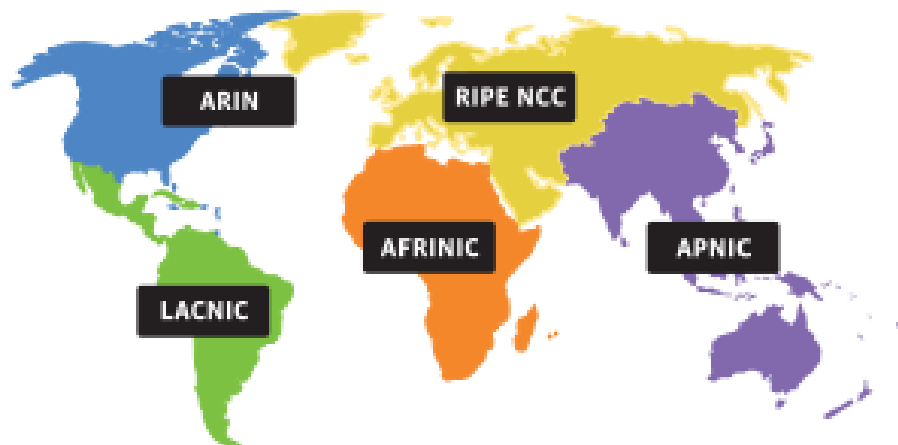


Figura 1.1: Mappa dei RIR^[8]

Queste 5 RIR collaborano attraverso l'Number Resource Organization (NRO) e sono sotto la supervisione generale dell'ente Internet Assigned Numbers Authority (IANA), che è il coordinatore globale delle risorse numeriche e dei nomi di dominio di Internet.

1.1.2 Struttura ASN

La nomenclatura degli ASN segue due formati principali: il tradizionale a 16 bit e quello a 32 bit, introdotto successivamente per rispondere all'aumento della domanda (RFC 4893). Gli ASN a 16 bit vanno da 1 a 65535, con alcune riserve speciali: per esempio, i numeri da 64512 a 65534 sono destinati all'uso privato, mentre l'ASN 23456 viene usato come placeholder nella transizione tra 16 e 32 bit (RFC 6996). Il nuovo spazio a 32 bit estende la numerazione fino a 4.294.967.295 e consente anche l'utilizzo della notazione m:n (es. 1:10), che rappresenta una forma più leggibile del numero intero.

Gli ASN vengono spesso preceduti dal prefisso AS (es: AS13335 per Cloudflare, AS15169 per Google) e sono registrati pubblicamente in database come il WHOIS.¹^[10]

1.2 Classificazione AS

Gli Autonomous System possono essere classificati secondo diversi criteri, in base al loro ruolo funzionale nella topologia globale di Internet, alla natura delle connessioni che intrattengono con altri AS, oppure alle politiche di routing adottate.

Una classificazione comunemente diffusa è quella che distingue gli AS in tre grandi categorie:

- **Stub AS:** AS che è connesso a uno o più provider, ma non fornisce connettività ad altri AS. È il caso tipico di una rete aziendale o universitaria.
- **Multihomed AS:** AS connesso a più provider, senza però fornire transito ad altri.
- **Transit AS:** AS che offre connettività ad altri sistemi autonomi, permettendo loro di scambiare traffico. Gli ISP operano tipicamente come transit AS.

Un'altra classificazione si basa sul ruolo gerarchico ricoperto all'interno dell'ecosistema di Internet:

¹WHOIS è un archivio pubblico che raccoglie informazioni sulla titolarità e sull'assegnazione di risorse di rete.

- **Tier 1:** AS che può raggiungere qualsiasi altra rete senza dover acquistare transito da altri AS.
- **Tier 2:** AS che acquista transito, ma può anche offrire servizi a clienti e stabilire peering.
- **Tier 3:** AS che acquista connettività esclusivamente da altri provider e non fornisce servizi di transito.

Infine, secondo le linee guida dell'Internet Engineering Task Force (IETF) (RFC 1930), un Autonomous System è definito anche in base all'autonomia decisionale rispetto alle politiche di routing. Questa indipendenza costituisce uno dei principali motivi per cui un'organizzazione può richiedere un ASN.

Capitolo 2

Protocollo di routing BGP 4.0

2.1 Cos'è il routing

2.2 Nascita del protocollo

2.2.1 Sostituzione EGP

2.3 Tipologia di protocollo (livello, a che altri protocolli di rete si appoggia)

2.4 Funzionamento BGP

2.4.1 Path vector

Come si applicano le politiche (import e export) policy

Attributi path vector

Come si evitano i cicli

2.4.2 Attributi BGP (a cosa serve ognuno)

2.4.3 Formato dei messaggi BGP

2.4.4 Sessioni BGP (eBGP, iBGP)

2.5 Differenze tra le varie versioni fino all'attuale 4.0

Capitolo 3

Metodologie di attacco e prevenzione del protocollo BGP

3.1 BGP prefix hijacking

3.1.1 funzionamento

3.1.2 conseguenze

3.1.3 che vulnerabilità sfrutta

3.2 BGP route leaking

3.2.1 funzionamento

3.2.2 conseguenze

3.2.3 che vulnerabilità sfrutta

3.3 BGP session hijacking

3.3.1 funzionamento

3.3.2 conseguenze

3.3.3 che vulnerabilità sfrutta

12

3.4 BGPsec

3.4.1 Come funziona BGPsec

3.4.2 Vulnerabilità risolte

3.4.3 Firma digitale con curve ellittiche (ECDSA)

3.5 BGP RPKI

3.5.1 Come funziona BGP RPKI

3.5.2 Vulnerabilità risolte

3.6 Monitoraggio e rilevamento anomalie

3.6.1 BGPStream (CAIDA)

3.6.2 CAIDA

3.6.3 RIPE RIS

3.6.4 Route Views

Capitolo 4

Implicazioni degli attacchi BGP nel mondo

4.1 BGP hijacking Pakistan Telecom 2008

4.2 BGP hijacking di China Telecom

Capitolo 5

Tecnologie utilizzate

In questo capitolo, esamineremo le tecnologie scelte e il loro ruolo nella costruzione del progetto.

5.1 Ambiente di virtualizzazione

Data l'impossibilità di utilizzare, gratuitamente o a basso costo, dei router in grado di supportare BGPsec e RPKI, utilizzeremo delle VM con sistema operativo Ubuntu server 24.04.2. Su queste VM, monteremo poi dei software (FRRouting per i IR e GoBGP per i BR) per rendere le VM dei router effettivi.

5.1.1 VMware Workstation Pro

Come software di virtualizzazione desktop per contenere le VM utilizziamo VMware Workstation Pro nella sua versione 17.6.3. VMware Workstation Pro, nel 2024 è diventato gratuito nella sua versione per uso personale, e ciò ha contribuito ulteriormente nella sua diffusione, ad oggi esso è infatti uno dei software di virtualizzazione desktop più utilizzato.^[9] Quando creiamo delle VM con VMware Workstation Pro, usiamo la virtualizzazione hardware di tipo 2 (hosted) con full virtualization assistita da hardware.

Terminologia della virtualizzazione

- **Sistema host:** è la macchina fisica su esegue il sistema operativo principale e che poi ospiterà le macchine virtuali.
- **Sistema guest:** è l'insieme delle risorse hardware e SO che viene eseguito "sopra" il sistema host.
- **Hypervisor o Virtual Machine Monitor (VMM):** è il componente software che crea e manda in esecuzione le VM. Ha il compito di astrarre e rendere disponibili le risorse hardware e svolge i compiti di monitoraggio e sicurezza.

Virtualizzazione Desktop

La virtualizzazione desktop, è un tipo di virtualizzazione che:

- Consente di utilizzare una VM che esegue sul PC dell'utente, in questo modo la macchina virtuale sfrutta le periferiche della macchina fisica (mouse, tastiera, schermo..) per consentire all'utente l'interazione con il sistema operativo guest.
- Viene realizzata mediante l'utilizzo di particolari software che permettono di eseguire altri sistemi operativi "sopra" il sistema operativo host.
- È utile per far funzionare un software non compatibile con il sistema operativo dell'host o un software che si vuole mantenere separato dal sistema operativo host.

Virtualizzazione livello hardware

Nella virtualizzazione livello hardware (macchine virtuali), all'utente del sistema di virtualizzazione viene presentata un'interfaccia su cui installare un sistema operativo, quindi una CPU virtuale (ma dello stesso tipo della CPU fisica) e risorse hardware virtuali. Invece, nel caso di virtualizzazione livello di sistema operativo (container), all'utente viene presentata una partizione del sistema operativo corrente, su cui installare ed eseguire applicazioni che rimangono isolate nella partizione, pur accedendo ai servizi di uno stesso sistema operativo.

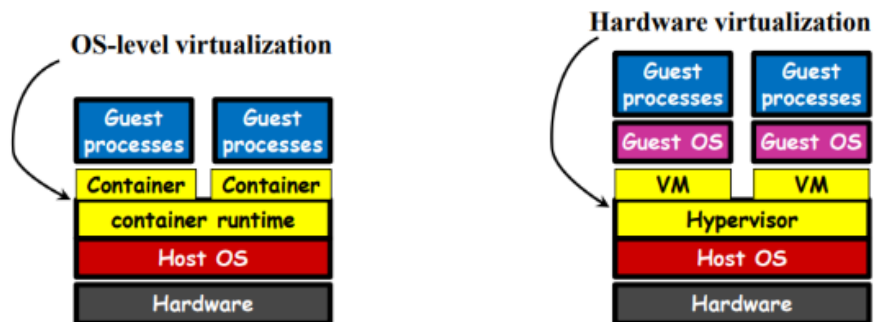


Figura 5.1: OS virtualization VS Hardware virtualization

Virtualizzazione di tipo 2 (hosted)

Si definisce virtualizzazione di tipo 2 o **hosted** un sistema di virtualizzazione nel quale l'hypervisor è un normale processo utente sul sistema operativo host. Mentre si definisce virtualizzazione di tipo 1 o **bare-metal** un sistema di virtualizzazione nel quale il sistema operativo host è assente e le sue funzioni vengono sostituite dall'hypervisor.

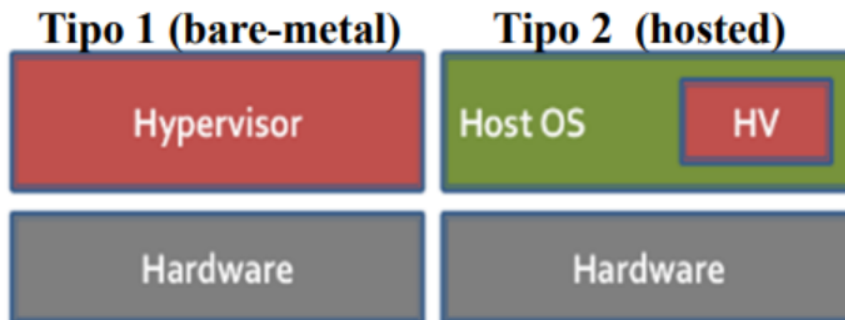


Figura 5.2: bare-metal vs hosted

Virtualizzazione completa

Nella virtualizzazione completa (o full virtualization), vengono fornite VM che hanno la medesima interfaccia di una macchina fisica. Idealmente, il sistema operativo guest non può identificare che si trova su una macchina virtuale (in realtà di solito è sempre possibile dedurlo attraverso opportune tecniche). Invece, nella para

virtualization la VM presenta un'interfaccia diversa confrontata ad una macchina fisica. Questo comporta dover modificare il sistema operativo guest per consentire l'esecuzione all'interno della macchina virtuale stessa.

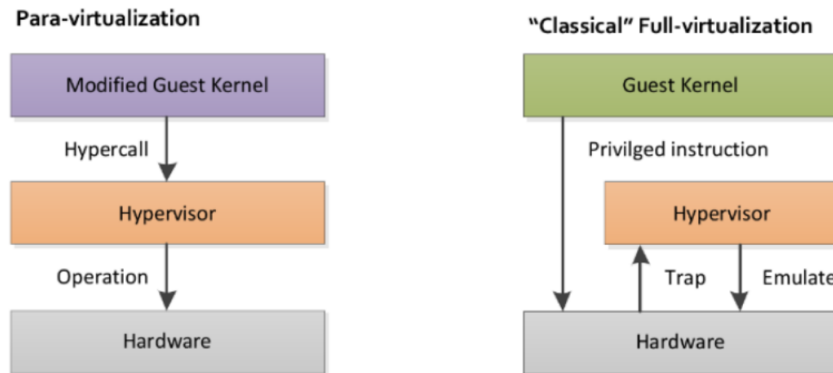


Figura 5.3: para vs full virtualization

5.1.2 LXC

Nel progetto, oltre all'utilizzo di macchine virtuali complete tramite VMware, si è resa necessaria una soluzione più leggera e scalabile per simulare un numero elevato di router appartenenti ai vari AS.

Cos'è LXC

LXC (Linux Containers) è una tecnologia che permette di creare container Linux con un sistema operativo completo, isolati tra loro ma in grado di condividere lo stesso kernel Linux dell'host. A differenza delle VM, i container LXC non virtualizzano l'intero hardware, ma sfruttano le funzionalità native del kernel come **cgroups** e **namespaces** per ottenere isolamento e gestione delle risorse.

Ogni container LXC può essere considerato una "mini-macchina Linux", capace di eseguire più processi, avviare demoni di sistema (come **systemd**) e simulare realisticamente il comportamento di un router software. Per questo motivo, è particolarmente adatto all'uso con **FRRouting**, in modo da trasformare ciascun container in un router.

Motivazioni della scelta

L'utilizzo esclusivo di VM per ogni router avrebbe comportato la creazione e gestione di oltre 20 istanze complete di Ubuntu Server, con un notevole consumo di risorse hardware. LXC offre una soluzione molto più efficiente: consente di simulare decine di router su un'unica VM host o macchina fisica, mantenendo un livello di realismo sufficiente per lo studio dei protocolli di routing. In questo modo, andremo a creare 7 VM ubuntu server, e su ognuna 4 LinuX Container (LXC).

LXC vs Docker

Caratteristica	LXC	Docker
Tipo di container	Container di sistema (full OS)	Container per singola applicazione
Sistema operativo	Completo, con supporto per systemd , più processi, demoni	Minimo, pensato per un processo alla volta; systemd non supportato nativamente
Architettura	Accesso diretto a kernel, cgroups, namespaces; somiglia a una VM leggera	Strato intermedio (Docker Engine); isolamento a livello applicativo
Facilità di configurazione	Più complesso, orientato a sysadmin	Semplice, pensato per sviluppatori e DevOps
Gestione dei processi	Più processi supportati nativamente	Singolo processo per container (multi-processo solo con workaround)
Rete	Networking avanzato con configurazioni personalizzabili (bridge, veth, ecc.)	Networking gestito dal motore Docker; meno flessibile
Supporto a demoni e init system	Completo (systemd , init , cron)	Limitato, solo con immagini o configurazioni speciali
Caso d'uso ideale	Simulare sistemi Linux completi (es. router, server reali)	Deploy di microservizi, app web, pipeline CI/CD
Ecosistema	Più ridotto, ma potente e vicino al kernel Linux	Ampissimo: Docker Hub, Docker Compose, Kubernetes

Tabella 5.1: Confronto tra LXC e Docker^[6]

Integrazione con VMware

In questo progetto, LXC è stato utilizzato all'interno di una macchina virtuale host Ubuntu Server, eseguita su VMware. In questo modo si ottiene il vantaggio di una struttura modulare: le VM rappresentano i confini tra gli AS, mentre i container LXC al loro interno permettono di simulare in modo efficiente i router interni. I

container sono interconnessi tramite bridge virtuali creati con **bridge-utils**, che permettono di emulare le reti locali tra gli IR e gli BR di ciascun AS.

Vantaggi principali

- **Efficienza:** ogni container consuma pochi megabyte di RAM.
- **Realismo:** ambiente Linux completo, con supporto per FRRouting e configurazioni di rete avanzate.
- **Scalabilità:** facilità nel creare e clonare container per simulare topologie complesse.
- **Automazione:** possibilità di creare script per lanciare e configurare decine di router containerizzati.

5.2 FRRouting

FRR è un software open-source che implementa numerosi protocolli di routing dinamico, tra cui BGP, Open Shortest Path First (OSPF), Routing Information Protocol (RIP) e altri. Nasce come fork di Quagga nel 2016, con l'obiettivo di offrire una suite più aggiornata, stabile e orientata alla produzione in ambienti carrier-grade, data center e reti ISP.^[1]

FRRouting è scritto in C e si basa su una struttura modulare: ogni protocollo è gestito da un demone¹ separato (es. **ospfd**, **bgpd**), coordinati dal demone **zebra**, responsabile dell'interazione con la tabella di routing del kernel Linux. La comunicazione tra i demoni avviene tramite un socket interno, e la configurazione può essere gestita tramite l'interfaccia a riga di comando **vtysh**. Utilizzeremo l'ultima versione aggiornata a luglio 2025: la 10.3.1.

Nel progetto, FRRouting è utilizzato per trasformare i container LXC in veri e propri IR interni all'AS. In particolare, viene impiegato il protocollo OSPF per abilitare il routing interno all'AS tra i router interni e il router di confine (BR). Alcuni motivi per cui è stato scelto FRR sono:

¹dall'inglese *daemon*, è un processo in background che opera in modo autonomo.

- pieno supporto al protocollo OSPF;
- leggerezza e compatibilità con ambienti containerizzati (es. LXC);
- documentazione aggiornata e ampia community.

FRRouting è usato da numerosi operatori e progetti di rete, tra cui Cumulus Linux, VyOS, Google e altri attori del settore delle telecomunicazioni e dei data center^[2].

5.3 GoBGP

GoBGP è un'implementazione software del protocollo BGP interamente sviluppata in Go. È stato progettato per essere un'applicazione BGP moderna e flessibile, ideale per ambienti che richiedono scalabilità e automazione, come le infrastrutture Software Defined Networking (SDN) e per testare funzionalità avanzate quali RPKI e BGPsec.^[4]

GoBGP è un progetto open-source mantenuto dalla sua community, con il supporto attivo di contributori da NTT Communications e altri operatori. A differenza di molti altri router software, GoBGP adotta un'architettura "daemon-less": si tratta di un unico binario che include tutte le funzionalità. Questo lo rende gestibile in modo flessibile tramite API gRPC o file di configurazione in formato YAML o TOML^[3]

Nel progetto, GoBGP viene utilizzato per configurare i router di confine (BR) di ciascun AS (in particolare utilizzeremo l'ultima versione aggiornata a luglio 2025: la 3.37.0). A differenza di FRRouting, GoBGP non implementa protocolli Interior Gateway Protocol (IGP) come OSPF o RIP, ma offre supporto avanzato per:

- validazione RPKI tramite RTR (RFC 6810/8210);
- supporto a BGPsec (in parte sperimentale);
- API per interazioni dinamiche e test automatizzati;
- capacità di annunciare/ritirare prefissi dinamicamente.

Questo rende GoBGP ideale per la simulazione di attacchi alla sicurezza di BGP (es. prefix hijacking, route leaking) e per testare le contromisure basate su RPKI.

5.4 Topologia rete

Una topologia di rete descrive come i nodi e le connessioni sono organizzati, sia fisicamente che logicamente, all'interno di una rete. Per il progetto è stata adottata una topologia fisica a maglia parziale, al fine di avvicinarci maggiormente a ciò che avviene nella realtà. Mentre nella topologia a maglia completa tutti i nodi sono collegati tra loro, nella topologia a maglia parziale non tutti i nodi sono collegati. Nella realtà, la topologia ha una struttura gerarchico-scalabile, in cui i grandi provider sono fortemente connessi tra loro, mentre i più piccoli hanno connessioni limitate verso l'alto. Vediamo ora alcune caratteristiche di una topologia a maglia parziale:

- ha una buona scalabilità ²
- non tutti i nodi sono collegati tra loro, ma molti dispongono di connessioni multiple verso determinati nodi, ciò garantisce percorsi alternativi in caso di malfunzionamenti
- rispetto a una topologia a maglia completa, quella parziale più semplice da realizzare e meno costosa (banalmente perché richiede meno link fisici e meno hardware).

²La scalabilità è la caratteristica di una rete di aggiungere o rimuovere nodi facilmente.

5.4.1 Struttura rete generale (topologia a maglia parziale tra 7 AS)

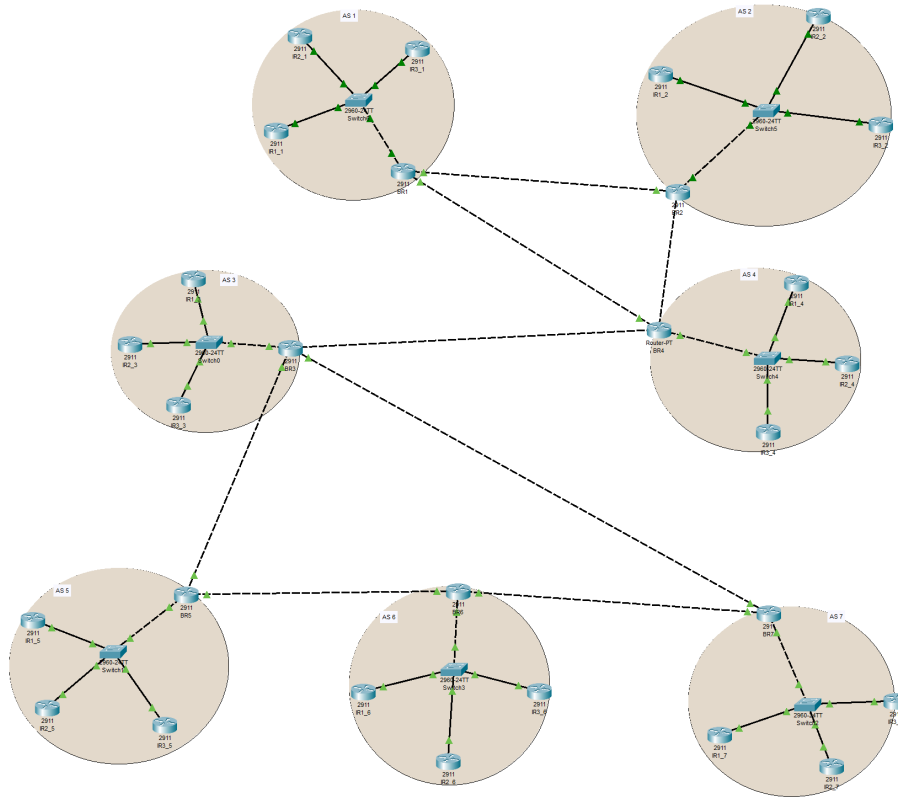


Figura 5.4: Topologia della rete completa

La rete è composta da sette AS, ciascuno dotato di un proprio sistema interno di routing e di un BR (router di confine). I BR sono connessi tra loro secondo una topologia logica a maglia parziale, in cui ogni router di confine stabilisce connessioni BGP soltanto con un sottoinsieme degli altri AS. I link BGP tra BR sono stati quindi progettati per formare una rete ridondata ma non simmetrica, garantendo percorsi alternativi senza replicare la connettività tra tutti i nodi. Un approccio a simulazioni con questa topologia consente di osservare il comportamento del protocollo BGP in scenari realistici, in cui le decisioni di instradamento devono tener conto di topologie non perfettamente connesse e della propagazione selettiva delle rotte.

5.4.2 Struttura singolo AS

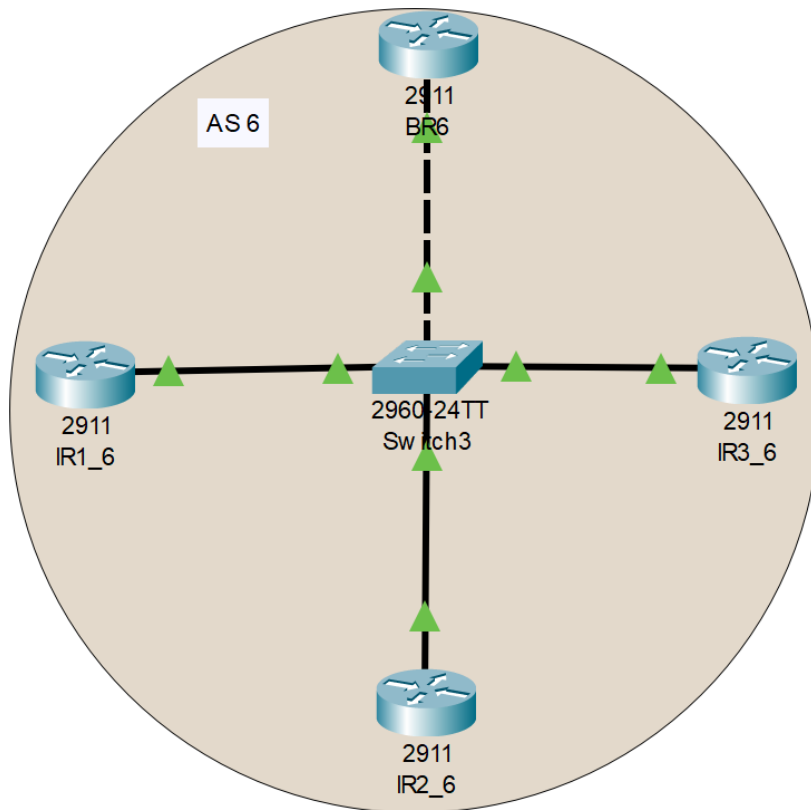


Figura 5.5: Topologia dell'AS

Ogni AS è progettato con una struttura interna composta da moduli funzionali distinti (router interni, router di confine, switch e VLAN), pensata per simulare in modo realistico l'architettura di una rete aziendale. La composizione di ogni AS è la seguente:

- **1 BR:** router di confine che comunica con gli altri BR tramite il protocollo BGP. È responsabile dell'interscambio delle rotte con l'esterno e dell'inoltro del traffico verso la rete interna dell'AS.
- **3 IR:** router interni all'AS, collegati tra loro tramite un protocollo IGP, in questo caso OSPF. Gli IR non comunicano direttamente con l'esterno, ma solo con il BR e tra loro.

- **1 Switch Layer 2** configurato con **4 VLAN**:
 - **3 VLAN dedicate**, ognuna per il collegamento tra un IR e lo switch;
 - **1 VLAN condivisa**, che collega i tre IR e il BR, consentendo il funzionamento del routing interno (OSPF) tra tutti e quattro i router.

Capitolo 6

Sviluppo e implementazione

In questo capitolo, vediamo tutti i passaggi pratici messi in pratica per realizzare l'ambiente di simulazione e i vari test alla sicurezza del protocollo BGP.

6.1 Creazione di un AS

Ora vediamo i passi dettagliati per la creazione di un AS, che ricordiamo essere composto da 3 IR e 1 BR. Inoltre abbiamo anche uno switch con 4 Virtual Local Area Network (VLAN): 3 VLAN per ogni IR e una VLAN che permette ai 4 router di essere collegati tra loro e di gestire il routing tramite il protocollo OSPF.

6.1.1 Creazione VM con ubuntu server

1. Dal sito ufficiale di ubuntu: www.ubuntu.com, andiamo ad installare l'ultima versione della iso del sistema operativo ubuntu server. Nel mio caso l'ultima versione in data Giugno 2025 è: "ubuntu-24.04.2-live-server-amd64.iso".
2. Avviamo VMware Workstation Pro e clicchiamo su "Create a New Virtual Machine".

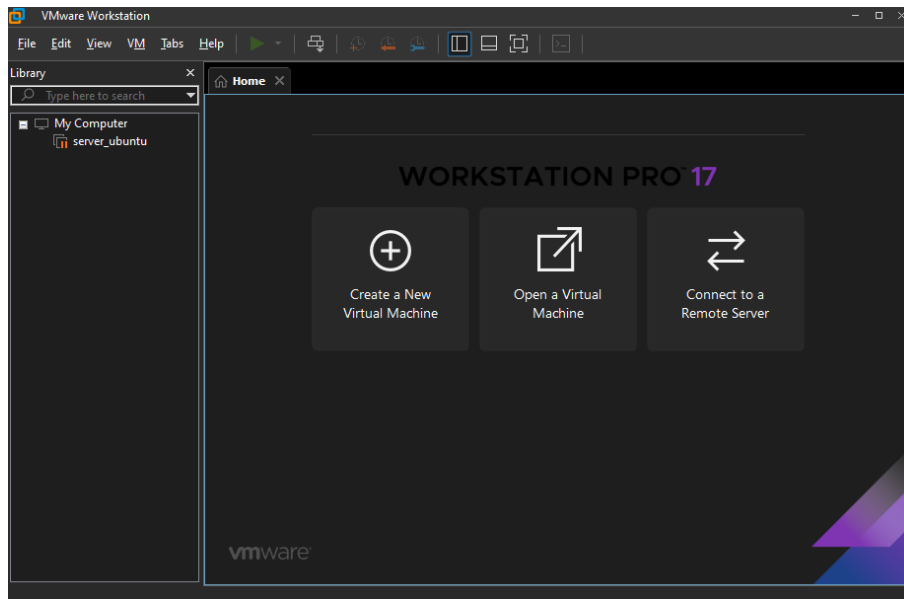


Figura 6.1: Schermata di avvio di workstation pro 17

3. Selezioniamo la iso precedentemente scaricata:

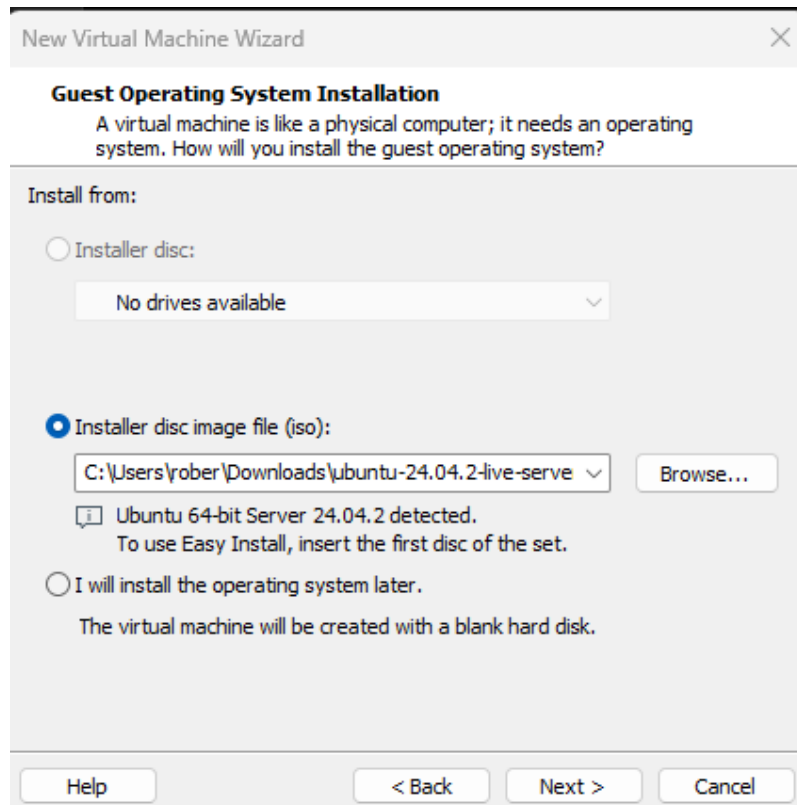


Figura 6.2: Selezione file iso

4. Gli diamo un nome e un percorso dove salvarla nel file system:

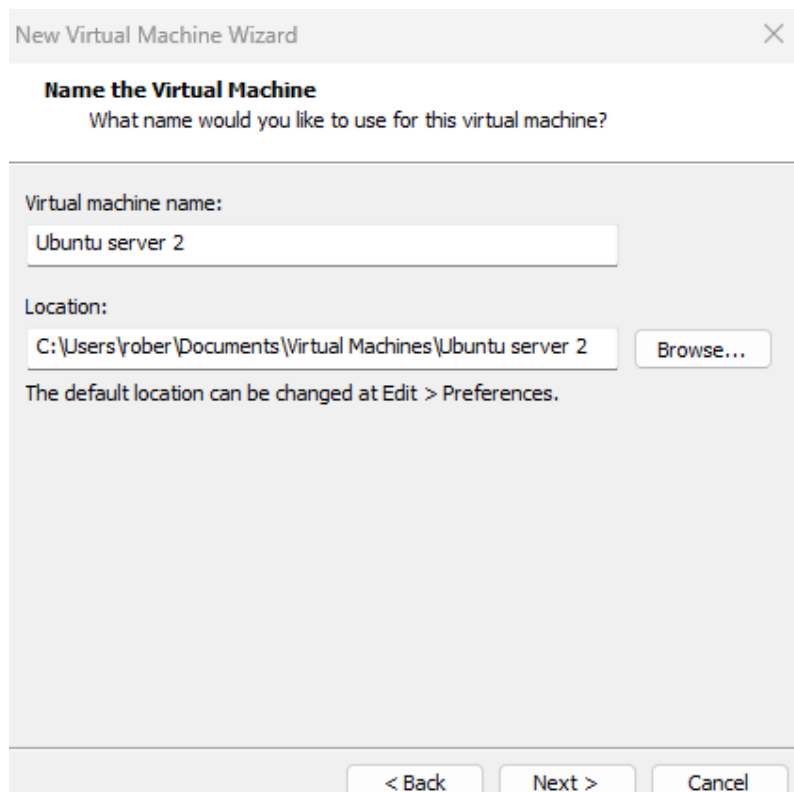


Figura 6.3: Nome e percorso VM

5. Specifichiamo 20 GB di archiviazione interna e l'opzione "Store virtual desk as a single file"

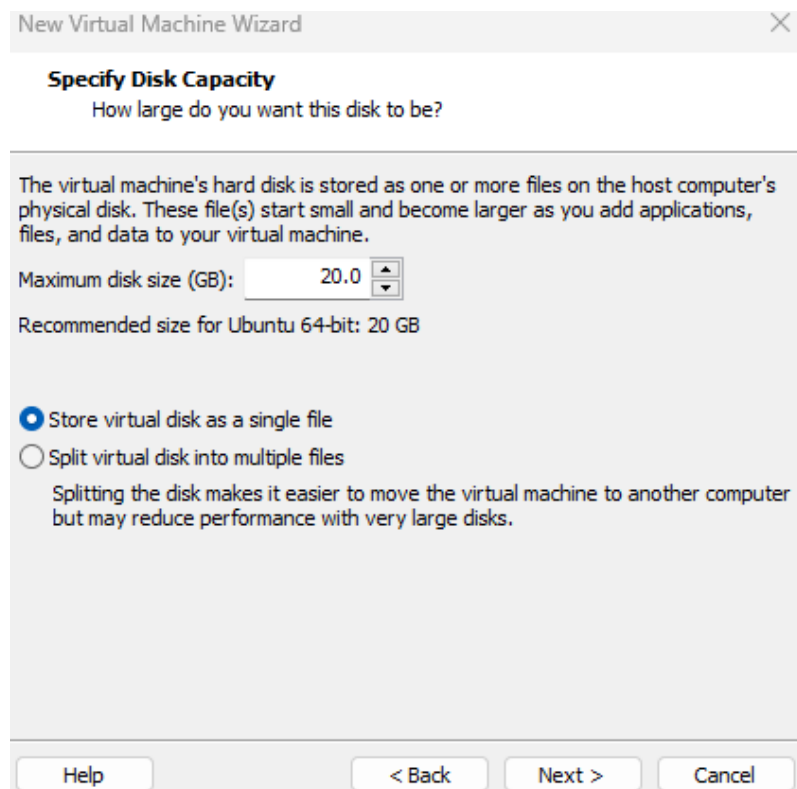


Figura 6.4: Impostazioni VM

6. A questo punto la configurazione è completa e si può avviare la macchina virtuale:

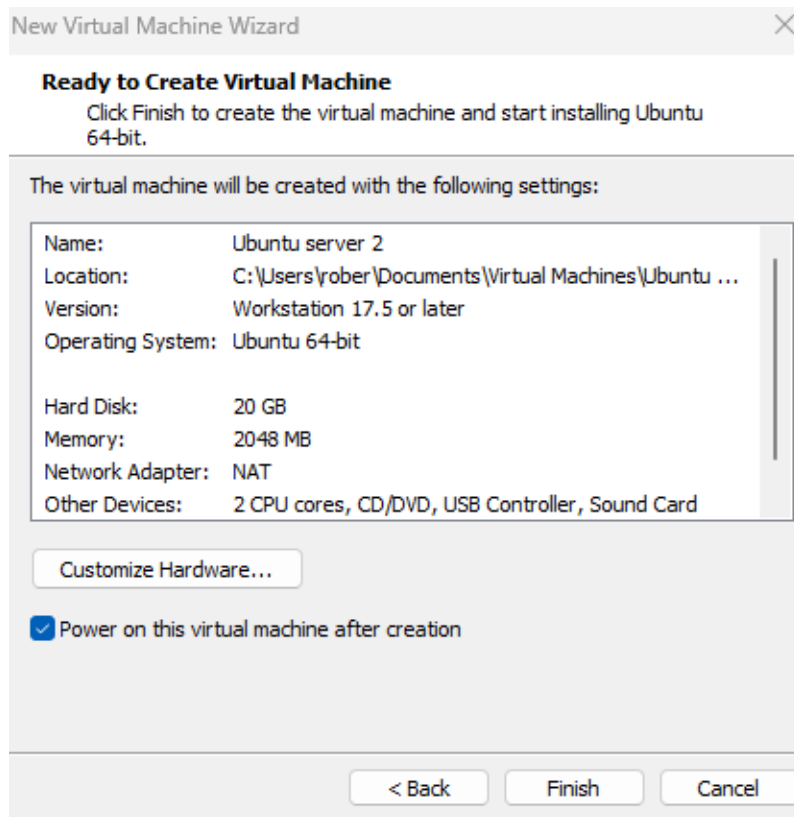


Figura 6.5: VM completata

6.1.2 Configurazione ubuntu server

Al primo avvio della macchina virtuale, il sistema operativo ubuntu server si configurerà in parte in automatico, mentre una parte della configurazione la dobbiamo completare noi manualmente:

1. Selezioniamo la lingua di sistema e quella della tastiera:

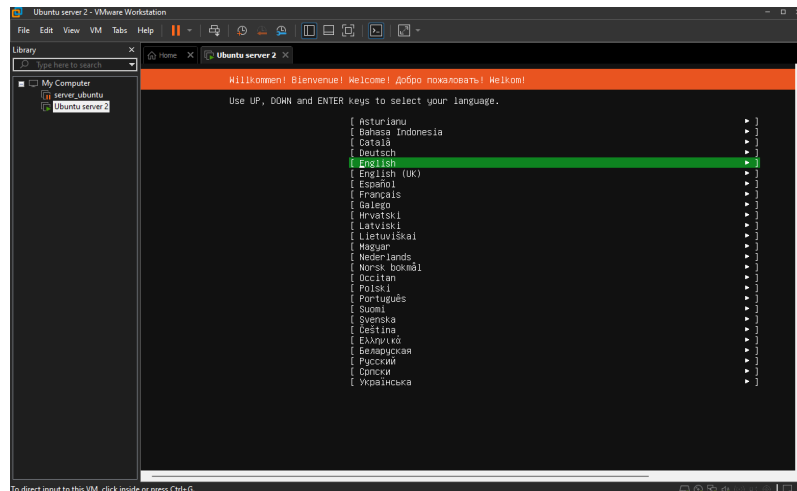


Figura 6.6: Selezione lingua

2. Scegliamo la modalità di installazione "minimized". La modalità minimized di una distribuzione Linux è una modalità più leggera rispetto a quella standard. È ideale al nostro contesto perché:

- riduce l'utilizzo di risorse della VM (disco, RAM);
- è più semplice da configurare;
- evita software non necessario.

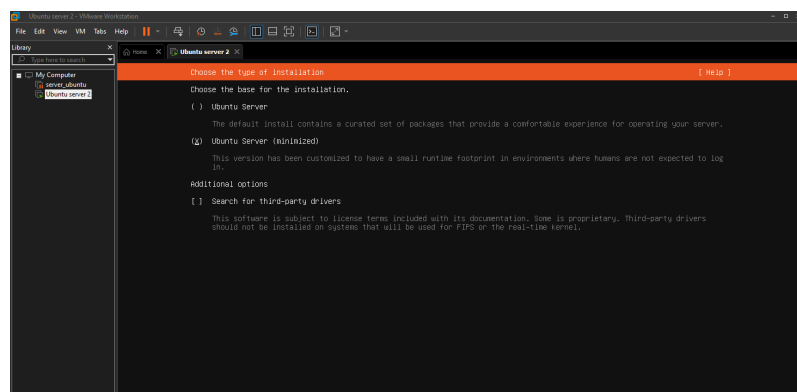


Figura 6.7: Ubuntu minimized

3. La configurazione della rete per ora la lasciamo così, la andremo a modificare in un secondo momento.

4. La configurazione del proxy la lasciamo vuota.
5. A questo punto ci viene chiesto quale mirror usare, lasciamo quello di default e andiamo avanti. Un mirror è un server che contiene una copia dell'archivio ufficiale di Ubuntu e serve principalmente a distribuire pacchetti del sistema operativo e dei software.

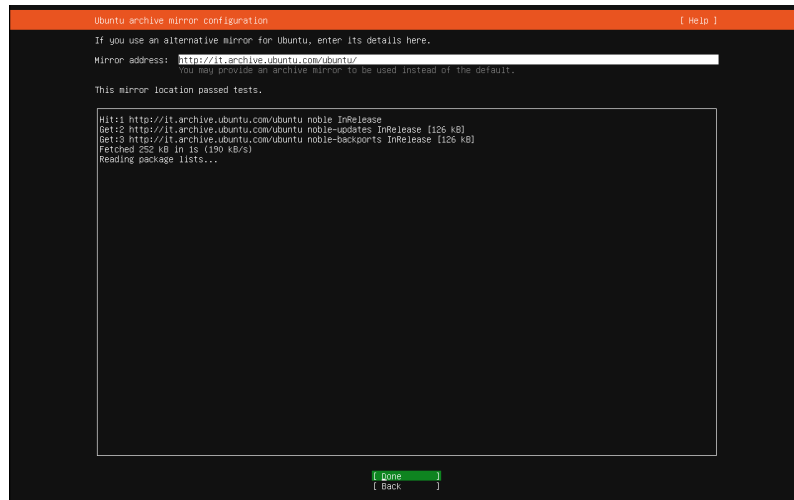


Figura 6.8: Configurazione mirror ubuntu

6. Per la configurazione storage lasciamo così com'è.

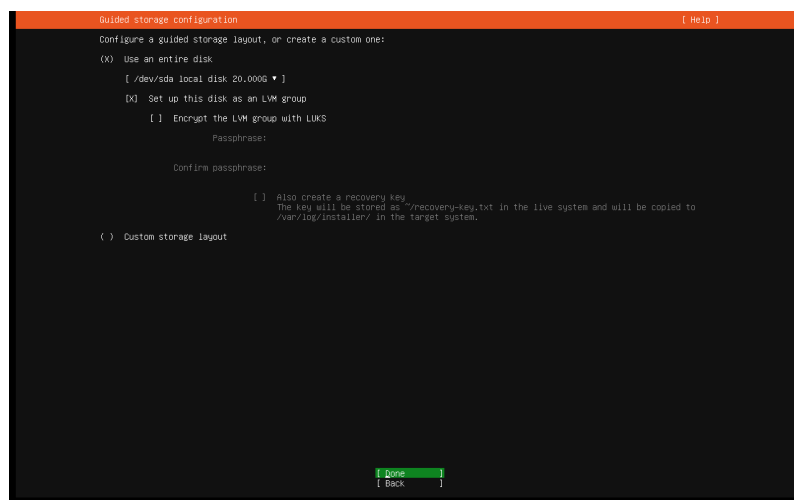


Figura 6.9: Configurazione storage

7. Ora ci viene chiesto di inserire username, nome del server, la password e il nome della macchina. Compiliamo i dati e andiamo avanti.
8. Infine ci verrà chiesto di installare qualche software tra i più utilizzati e altre impostazioni che possiamo ignorare (lasciare tutto com'è di default e andare avanti). A questo punto, Ubuntu server minimized è installato sulla nostra VM e pronto all'uso.

6.1.3 Installazioni preliminari sulla VM host

Installazione LXC e dipendenze

```
$ sudo apt update
$ sudo apt install lxc bridge-utils net-tools -y
```

Verifica se LXC è configurato correttamente

```
$ lxc-checkconfig
```

Se lo è, darà in output tutte flag verdi con scritto "Enabled".

Creazione container

Ora creiamo i container che conterranno i router virtuali del nostro AS. Avranno tutti sistema operativo Ubuntu 20.04 LTS a 64 bit.

```
$ sudo lxc-create -t download -n ir1a -- -d ubuntu -r focal -
a amd64
$ sudo lxc-copy -n ir1a -N ir1b
$ sudo lxc-copy -n ir1a -N ir1c
$ sudo lxc-copy -n ir1a -N br1
```

Verifichiamo che i container siano visibili:

```
$ lxc-ls -f
```

Se lo sono, ci darà come output l'elenco dei nomi dei container con alcune caratteristiche.

Creazione di bridge virtuali sulla VM host

Per le 4 VLAN interne, andiamo a creare 4 rispettivi bridge virtuali modificando il file `”/etc/netplan/50-cloud-init.yaml”` in questo modo:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens33:
      dhcp4: true

  bridges:
    vlan1:
      dhcp4: no
      parameters:
        stp: true
        forward-delay: 0

    vlan2:
      dhcp4: no
      parameters:
        stp: true
        forward-delay: 0

    vlan3:
      dhcp4: no
      parameters:
        stp: true
        forward-delay: 0

    vlan4:
      dhcp4: no
      parameters:
        stp: true
        forward-delay: 0

    br-bgp:
      dhcp4: no
      parameters:
        stp: true
        forward-delay: 0

    br-external:
      interfaces: [ens33]
      dhcp4: true
      parameters:
        stp: true
        forward-delay: 0
```

Figura 6.10: Configurazione bridge virtuali

e applichiamo la configurazione:

```
$ sudo netplan apply
```

Una volta creati, li attiviamo:

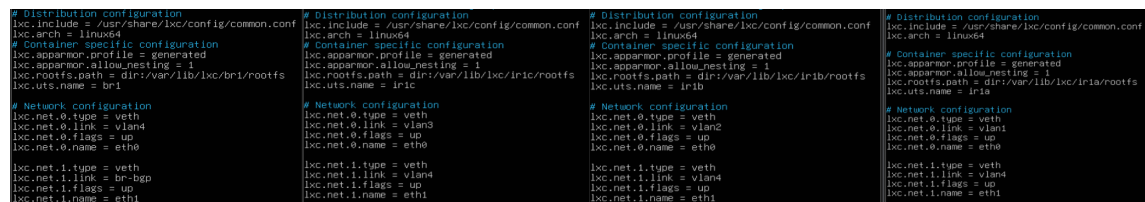
```
$ sudo ip link set vlan1 up
$ sudo ip link set vlan2 up
$ sudo ip link set vlan3 up
```

```
$ sudo ip link set vlan4 up
$ sudo ip link set br-bgp up
```

e assegnamo le interfacce dei container ai bridge. Per ogni container andiamo a fare il comando:

```
$ sudo nano /var/lib/lxc/<nome-container>/config
```

e li configuriamo come nell'immagine sottostante:



The figure shows four screenshots of the configuration file for different LXC containers. Each screenshot displays the same configuration structure, with the following key settings highlighted:

- Distribution configuration:**
 - `lxc.include = /usr/share/lxc/config/common.conf`
 - `lxc.arch = linux64`
- Container specific configuration:**
 - `lxc.apparmor.profile = generated`
 - `lxc.apparmor.allow_nesting = 1`
 - `lxc.rootfs.path = dir:/var/lib/lxc/<nome-container>/rootfs`
 - `lxc.uts.name = <nome-container>`
- Network configuration:**
 - `lxc.net.0.type = veth`
 - `lxc.net.0.link = vian<id>` (where <id> is 3, 2, 1, or 4 for the respective containers)
 - `lxc.net.0.flags = up`
 - `lxc.net.0.name = eth0`
 - `lxc.net.1.type = veth`
 - `lxc.net.1.link = br-bgp`
 - `lxc.net.1.flags = up`
 - `lxc.net.1.name = eth1`

Figura 6.11: Configurazione interfacce

Avvio dei container

Per ogni container andiamo ad eseguire:

```
$ sudo lxc-start -n <nome-container>
```

E per controllare che tutti siano partiti correttamente:

```
$ sudo lxc-ls -f
```

Configurazione della rete nei router

Per prima cosa entriamo nel container che rappresenta il nostro router (ad esempio: ir1a)

```
$ sudo lxc-attach -n ir1a
```

Poi andiamo a modificare il file "10-lxc.yaml" di netplan ¹. Questo file conterrà la configurazione di rete per i bridge virtuali che verranno usati dai container LXC.

¹Netplan è un utility di configurazione di rete.

```
$ sudo nano /etc/netplan/10-lxc.yaml
```

E lo modifichiamo in questo modo:

```
network:
  version: 2
  ethernets:
    eth0:
      addresses: [172.16.32.1/19]
    eth1:
      addresses: [172.16.128.2/19]
    eth2:
      dhcp4: yes
```

Poi per gli altri router abbiamo rispettivamente:

- **ir1b**: 172.16.64.1, 172.16.128.3
- **ir1c**: 172.16.96.1, 172.16.128.4
- **br1**: 10.0.0.2 (esterna per BGP), 172.16.128.5 (interna)

Abilitiamo la connessione a internet a tutti i container

Per scaricare i software necessari per rendere i nostri container dei router virtuali, è necessario che essi riescano ad accedere ad internet, per cui:

- Nella VM host, apriamo il file `/etc/netplan/50-cloud-init.yaml` e creiamo il bridge `br-external`:

```

network:
  version: 2
  renderer: networkd
  ethernet:
    ens33:
      dhcp4: true

  bridges:
    br-external:
      interfaces: [ens33]
      dhcp4: true
      parameters:
        stp: true
        forward-delay: 0

```

Figura 6.12: Configurazione rete VM host

Poi salviamo la configurazione:

```
sudo netplan apply
```

- Abilitiamo il NAT sulla VM host:

```
sudo nano /etc/sysctl.conf
```

e decommentiamo "net.ipv4.ip_forward=1"

- Creiamo poi una regola NAT:

```
sudo iptables -t nat -A POSTROUTING -o br-external -j
MASQUERADE
```

e facciamo in modo che si salvi anche al reboot:

```
sudo apt install iptables-persistent
sudo netfilter-persistent save
```

- Aggiungiamo il bridge "br-external" nella configurazione di rete del container. Apriamo il file "/var/lib/lxc/<nome-container>/config" e aggiungiamo:

```
lxc.net.2.type = veth
lxc.net.2.link = br-external
lxc.net.2.flags = up
lxc.net.2.name = eth2
```

e poi riavviamo il container

```
lxc-stop -n ir1a
lxc-start -n ir1a
```

- Accediamo al container e diamo una configurazione dhcp all'interfaccia:

```
lxc-attach -n ir1a
dhclient eth2
```

Ora il container `ir1a` è abilitato ad accedere a internet (possiamo testare con un semplice *ping google.com*), ripetiamo questi passaggi per tutti gli altri container e possiamo andare avanti.

6.1.4 Installazione FRR sui container

Ora vediamo come installare FRR sui container ai router:

- Aggiorniamo l'indice dei pacchetti disponibili sui repository:

```
sudo apt update
```

- Installiamo 3 tool che ci saranno utili: `curl`², `gnupg2`³ e `lsb-release`⁴

```
sudo apt install -y curl gnupg2 lsb-release
```

- Ora, siccome FRR non è in un repository predefinito di linux, dobbiamo:

- scaricare una chiave pubblica GPG di FRR con `curl`

²Curl è un tool utile per scaricare file o interagire con API web da riga di comando.

³Gnupg2 fornisce un'implementazione del software di crittografia OpenPGP, usato per verificare l'autenticità dei pacchetti.

⁴Tool che fornisce informazioni sulla distribuzione Linux in uso.

- rendere questa chiave utilizzabile dal sistema Linux con "gpg --dearmor"
- installarla in un posto sicuro dove apt (il gestore dei pacchetti Ubuntu) la cercherà

```
curl -s https://deb.frrouting.org/frr/keys.asc | sudo gpg
--dearmor -o /usr/share/keyrings/frr.gpg
```

- Ora aggiungiamo una nuova voce di repository per FRR alla configurazione del gestore di pacchetti apt.

```
echo "deb [signed-by=/usr/share/keyrings/frr.gpg] https
://deb.frrouting.org/frr/ $(lsb_release -cs) frr-
stable" | sudo tee /etc/apt/sources.list.d/frr.list
```

- Aggiorniamo nuovamente l'indice dei pacchetti disponibili sui repository:

```
sudo apt update
```

- E finalmente installiamo FRR e i suoi strumenti python:

```
sudo apt install frr frr-pythontools -y
```

- Per testare se tutto l'installazione è avvenuta correttamente, il seguente comando deve mostrare la versione di FRR installata:

```
vtysh -c "show version"
```

6.1.5 Installazione GoBGP sui container dei Border Router

Invece, per installare GoBGP sui container dedicati ai BR:

- Aggiorniamo l'indice dei pacchetti disponibili sui repository:

```
sudo apt update
```

- Installiamo i pacchetti wget⁵ e unzip⁶:

⁵Tool da riga di comando per scaricare file da internet

⁶Tool da riga di comando per decomprimere file compressi in formato .zip.

```
sudo apt install wget unzip -y
```

- Scarichiamo con wget l'ultima versione dell'archivio zip contenente l'eseguibile di GoBGP, prendendolo dal repository Github ufficiale del progetto.

```
wget https://github.com/osrg/gobgp/releases/download/v3.37.0/gobgp_3.37.0_linux_amd64.tar.gz
```

- Decomprimiamo il file .tar.gz:

```
tar -xzf gobgp_3.37.0_linux_amd64.tar.gz
```

- Spostiamo gli eseguibili "gobgp" e "gobgpd" estratti dall .tar.gz nella directory "/usr/local/bin/", in modo da poterli eseguire da qualsiasi posizione del terminale:

```
sudo mv gobgp gobgpd /usr/local/bin/
```

- Infine controlliamo se abbiamo fatto tutto correttamente, controllando la versione di GoBGP:

```
gobgp --version
```

6.1.6 Configurazione OSPF nei router

Per ogni router appartenente all'AS, andiamo ora a configurare OSPF in questo modo:

- Avviamo il container:

```
lxc-start -n <nome-router>
```

- Accediamo al container:

```
lxc-attach -n <nome-router>
```

- Accediamo alla console del router:

```
vytysh
```

- Entriamo nella modalità di configurazione avanzata:

```
conf t
```

- Abilitiamo il processo OSPF:

```
router ospf
```

- Selezioniamo l'interfaccia preposta per la vla4:

```
interface eth1
```

- Colleghiamo l'interfaccia di rete alla backbone OSPF (area 0):

```
ip ospf area 0
```

- Usciamo dalla modalità di configurazione avanzata:

```
exit
```

- E salviamo la configurazione:

```
write
```

- Per vedere se i vicini ospf sono corretti:

```
show ip ospf neighbor
```

- Per vedere se i percorsi ospf sono corretti:

```
show ip route ospf
```


Capitolo 7

Prospettive future: SDN e BGP

7.1 Cos'è la SDN

7.1.1 Architettura e principio di separazione control/data plane

7.1.2 Vantaggi principali: flessibilità, programmazione, automazione

7.2 Integrazione tra SDN e BGP

7.2.1 Routing interdominio gestito centralmente

7.2.2 Esempi di progetti o framework (es. SDX, BGP-SDN)

7.3 Prospettive evolutive

7.3.1 Reti programmabili e scenari futuri

7.3.2 Possibili impatti su sicurezza e gestione globale

Riferimenti bibliografici

- [1] Frrouting official documentation, 2025. Accessed: 2025-07-04. URL: <https://docs.frrouting.org/>.
- [2] Frrouting project - overview, 2025. Accessed: 2025-07-04. URL: <https://frrouting.org/>.
- [3] Gobgp api and configuration reference, 2025. Accessed: 2025-07-04. URL: <https://osrg.github.io/gobgp/docs/reference/>.
- [4] Gobgp documentation, 2025. Accessed: 2025-07-04. URL: <https://osrg.github.io/gobgp/>.
- [5] Cloudflare, Inc. What is an autonomous system?, n.d. Accessed: 2025-07-05. URL: <https://www.cloudflare.com/learning/network-layer/what-is-an-autonomous-system/>.
- [6] Docker. Lxc vs docker, 2013. Accessed: 2025-07-06. URL: <https://www.docker.com/blog/lxc-vs-docker/>.
- [7] John Hawkinson and Tony Bates. Guidelines for creation, selection, and registration of an autonomous system, 1996. RFC 1930, IETF. URL: <https://datatracker.ietf.org/doc/html/rfc1930>.
- [8] Internet Assigned Numbers Authority. IANA number resources, 2024. Accessed July 5, 2025. URL: <https://www.iana.org/numbers>.
- [9] IONOS. Software di virtualizzazione a confronto: i migliori strumenti per creare macchine virtuali, 2024. Ultimo accesso: 5 luglio 2025.

URL: <https://www.ionos.it/digitalguide/server/configurazione/software-di-virtualizzazione-a-confronto/>.

- [10] J. Mitchell and J. G. Scudder. Autonomous system (as) reservation for private use, 2011. RFC 6996, IETF. URL: <https://datatracker.ietf.org/doc/html/rfc6996>.

Ringraziamenti

Grazie a tutti