

Lezione 1: Software Defined Networking e IMUNES

Claudio Ardagna, Patrizio Tufarolo – Università degli Studi di Milano

Insegnamento di Laboratorio di Reti di Calcolatori



Introduzione

- ▶ Concetti di base del Software Defined Networking
- ▶ Emulazione, simulazione, virtualizzazione
- ▶ OpenVSwitch
- ▶ Commandistica Linux
- ▶ Presentazione del software IMUNES
- ▶ Esempio di una rete basilare emulata con IMUNES
- ▶ Esempio di una rete emulata con OpenStack

Terminologia - 1

- ▶ ISO/OSI (Open System Interconnection)
- ▶ Ethernet
- ▶ IP
- ▶ ARP

Terminologia - 1

- ▶ ISO/OSI (Open System Interconnection)
 - ▶ Standard de iure che organizza l'architettura di una rete di calcolatori in una struttura composta da 7 livelli (stack di rete)
- ▶ Ethernet
 - ▶ Famiglia di tecnologie standardizzate per le reti che definisce specifiche tecniche per i livelli 1 e 2 (fisico e datalink) dello stack ISO/OSI
- ▶ IP
 - ▶ Protocollo di interconnessione di reti utilizzato a livello 3 dello stack ISO/OSI, nato per connettere reti eterogenee
- ▶ ARP
 - ▶ Address Resolution Protocol, protocollo di risoluzione degli indirizzi, associa il MAC address (Livello 2) al corrispondente indirizzo IP (Livello 3)

Terminologia - 2

- ▶ Hub
- ▶ Switch
- ▶ Router

Terminologia - 2

▶ Hub

- ▶ Dispositivo per l'organizzazione di una rete con topologia a stella a livello fisico (1) dello stack ISO/OSI

▶ Switch

- ▶ Dispositivo per l'organizzazione di una rete con topologia a stella a livello datalink (2) dello stack ISO/OSI
- ▶ Implementa intelligenza e meccanismi di segregazione (VLAN) e separa i domini di collisione

▶ Router

- ▶ Dispositivo per l'organizzazione e l'interconnessione di reti disomogenee a livello IP (3) dello stack ISO/OSI
- ▶ Implementa le logiche di instradamento dei pacchetti

Terminologia - 3

▶ SDN

- ▶ Software defined networking – approccio software per la realizzazione e la gestione di topologie di rete complesse
- ▶ Permette l'amministrazione della rete mediante l'astrazione dei concetti di alto livello, separando il control plane (che controlla le decisioni su come i dati vengono instradati) e il data plane (la logica di forwarding dei pacchetti)
- ▶ Il meccanismo più usato è OpenFlow
- ▶ Software per l'implementazione e la gestione di switch software-oriented sono
 - ▶ OpenVSwitch (utilizzato da IMUNES)
 - ▶ OpenContrail

Simulazione, virtualizzazione, emulazione

- ▶ Hardware reale:
 - ▶ Vantaggi: veloce, accurato, percezione reale di ciò che avviene
 - ▶ Svantaggi: estremamente costoso, difficile da riconfigurare e da mantenere, cambiare, aggiornare
- ▶ Simulatore – simula il comportamento di hardware reale
 - ▶ Vantaggi: economico, flessibile
 - ▶ Svantaggi: non accurato, nessuna percezione reale di ciò che avviene
- ▶ Emulatore – emula il comportamento esatto di hardware reale
 - ▶ Vantaggi: flessibile, accurato
 - ▶ Svantaggi: occorre trovare compromessi in termini di prestazioni per la macchina che emula, potrebbe diventare molto costoso (emulazione hw)
- ▶ Virtualizzatore – virtualizza il comportamento esatto di hardware reale
 - ▶ Vantaggi: riproduzione esatta ed accurata di uno scenario
 - ▶ Svantaggi: estremamente costoso

Software defined networking – 1

► Situazione attuale:

- Funzioni di networking totalmente gestite da dispositivi hardware dedicati
- Protocolli e implementazione proprietarie del vendor
- Interfacce (CLI e GUI) proprietarie del vendor
- Scarsa possibilità di automazione dei processi con lo scripting
- Fornitura dei servizi di networking lenta

► Soluzione: Virtualizzazione dello stack di rete per garantire

- Efficienza
- Indipendenza dalle soluzioni hardware
- Elasticità

Software defined networking - 2

► Obiettivi

- ▶ Integrazione migliore con i processi aziendali e IT
- ▶ Paradigma «tell the network what you need to do, ask the network what you need to know»
- ▶ Astrazione delle funzionalità di rete tramite API

SDN – Control Plane

- ▶ Prende decisioni su dove mandare il traffico
- ▶ I pacchetti possono essere marcati quindi come
 - ▶ "Destinati a"
 - ▶ "Generati localmente" (dallo switch stesso)
- ▶ Le funzioni del control plane includono la gestione, la configurazione e lo scambio delle informazioni nelle tabelle di routing
- ▶ Il controller scambia le informazioni sulla topologia di rete con altri router e costruisce una tabella di routing basata su un protocollo di scambio (per esempio RIP, OSPF, BGP)
- ▶ I pacchetti del control plane sono processati dal router per aggiornare la tabella di routing

SDN – Data Plane

- ▶ Anche noto come Forwarding Plane
- ▶ Effettua il vero e proprio forwarding del traffico verso il nodo successivo lungo il percorso per la rete di destinazione selezionata, in base alle politiche del control plane
- ▶ I pacchetti passano attraverso il router
- ▶ I router e gli switch usano ciò che è stato stabilito nel control plane per gestire i flussi di frame e pacchetti

Open vSwitch - 1

- ▶ Open vSwitch (OVS), è un software che implementa uno switch virtuale distribuito multilayer
- ▶ Lo scopo principale è quello di fornire uno switch per gli ambienti di virtualizzazione, mantenendo il supporto per numerosi protocolli e standard usati nelle reti di calcolatori
- ▶ Tra i protocolli supportati ricordiamo lo Spanning Tree Protocol (STP) e il supporto a 802.1q per il partizionamento della rete a livello 2 con VLAN e trunking

Open vSwitch - 2

- ▶ Può operare sia come switch implementato in software sugli hypervisor (per gestire scenari di virtualizzazione) che come gestore del control plane in scenari bare-metal (switch hardware)
- ▶ Può essere utilizzato con protocolli per la gestione dei flussi (OpenFlow, Cisco NetFlow etc.)

Open vSwitch - 3

- ▶ È utilizzato in combinazione con molti hypervisor, ambienti di virtualizzazione (XEN, KVM, Proxmox, VirtualBox) e cloud (OpenStack, oVirt)
- ▶ Esistono anche alternative a Open vSwitch, come OpenContrail
- ▶ Lo vedremo all'opera all'interno del simulatore di rete IMUNES

La riga di comando Unix - 1

- ▶ Unix utilizza una command-line interface (CLI), ovvero un'interfaccia a riga di comando
- ▶ La riga di comando funziona grazie a un interprete, che offre dei comandi built-in all'utente.
- ▶ I comandi non built-in sono in realtà dei programmi veri e propri, pertanto sono salvati su disco e hanno un loro path
 - ▶ Affinché i binari possano essere chiamati senza specificare il loro path assoluto, è necessario che la directory che li contiene sia specificata all'interno della variabile di ambiente PATH

La riga di comando Unix - 2

- ▶ Gli interpreti a riga di comando più diffusi sono sh, bash e ksh
- ▶ Esistono altri interpreti, più o meno versatili, come zsh o fish
- ▶ Per il corso utilizzeremo bash, l'interprete più diffuso nel sistema operativo unix-like Linux

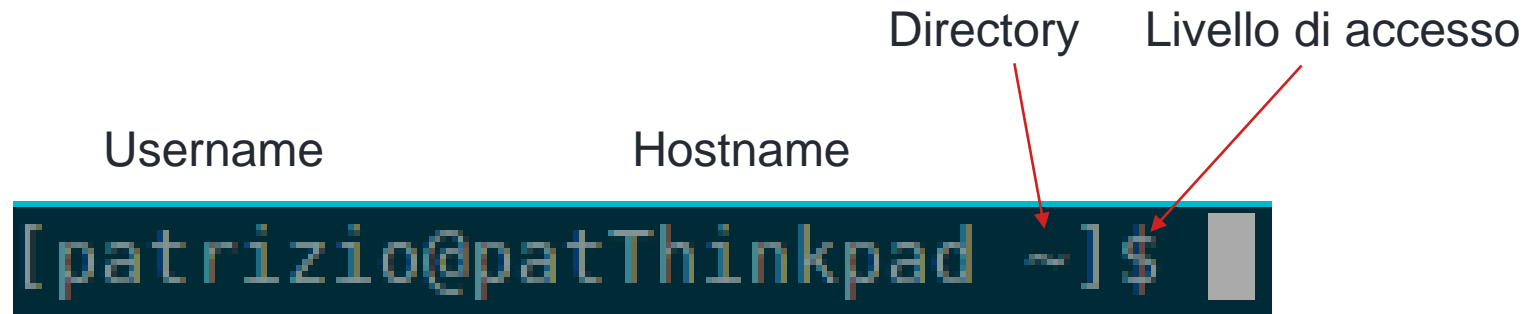
La riga di comando Unix - 3

- ▶ In bash ogni comando è preceduto da un prompt che ci fornisce informazioni su:
 - ▶ Nome dell'utente che esegue il comando
 - ▶ Hostname della macchina
 - ▶ Directory corrente
 - ▶ Livello di accesso con il quale il comando viene eseguito
 - ▶ \$ -> utente , # -> superutente
- ▶ Ogni comando, per essere eseguito, deve essere seguito dalla pressione del tasto Invio
- ▶ Una serie di comandi può costituire uno script
 - ▶ Lo script può essere memorizzato in un file e reso eseguibile in modo da comportarsi come un binario, assegnando ad esso il flag di esecuzione

La riga di comando Unix – esempio di prompt

Username Hostname Directory Livello di accesso

```
[patrizio@patThinkpad ~]$
```



Commandistica di base - 1

- ▶ Aiuto (comando built-in)
 - ▶ Help – offre all'utente la lista di alcuni comandi comuni
- ▶ Manuale (comando binario)
 - ▶ `man comando` - offre all'utente il manuale (manpage) per un dato comando
 - ▶ `Man man` – offre all'utente il manuale per il comando `man`.
 - ▶ Per uscire da `man` premere il tasto `q`. Per cercare l'occorrenza di una parola premere il tasto `/`, per navigare all'interno del manuale usare le frecce. Per tutto il resto, c'è `man man`.
 - ▶ Una versione ridotta della manpage, viene generalmente (ma non sempre!) fornita tramite l'argomento `-h/--help`.

Commandistica di base - 2

▶ Directory traversal

- ▶ `cd ..` - cambia directory passando al livello superiore
- ▶ `cd /path/assoluto` -
- ▶ `cd ./pathrelativo`
- ▶ Nota: `.` sta per directory corrente, mentre `..` sta per directory di livello superior

▶ Directory listing

- ▶ `ls` – restituisce all'utente la lista dei file
- ▶ `ls -a` – restituisce all'utente la lista dei file compresi i file nascosti (in Unix i file nascosti hanno il nome che inizia con un punto, es `.file`)
- ▶ `ls -l` – restituisce una lista dettagliata dei file
- ▶ `ls -lh` – restituisce una lista dettagliata utilizzando una notazione human readable per la dimensione dei file (converte la dimensione in KB, MB, GB etc.)

Commandistica di base - 3

▶ File reading and writing

- ▶ cat – sta per concat, consente di leggere il contenuto di un file. Tramite tecniche di redirectione dell'input e dell'output, consente anche di scrivere su un file
- ▶ head – ottiene le prime righe di un file
- ▶ tail – ottiene le ultime righe di un file
 - ▶ tail -f - autoaggiorna l'output al cambiamento del file, utile per osservare i flussi di log
- ▶ grep – effettua la ricerca di una stringa in un file
- ▶ sed – permette di applicare delle espressioni regolari su un file
- ▶ awk – è un vero e proprio linguaggio di programmazione per la manipolazione di file di testo

Commandistica di base - 4

- ▶ Redirezione dell'input e dell'output
 - ▶ comando > nomefile – scrive lo standard output del comando sul file chiamato nomefile, sostituendo il contenuto
 - ▶ comando >> nomefile – scrive lo standard output del comando sul file nomefile, effettuando l'append del contenuto (non sovrascrivendo)
 - ▶ comando < nomefile – passa tramite standard input il contenuto di nomefile al comando
 - ▶ Comando1 | comando2 – pipe: lo standard output del comando “comando1” diventa lo standard input del comando “comando2”. I Pipe possono essere concatenati a piacimento.

Commandistica di base - 5

▶ Gestione dei permessi

- ▶ `chmod [permesso] nomefile` : permette di cambiare i permessi associati a un dato file o directory. Tramite l'argomento `-r`, il comando viene eseguito ricorsivamente su tutti i file e le sottocartelle. I permessi devono essere passati in forma ottale o con la notazione `[u|g|o][+|-][r|w|x]`
- ▶ `chown [utente][:gruppo] nomefile`: permette di modificare il proprietario o il gruppo di un file/directory.
- ▶ `chgrp gruppo nomefile`: permette di modificare il gruppo di un file/directory

▶ Altri strumenti utili

- ▶ `nano` – editor di testo semplice molto facile da utilizzare
- ▶ `vi` / `vim` – editor di testo più complessi ma anche più potenti

Esercizio

- ▶ Creare una cartella nella propria home folder
- ▶ Creare un file vuoto all'interno della cartella chiamato «esempio»
- ▶ Aprire il file con un editor di testo (da terminale) a piacere
- ▶ Scrivere nel file la frase «Ciao, questo è un file di configurazione»
- ▶ Utilizzare il comando echo con redirectione dell'output per scrivere la frase «Questa è la seconda linea» senza sovrascrivere il contenuto
- ▶ Tramite il comando grep ottenere esclusivamente la prima linea e salvarla nel file «esempio2»



Commandistica di rete per Linux (netlink) - 1

- ▶ `ip link` – consente di visualizzare lo stato dei link a livello 2 definiti nel sistema. Ogni link corrisponde a un'interfaccia fisica o definita in software
 - ▶ `ip link show dev eth0`
 - ▶ Mostra le informazioni di livello 2 per la periferica eth0
 - ▶ `Ip link set up dev eth0`
 - ▶ Attiva il link per l'interfaccia eth0
 - ▶ `Ip link set down dev eth0`
 - ▶ Disattiva il link per l'interfaccia eth0
 - ▶ `Ip link set mtu 9000 dev eth0`
 - ▶ Imposta l'MTU a 9000 per l'interfaccia eth0
 - ▶ `Ip link set promisc on dev eth0`
 - ▶ Attiva la modalità promiscua per l'interfaccia eth0

Commandistica di rete - 2

- ▶ `Ip addr` – consente di visualizzare lo stato delle interfacce a livello 3
 - ▶ `Ip addr show dev eth0`
 - ▶ Mostra le informazioni di livello 3 per la periferica eth0
 - ▶ `Ip addr add 10.0.0.1/24 dev eth0`
 - ▶ Aggiunge l'indirizzo 10.0.0.1 con netmask /24 (notazione cidr) al device eth0
 - ▶ `Ip addr add 10.0.0.1 netmask 255.255.255.0 dev eth0`
 - ▶ Aggiunge l'indirizzo 10.0.0.1 con netmask 255.255.255.0 (notazione ip) al device eth0
 - ▶ `ip addr del 10.0.0.1/24 dev eth0`
 - ▶ Rimuove l'indirizzo 10.0.0.1/24 al device eth0

Commandistica di rete – 3 – ARP Table

▶ Ip neigh

▶ `Ip neigh add 10.0.0.1 lladdr 1:2:3:4:5:6 dev eth0`

- ▶ Aggiunge alla tabella ARP una entry che associa il MAC 1:2:3:4:5:6 all'indirizzo 10.0.0.1 per la rete gestita dal device eth0

▶ `Ip neigh del 10.0.0.1 lladdr 1:2:3:4:5:6 dev eth0`

- ▶ Invalida dalla tabella ARP la entry tra 1:2:3:4:5:6 e il device eth0

▶ `Ip neigh replace 10.0.0.2 lladdr 1:2:3:4:5:6 dev eth0`

- ▶ Sostituisce la entry della tabella ARP per 1:2:3:4:5:6 con l'indirizzo 1.0.0.2

Commandistica di rete – 4 – Routing table

- ▶ **Ip route**
 - ▶ **ip route add default via 192.168.1.1 dev em1**
 - ▶ Aggiunge una default route (instrada tutto il traffico che non rispetta nessun'altra regola) con gateway 192.168.1.1 raggiungibile sul device em1
 - ▶ **ip route add 192.168.1.0/24 via 192.168.1.1**
 - ▶ Aggiunge una route verso 192.168.1.0/24 tramite il gateway 192.168.1.1
 - ▶ **ip route add 192.168.1.0/24 dev em1**
 - ▶ Aggiunge una route verso la rete 192.168.1.0/24 che può essere raggiunta tramite il device em1
 - ▶ **ip route delete 192.168.1.0/24 via 192.168.1.1**
 - ▶ Elimina la route verso la rete 192.168.1.0/24 tramite gateway 192.168.1.1
 - ATTENZIONE: se ci stanno altri instradamenti verso quella rete, ma tramite un altro gateway, questi non vengono rimossi!
 - ▶ **ip route get 192.168.1.4**
 - ▶ Mostra la regola di routing seguita per raggiungere 192.168.1.4

Commandistica di rete – 5 - Namespaces

- ▶ Un namespace di rete è una copia logica dello stack di rete all'interno del Sistema operativo, che ha dispositivi di rete proprietari, regole di instradamento proprietarie e così via
 - ▶ Ogni processo, di default, eredita il namespace del processo padre; inizialmente tutti ereditano il namespace predefinito del processo init
- ▶ Ip netns
 - ▶ Ip netns add <name>
 - ▶ Crea un nuovo namespace
 - ▶ Ip netns delete <name>
 - ▶ Elimina un namespace esistente
 - ▶ Ip netns list
 - ▶ Restituisce la lista dei namespace (sono memorizzati in /var/run/netns)
 - ▶ Ip netns exec <namespace> <command>
 - ▶ Esegue un processo/comando, all'interno di uno specific namespace

Commandistica di rete – 6 – Altri comandi utili

▶ Arping

▶ `arping -I eth0 192.168.1.1`

- ▶ Invia una richiesta ARP per l'indirizzo 192.168.1.1 sull'interfaccia eth0

▶ `arping -D -I eth0 192.168.1.1`

- ▶ Controlla l'esistenza di un MAC address duplicato per l'indirizzo 192.168.1.1 sull'interfaccia eth0

▶ Sockstat / Netstat

▶ Sono due comandi (alternativi) che permettono di visualizzare lo stato delle socket sul sistema

▶ `netstat -plnt46`

- ▶ Mostra tutti i processi che usano socket in ascolto sulla macchina, sia per IPv4 che per IPv6

Commandistica OpenVSwitch - 1

- ▶ Il funzionamento di OpenVSwitch è gestito tramite questi quattro comandi:
 - ▶ `ovs-vsctl` : Utilizzato per configurare lo switch
 - ▶ `ovs-ofctl` : Utilizzato per monitorare e amministrare i flussi OpenFlow
 - ▶ `ovs-dpctl` : Utilizzato per amministrare il data plane di OpenVSwitch
 - ▶ `ovs-appctl` : Utilizzato per gestire i demoni basilari di OpenVSwitch
- ▶ Nell'ambito del corso utilizzeremo per il momento esclusivamente il comando `ovs-vsctl`

Commandistica OpenVSwitch - 2

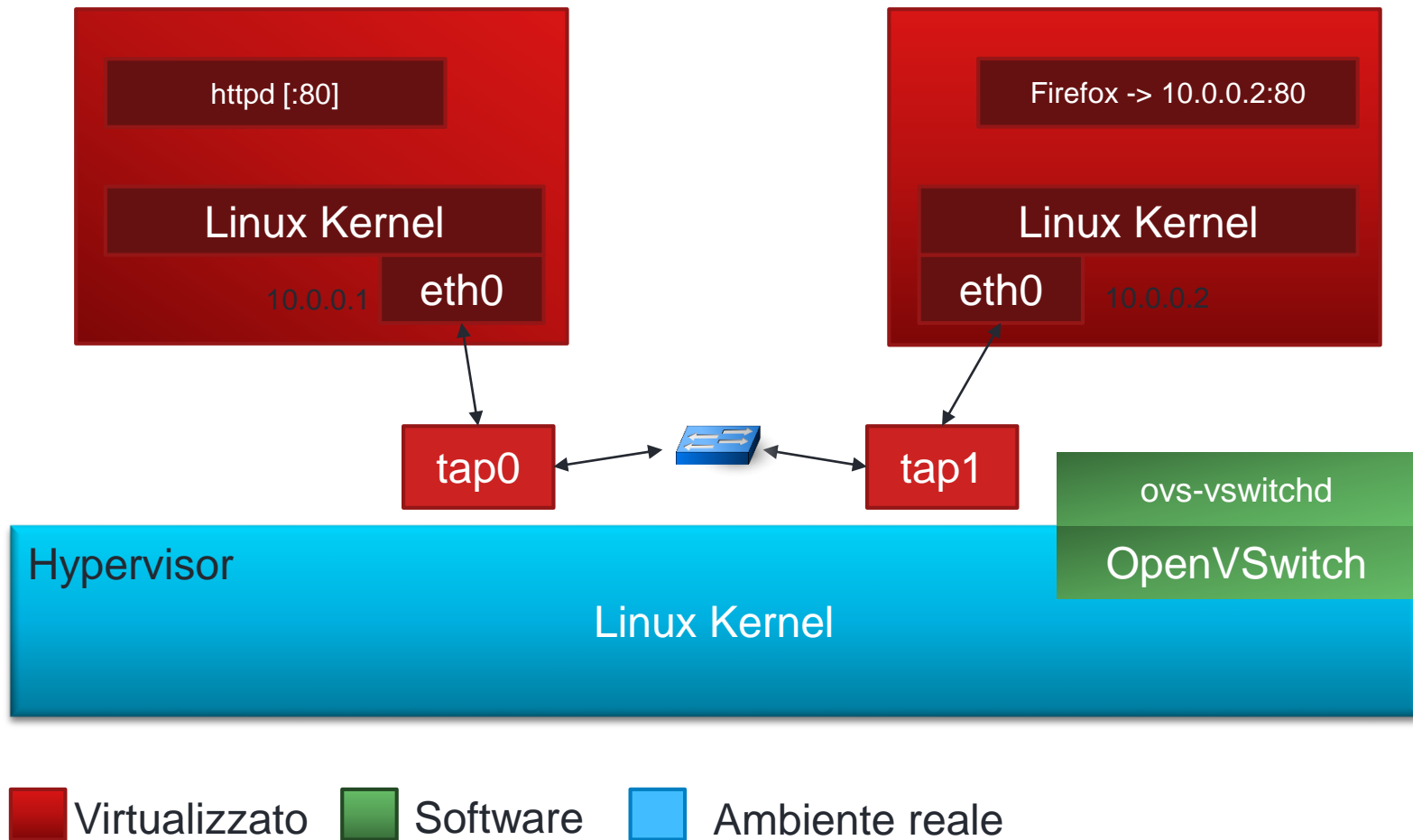
- ▶ Ovs-vsctl
 - ▶ ovs-vsctl show
 - ▶ Stampa un breve riepilogo della configurazione dello switch
 - ▶ ovs-vsctl list-br
 - ▶ Stampa la lista dei bridge configurati
 - ▶ ovs-vsctl list-ports <bridge>
 - ▶ Stampa la lista delle porte su un determinato bridge
 - ▶ ovs-vsctl list interface
 - ▶ Stampa una lista delle interfacce

Commandistica OpenVSwitch - 3

▶ Ovs-vsctl

- ▶ `ovs-vsctl add-br <bridge>`
 - ▶ Crea un bridge
- ▶ `ovs-vsctl add-port <bridge> <interface>`
 - ▶ Aggancia un'interfaccia (fisica o virtuale) a uno specifico bridge
- ▶ `ovs-vsctl add-port <bridge> <interface> tag=<VLAN number>`
- ▶ `sudo ovs-vsctl set port <interface> tag=<VLAN number>`
 - ▶ Associa ad una porta un determinato VLAN tag. Attenzione: tutte le porte di OVS sono porte di trunk!
- ▶ `ovs-vsctl set interface <interface> type=patch options:peer=<interface>`
 - ▶ Utilizzato per creare patch per connettere due o più bridge insieme.

Rete SDN in un ambiente virtualizzato

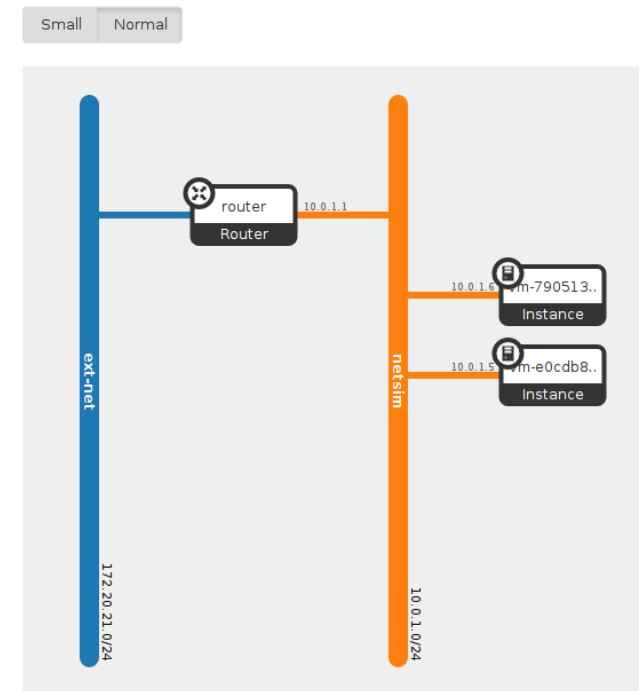


OpenStack – Realizzazione della topologia di rete con OpenStack

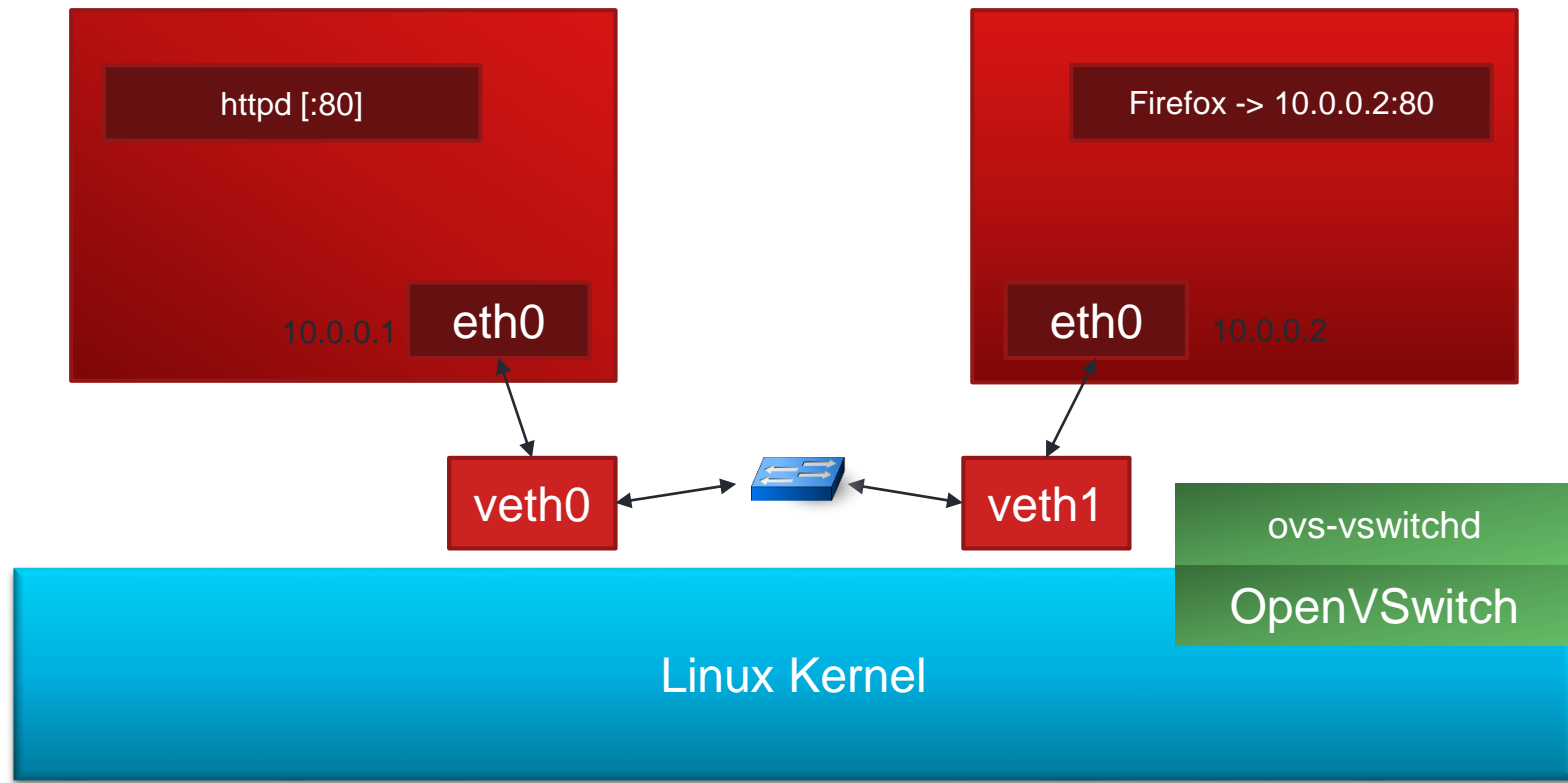
<input type="checkbox"/>	Instance Name	Image Name	IP Address	Size	Key Pair	Status
<input type="checkbox"/>	Vm-79051330-bd3c-40c8-8f2e-b9f0105f3433	cirros-0.3.3-x86_64	10.0.1.6	m1.tiny	pat	Active
<input type="checkbox"/>	Vm-e0cdb83e-6de6-4cde-8c30-68e92be2474d	cirros-0.3.3-x86_64	10.0.1.5	m1.tiny	pat	Active

Displaying 2 items

Network Topology



Rete SDN in un ambiente simulato



Simulato



Software



Ambiente reale



Realizzazione della topologia di rete con Linux

- ▶ `#ip netns add host1`
- ▶ `#ip netns add host2`
- ▶ `#ovs-vsctl add-br s1`

- ▶ `#ip link add h1-eth0 type veth peer name s1 s1-eth1`
- ▶ `#ip link add h2-eth0 type veth peer name s1 s1-eth2`
- ▶ `#ip link set h1-eth0 netns h1`
- ▶ `#ip link set h2-eth0 netns h2`
- ▶ `#ip netns exec h1 ifconfig h1-eth0 10.0.0.1`
- ▶ `#ip netns exec h2 ifconfig h2-eth0 10.0.0.2`
- ▶ `#ip netns exec h1 ifconfig lo up`
- ▶ `#ip netns exec h2 ifconfig lo up`

- ▶ `#ovs-vsctl add-port s1 s1-eth1`
- ▶ `#ovs-vsctl add-port s1 s1-eth2`

- ▶ `#ip netns h1 python2 -m SimpleHTTPServer 80`
- ▶ `#ip netns h2 firefox http://10.0.0.1`



IMUNES - 1

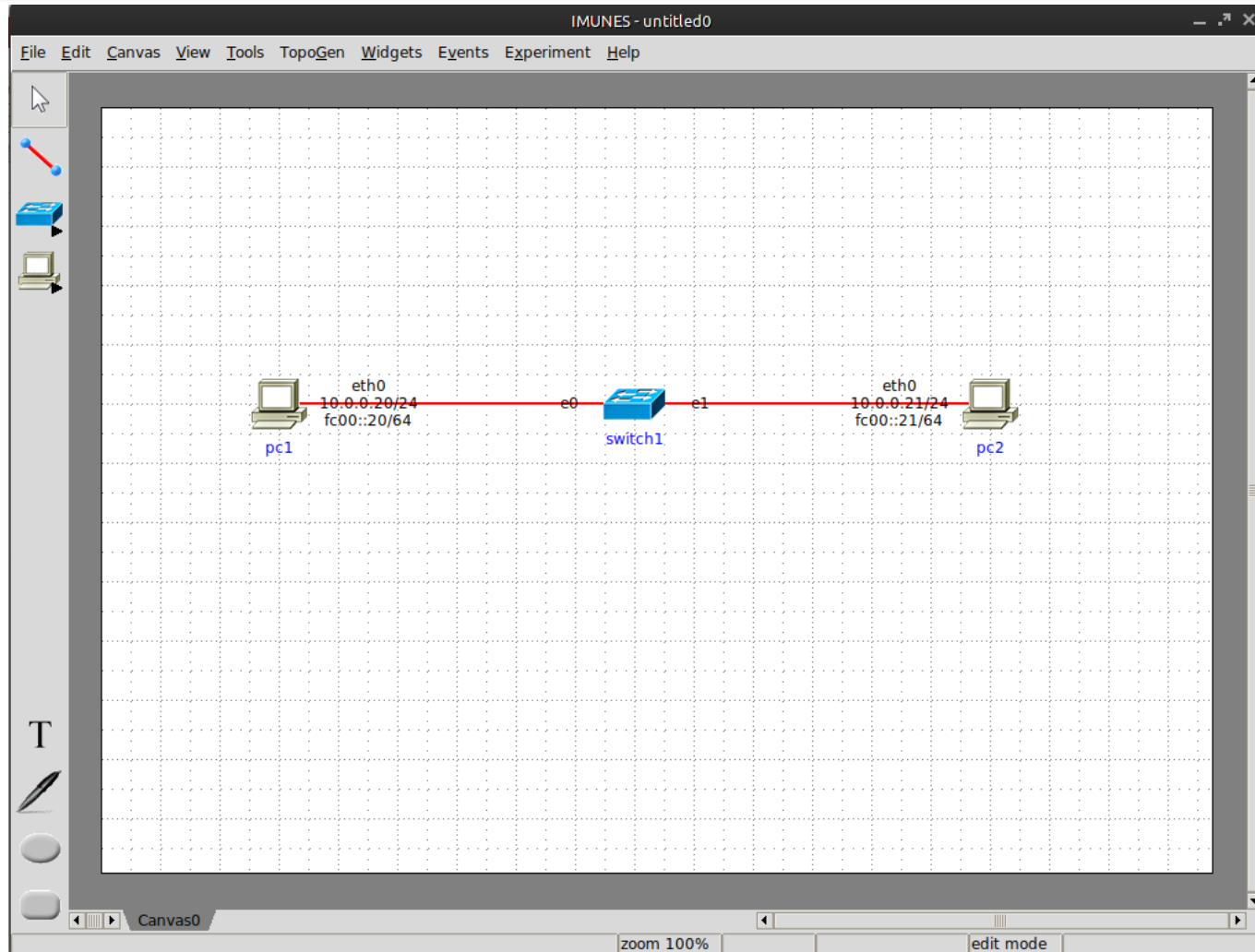
- ▶ IMUNES è un framework di emulazione/simulazione di topologie di rete che partiziona lo stack di rete del sistema operativo al fine di realizzare dei nodi virtuali, i quali possono essere interconnessi tra di loro con collegamenti simulati dal kernel, permettendo di comporre topologie di rete complesse
- ▶ Utilizza funzionalità native del kernel
- ▶ È compatibile con FreeBSD e Linux

IMUNES - 2

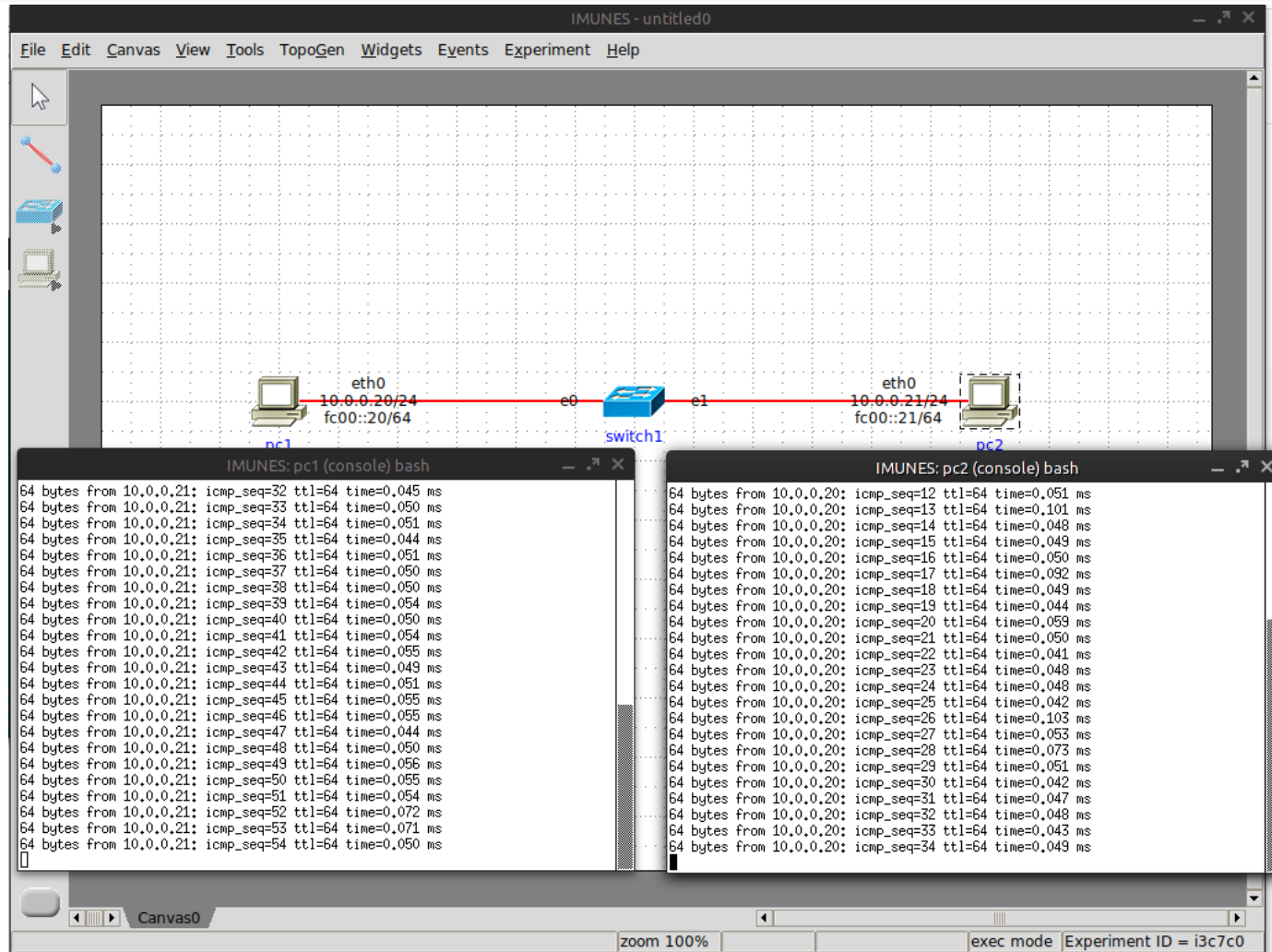
- ▶ Dispone di un'interfaccia grafica per disegnare il grafo di rete e gestire singolarmente sia i nodi dell'infrastruttura che le proprietà dei link
- ▶ Implementa la possibilità di introdurre servizi di uso comune all'interno della rete su tutti i livelli dello stack
- ▶ Per funzionare, su Linux, usa i Namespace, OpenvSwitch e il software Docker



IMUNES – Realizzazione della topologia di rete con IMUNES



IMUNES – Realizzazione della topologia di rete con IMUNES



Conclusioni - 1

- ▶ Abbiamo ripassato i concetti fondamentali delle reti di calcolatori (stack ISO/OSI, protocolli e device L2 e L3)
- ▶ Abbiamo appreso i concetti di base del Software Defined Networking
- ▶ Abbiamo compreso le differenze tra emulazione, simulazione, virtualizzazione
- ▶ Abbiamo accennato al software OpenVSwitch
- ▶ Abbiamo appreso alcuni comandi POSIX per Unix, Linux e OpenVSwitch



Conclusioni - 2

- ▶ Abbiamo imparato a realizzare una piccola rete SDN su Linux
- ▶ Abbiamo conosciuto il software IMUNES e abbiamo lanciato un esempio basilare di topologia di rete
- ▶ Abbiamo conosciuto il software OpenStack utilizzandolo per realizzare la stessa topologia di rete

