Operazione Rif. PA 2023-19410/RER approvata con DGR 1317/2023 del 31/07/2023 finanziata con risorse del Programma Fondo sociale europeo Plus 2021-2027 della Regione Emilia –Romagna.

Progetto n. 1 - Edizione n. 1

TECNICO PER LA PROGETTAZIONE E LO SVILUPPO DI APPLICAZIONI INFORMATICHE

MODULO: N. 5 Titolo: SICUREZZA DEI SISTEMI INFORMATICI E DISPIEGO DELLE APPLICAZIONI
DURATA: 21 ORE DOCENTE: MARCO PRANDINI

PROTOCOLLI E SERVIZI DI INTERNET

A cosa serve una rete

Dove viene utilizzata l'informatica esistono:

- Una molteplicità di risorse
  - di calcolo
  - di memorizzazione
  - per l'ingresso e l'uscita dei dati

  associate a stazioni di lavoro diverse
- Una continua necessità
  - di accedere alle risorse
  - di fornire o recuperare informazioni

  indipendentemente dalla stazione su cui si lavora

Connessione di calcolatori

Idea di fondo: far “parlare” tra loro due calcolatori collegandoli elettricamente.

- Vantaggi rispetto allo spostamento fisico dei dati:
  - Capacità (sempre?)
  - Trasparenza
  - Robustezza
  - Interattività
- Problemi da risolvere:
  - costruire l’infrastruttura
  - concordare un linguaggio comune
  - proteggere il sistema dalle intrusioni

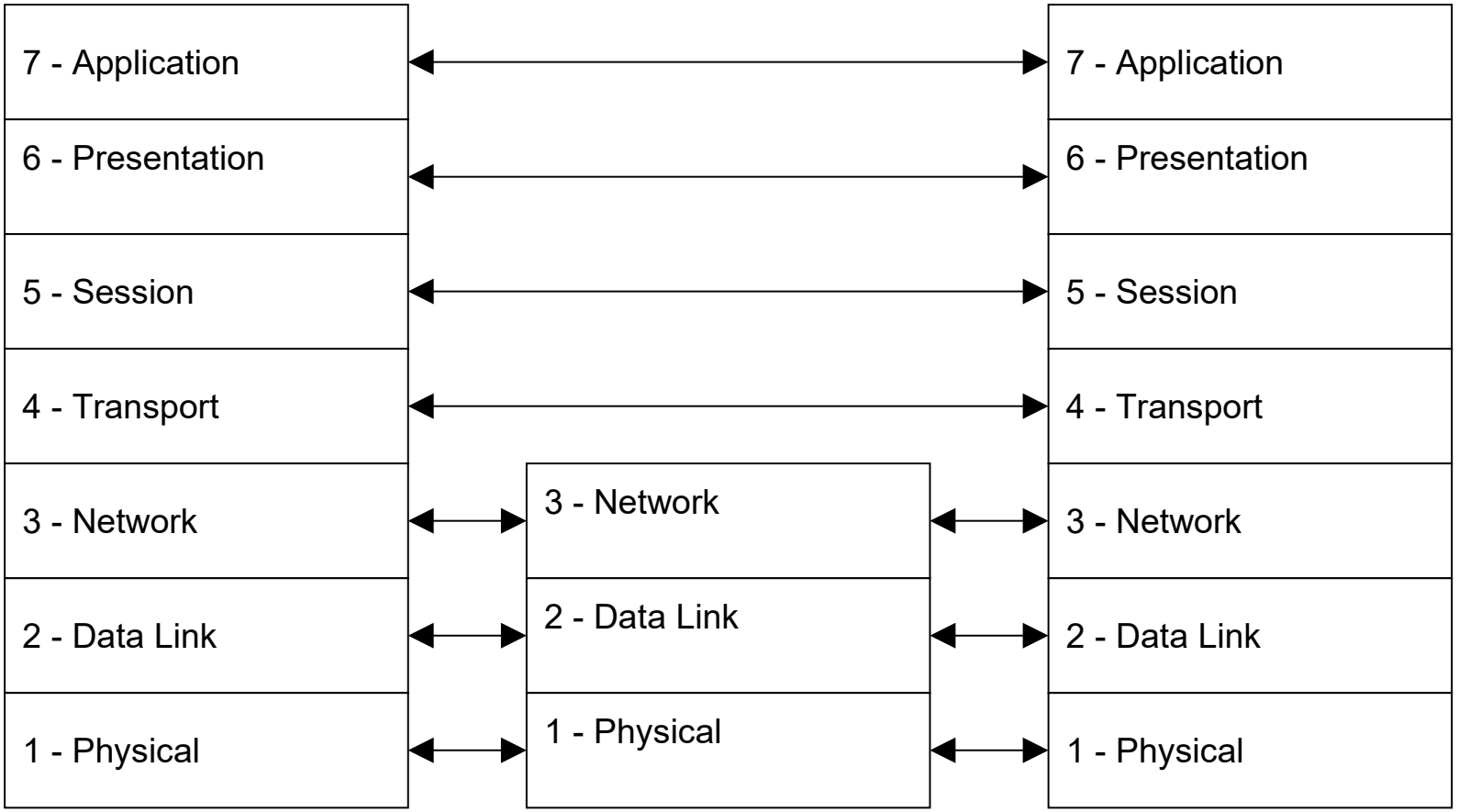
Stabilire le regole del dialogo

Perchè i calcolatori connessi si capiscano, è necessario stabilire uno standard sul formato fisico dei segnali, la temporizzazione dei messaggi, ed il significato dei simboli trasmessi dall'uno all'altro.

Se affrontato in modo monolitico, il problema risulta complesso e le soluzioni non generalizzabili.

Per questo motivo, l'ISO ha formalizzato il modello OSI, che rappresenta la comunicazione come una pila (stack) di livelli, ognuno deputato a svolgere un ruolo preciso senza che i dettagli implementativi siano noti agli altri livelli.

Modello ISO/OSI



Strati 1-2-3

- **Lo strato 1 (Physical) definisce le caratteristiche elettriche e meccaniche dei mezzi di trasporto delle informazioni.**
- **Lo strato 2 (Data Link) gestisce le comunicazioni tra sistemi connessi alla stessa rete fisica. Esegue il controllo d'errore sui dati trasportati dal livello fisico, definisce i metodi per identificare un sistema all'interno una rete locale e le politiche di accesso dei sistemi al livello fisico.**
- **Lo strato 3 (Network) si occupa dell'***instradamento (routing)***, ovvero della ricerca del percorso per connettere due sistemi che vogliono comunicare tra loro anche se sono su reti fisiche distinte. I metodi per identificare un sistema su scala globale (come gli indirizzi IP usati in Internet) sono definiti in questo strato.**

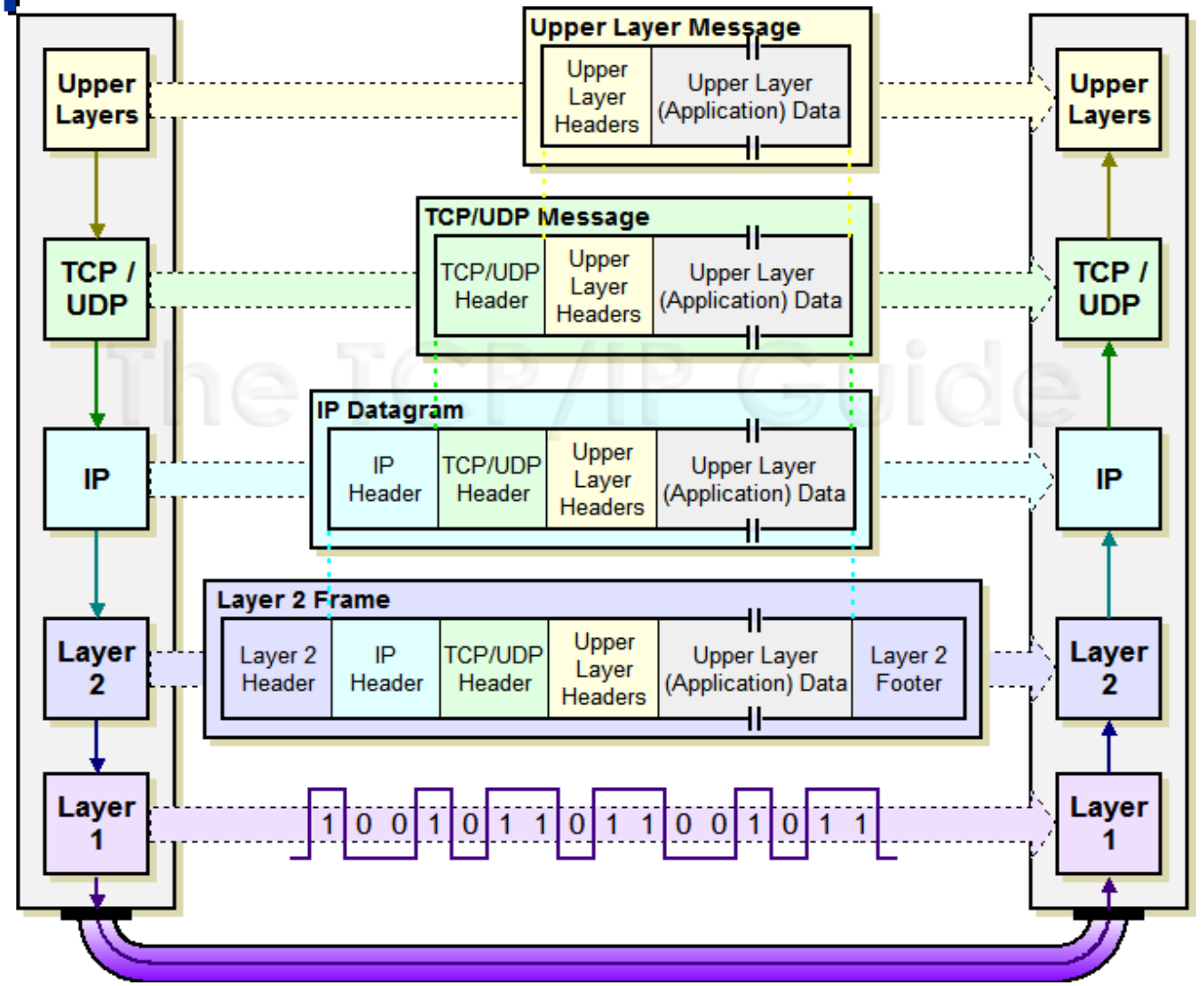
Strati 4-5-6-7

- Lo strato 4 (Transport) si occupa del trasporto dei dati *end-to-end*. Fornisce allo strato superiore canali affidabili tra i due estremi della comunicazione, occupandosi del controllo di flusso, della correzione degli errori, della divisione e ricomposizione dei dati. Nasconde la frammentazione dei percorsi "reali" che gli sono forniti dallo strato 3.
- Gli strati 5 e 6 (Session-Presentation) sono raramente implementati, e modellano gli strumenti quali le procedure di negoziazione della localizzazione, i controlli di integrità e validità dei documenti, e la sincronizzazione tra applicazioni interattive.
- Lo strato 7 (Application) è quello in cui si collocano le applicazioni che originano e ricevono i dati oggetto della comunicazione

Percorso logico e fisico dei dati

- **Fisicamente un messaggio originato da un'applicazione (strato 7) percorre il sistema di origine scendendo di strato in strato, attraverso le *interfacce*.**
  - **Ogni strato sottopone il messaggio alle elaborazioni necessarie per lo svolgimento del proprio compito, lo *incapsula* in un pacchetto arricchito di metadati in modo che il proprio omologo nel sistema di destinazione possa effettuare le operazioni inverse, e lo passa allo strato inferiore.**
- **Logicamente, durante l'interazione di due sistemi avviene uno scambio di messaggi tra ciascuno degli strati omologhi, secondo le regole stabilite da un *protocollo*.**
  - **Il vantaggio della struttura a strati risiede nella possibilità di scegliere qualsiasi protocollo per uno strato senza influenzare quelli circostanti. (Es. passaggio di Microsoft a TCP/IP)**

Incapsulazione



Reti Locali

- Una rete locale può essere definita come un insieme di calcolatori connessi da un'infrastruttura tale per cui ogni partecipante ha visibilità fisica diretta di tutti gli altri.
  - Infrastruttura basata sui soli strati 1 e 2
  - Caratterizzate da
    - *Topologia (bus, anello, stella)*
    - *Mezzo trasmissivo (coassiale, doppino, fibra, etere, ...)*
    - *Politica di accesso (token-based, collisione, ...)*
- L'identificazione di ogni nodo avviene per mezzo di un indirizzo che può avere caratteristche arbitrarie, perchè per definizione deve essere raggiungibile senza istruire intermediari --> problemi di interconnessione

Internetworking

- Usando il solo livello 2 ...
  - Come garantire l'univocità degli indirizzi a livello globale?
  - Come limitare la propagazione del traffico?
- Usando il livello 3 si sovrappone uno schema di indirizzi configurabili all'indirizzamento di livello 2
  - Gli indirizzi sono concessi, sia numericamente che organizzativamente, secondo una gerarchia
  - La contiguità numerica degli indirizzi rispecchia quella fisica dei corrispondenti nodi della rete
- Da questo momento non tratteremo più concetti generali ma faremo riferimento allo standard mondiale di fatto: TCP/IP

Internet

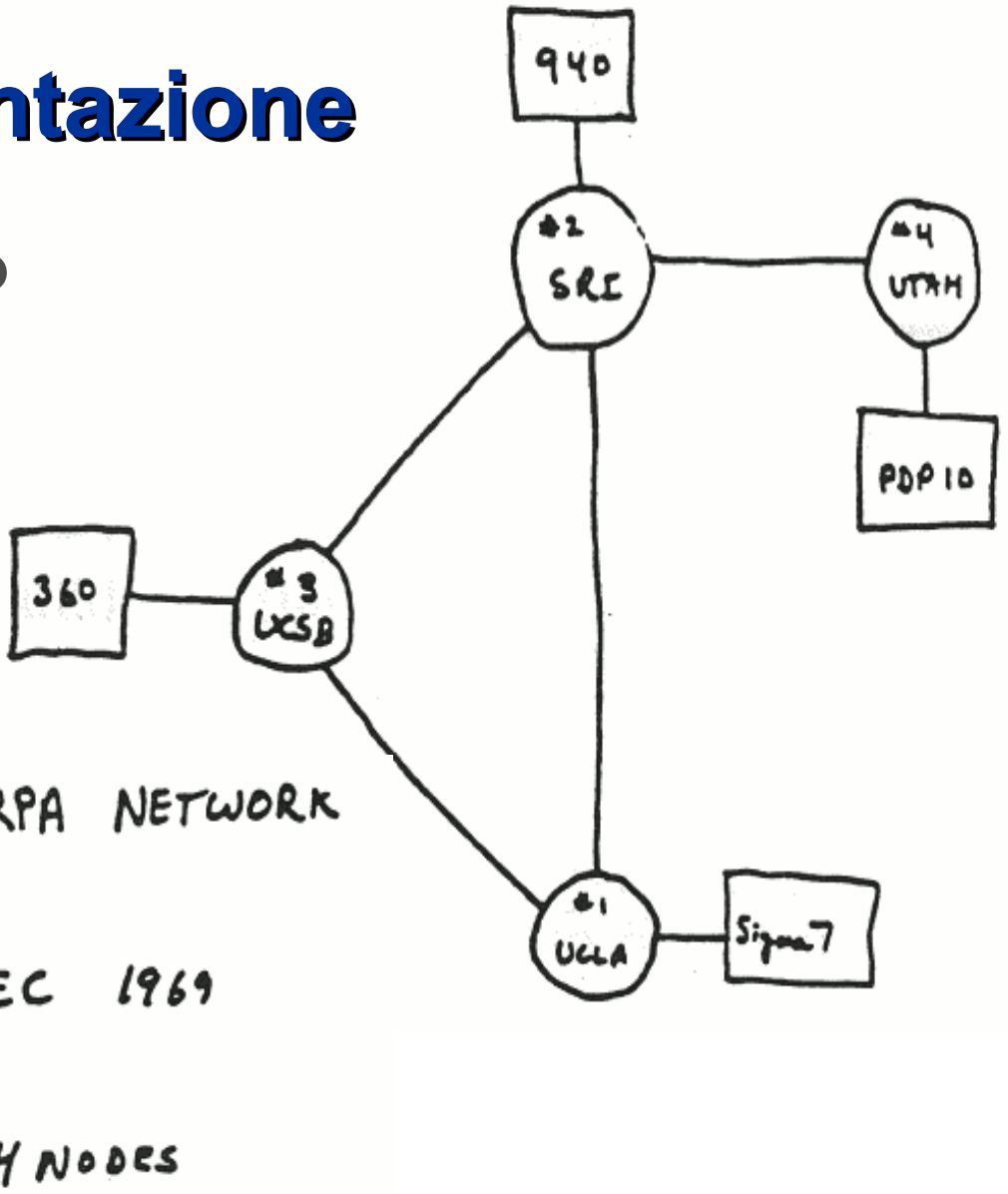
Origini di TCP/IP

Le radici di Internet affondano nella Guerra Fredda.

- Problema:
  - Un sistema di comunicazione a prova di attacco nucleare.
- Soluzione (RAND Corporation, 1964):
  - Una *rete*, anzichè un albero di interconnessioni, modello tradizionalmente utilizzato nelle telecomunicazioni
  - Moltitudine di nodi nessuno dei quali abbia un “ruolo di comando”
  - Moltitudine di connessioni tra i nodi (ridondanza dei percorsi)
  - Traffico suddiviso in pacchetti
  - La funzione di ogni nodo è il *packet forwarding*

La prima implementazione

- DARPA: Dipartimento della Difesa USA + Advanced Research Projects Agency
- BBN
- RAND Corporation



Alcune tappe fondamentali

- 1972 Invenzione della posta elettronica.
- 1973 Primi nodi internazionali (UK, Norvegia).
  Vinton Cerf pubblica TCP.
- 1983 Smilitarizzazione.
  Primo uso del termine Internet.
  Introduzione dei nomi.
- 1986 NSF collega i propri centri e le università al di fuori di
  ARPAnet (che ora conta 1000 nodi) usando TCP/IP.
- 1990 ARPAnet viene dismessa.
  I nodi TCP/IP superano le 100.000 unità.
- 1992 Nascita del World Wide Web.
  Più di 1 milione di nodi.
- 1993 Primo browser grafico (NCSA Mosaic)
  Il tasso di crescita di Internet sfiora il 350% annuo.

Organismi, standard, documentazione

L'impulso all'evoluzione di Internet è stato dato da una forma di collaborazione estremamente libera. I documenti fondamentali (Request For Comments – RFC) prodotti dal Network Working Group seguivano queste "regole" (RFC 3):

«Il contenuto di una nota del NWG può essere qualsiasi riflessione, suggerimento, o altro soggetto relativo al software per host o ad aspetti diversi della rete. È incoraggiata la sottomissione tempestiva delle note piuttosto che la cura della forma. Questioni teoriche prive di esempi o applicazioni immediate, suggerimenti specifici, tecniche di implementazione prive di spiegazioni introduttive e di contesto, domande esplicite prive di qualunque tentativo di risposta sono tutti argomenti accettabili. La lunghezza minima per una nota del NWG è di una frase.»

Organismi, standard, documentazione

- La RFC 1 “Host Software” è del 7 aprile 1969. Ora sono oltre 4400.
- Attualmente, esistono diversi organismi che regolano i vari aspetti di Internet, dalla ricerca avanzata ai dettagli applicativi:
  - IAB (Internet Architecture Board) svolge il coordinamento e decide le strategie a lungo termine
  - IRTF (Internet Research Task Force) si occupa della ricerca di base, con progetti a lungo termine
  - IETF (Internet Engineering Task Force) si occupa degli aspetti tecnici, e produce gli Internet Drafts che precorrono le RFC.
  - IANA (Internet Assigned Numbers Authority) si occupa della registrazione degli IP, degli host, e dei servizi TCP e UDP.
- Il coordinamento, le attività di promozione, finanziamento e formazione sono svolte dall’Internet Society.

TCP/IP - generalità

La sigla TCP/IP indica l'insieme dei protocolli usati in ambito Internet.

Alcune caratteristiche:

- È uno standard aperto, slegato da qualsiasi produttore
- È definito ufficialmente dall'IETF nelle Request For Comments
- È indipendente dall'hardware
- Svolge principalmente le funzioni proprie degli strati 3 e 4:
  - Prevede una modalità di indirizzamento globale
  - Supporta diverse modalità di routing
  - È basato sulla commutazione di pacchetto
- Fornisce agli strati superiori connessioni multiple e di diversi tipi

TCP/IP – schema a livelli

Lo stack TCP/IP è più semplice di quello OSI, e prevede solo 4 livelli

Application Layer	(~ 5,6,7)	Protocolli di posta, trasferimento file, web, condivisione dischi, ...
Transport Layer	(~ 4)	Servizi di multiplazione e di garanzia della completezza dei dati
Internet Layer	(~ 3)	Indirizzamento dei nodi ed instradamento dei pacchetti
Network Access Layer	(~ 1,2)	Mappatura degli indirizzi Internet sulla LAN ed accesso al mezzo

Internet Protocol

- **IP (*Internet Protocol*) è il protocollo che svolge le funzioni classificate come strato 3 dal modello OSI; fornisce un servizio di trasporto dati attraverso un'internet, mediante il meccanismo della commutazione di pacchetto:**
  - **tutti i dati consegnati ad IP dallo strato superiore vengono frammentati in unità di dimensione massima prefissata (*pacchetti*)**
  - **per ogni singolo pacchetto viene individuato un percorso che porti dal sistema sorgente al sistema di destinazione, eventualmente attraverso sistemi intermedi**
- **IP è un protocollo di tipo *best effort*, ovvero non dà garanzie:**
  - **sull'effettivo arrivo di un pacchetto**
  - **sull'ordine di arrivo dei pacchetti**
  - **sulla velocità di trasmissione o sui tempi di percorrenza**

Indirizzi IP

- **Associazione univoca indirizzo –> sistema (*host*)**
  - **(non è imposto il contrario –> *multi-homed host*)**
- **Indirizzi IPv4: 32 bit divisi in 4 byte, normalmente rappresentati con 4 numeri in base 10 separati da punti (*dotted decimal notation*)**
  - **Esempio: 137.204.59.1**
- **Ogni indirizzo fa parte di una rete *(subnet)* che inizia da un indirizzo di *network***
- **In origine l' estensione era implicita, ora è specificata da una *netmask***
  - **Una subnet logica coincide con una LAN fisica**
  - **Per instradare un pacchetto verso la destinazione non serve considerare il suo indirizzo, basta la subnet cui appartiene**

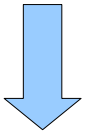
Reti class-based

Originariamente sono state definite *classi* di indirizzi, ovvero raggruppamenti da usare per i sistemi di una rete locale, in cui i byte erano rigidamente divisi tra network id e host id:

- **128 reti di Classe A (contenenti fino a 16 milioni di host circa):**
  - **indirizzi da 0.*.*.* a 127.*.*.***
- **16.384 reti di Classe B (contenenti fino a 65000 host circa):**
  - **indirizzi da 128.0.*.* a 191.255.*.***
- **2.097.152 reti di Classe C (contenenti fino a 254 host):**
  - **indirizzi da 192.0.0.* a 223.255.255.***
- **Gli indirizzi da 224.*.*.* a 239.*.*.* sono riservati al *multicast***
- **Gli indirizzi da 240.*.*.* a 255.*.*.* sono riservati a usi futuri**

Il vantaggio sulle LAN

- Host su una stessa rete locale:
  - collocati in una stessa area, servita da un determinato apparato di rete
  - numericamente identificati da indirizzi di una stessa subnet

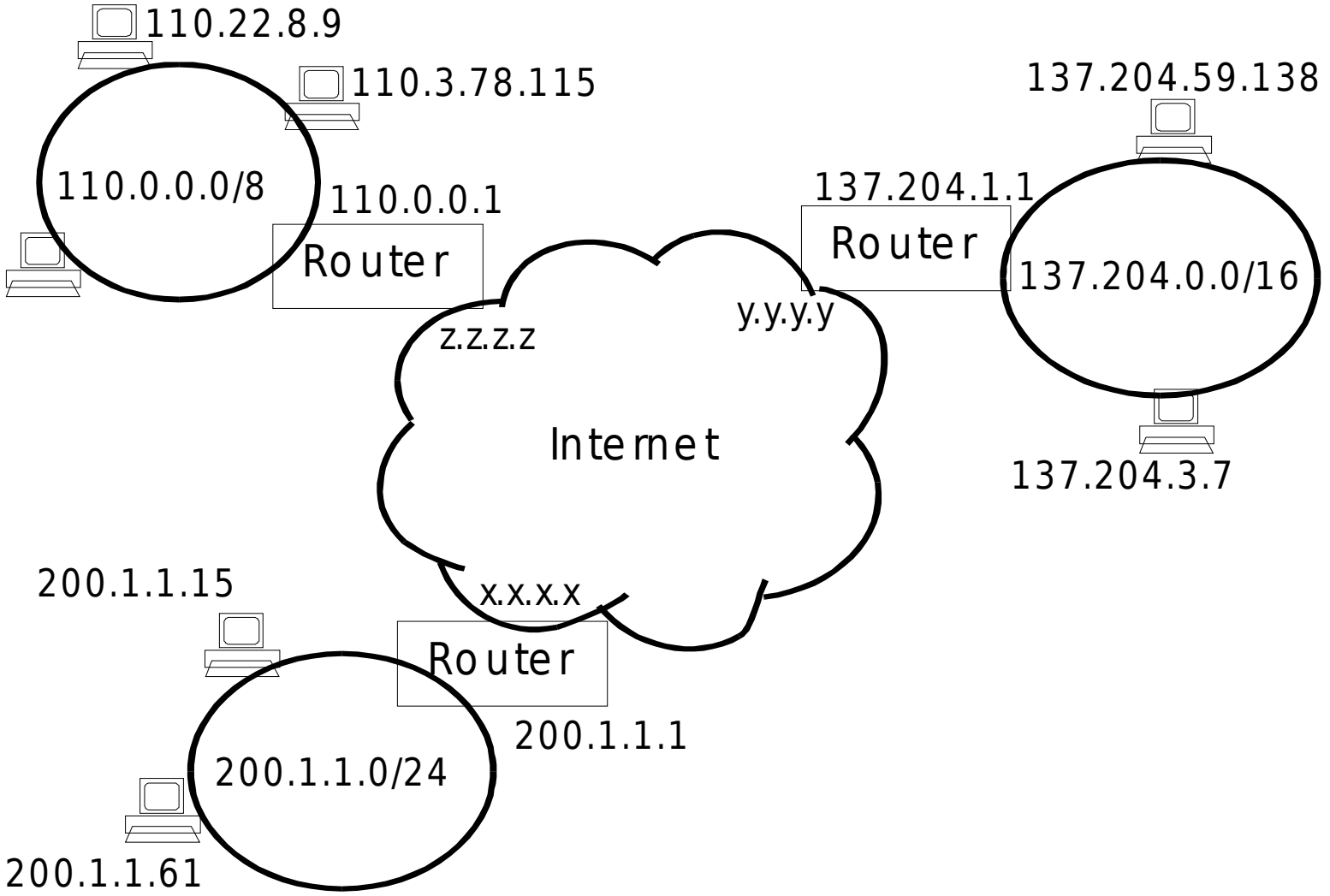


- Per raggiungerli non ho bisogno di sapere dove si trova ognuno, mi basta sapere come raggiungere la subnet

Instradamento del traffico

- Ogni host di una rete locale IP “sa” quali altri host può raggiungere direttamente, grazie a network address e netmask.
- Ogni host può essere configurato per raggiungere destinazioni al di fuori della rete locale usando degli intermediari: i *router*
  - Un router per instradare traffico tra LAN diverse deve avere un'interfaccia connessa ad ognuna, a cui deve essere associato un indirizzo della corrispondente subnet.
  - Più in generale, un router sa come raggiungere *ogni* host/subnet, direttamente o attraverso altri router
  - L'elenco delle destinazioni non è conservato da ogni router in modo estensivo, ma tipicamente contiene l'indicazione di un router a cui delegare l'instradamento verso tutte le subnet non esplicitamente note: il *default gateway*.

Internetworking con TCP/IP



CIDR

- Poche classi di dimensioni fissate = spreco di indirizzi
- Soluzione: CIDR (*classless inter-domain routing*)
- Con classless-IP gli indirizzi sono visti come una stringa di 32 bit divisa in net-id e host-id in un punto arbitrario, anzichè per byte.
  - Unico vincolo (ovvio): la dimensione di una rete è potenza di 2 (in questo esempio 2^6 = 64 indirizzi)

    144 . 156 . 166 . 151

    10010000 10011100 10100110 10010111

    26 bit net-id **6 bit host-id**
- Una rete locale è identificata per mezzo di un *network address* e di una *netmask*, noti a tutti gli host che ne fanno parte.

Netmask, network, broadcast

- Serve un modo per specificare dove cade la divisione: la *netmask* è un valore di 32 bit composto da tanti “1” quanti sono i bit che identificano la subnet, e tanti “0” quanti sono i bit che specificano l'host al suo interno
  - Nell'esempio precedente:
    11111111.11111111.11111111.11000000 = 255.255.255.192
- Due valori in ogni subnet hanno un significato speciale e non possono essere usati per indirizzare un host:
  - Network address (ottenuto mettendo a 0 tutti i bit dell’host-id)
    - 10010000.10011100.10100110.10000000 = 144.156.166.128
  - Broadcast (ottenuto mettendo a 1 tutti i bit dell’host-id)
    - 10010000.10011100.10100110.10111111 = 144.156.166.191

Netmask, network, broadcast... perchè?

- Il sistema è stato pensato per velocizzare il lavoro degli apparati di instradamento
  - ogni pacchetto contiene un indirizzo di destinazione
  - deve essere inviato verso la subnet che contiene tale indirizzo
  - i router conoscono il modo di raggiungere le subnet per mezzo di una tabella che elenca (network, netmask, interfaccia da usare)
  - l'operazione di AND bit-a-bit tra un indirizzo e una netmask restituisce il valore da confrontare con il nework
- Facile costruzione di gerarchie di subnet, per diminuire ulteriormente il numero di regole di instradamento memorizzata su ogni router

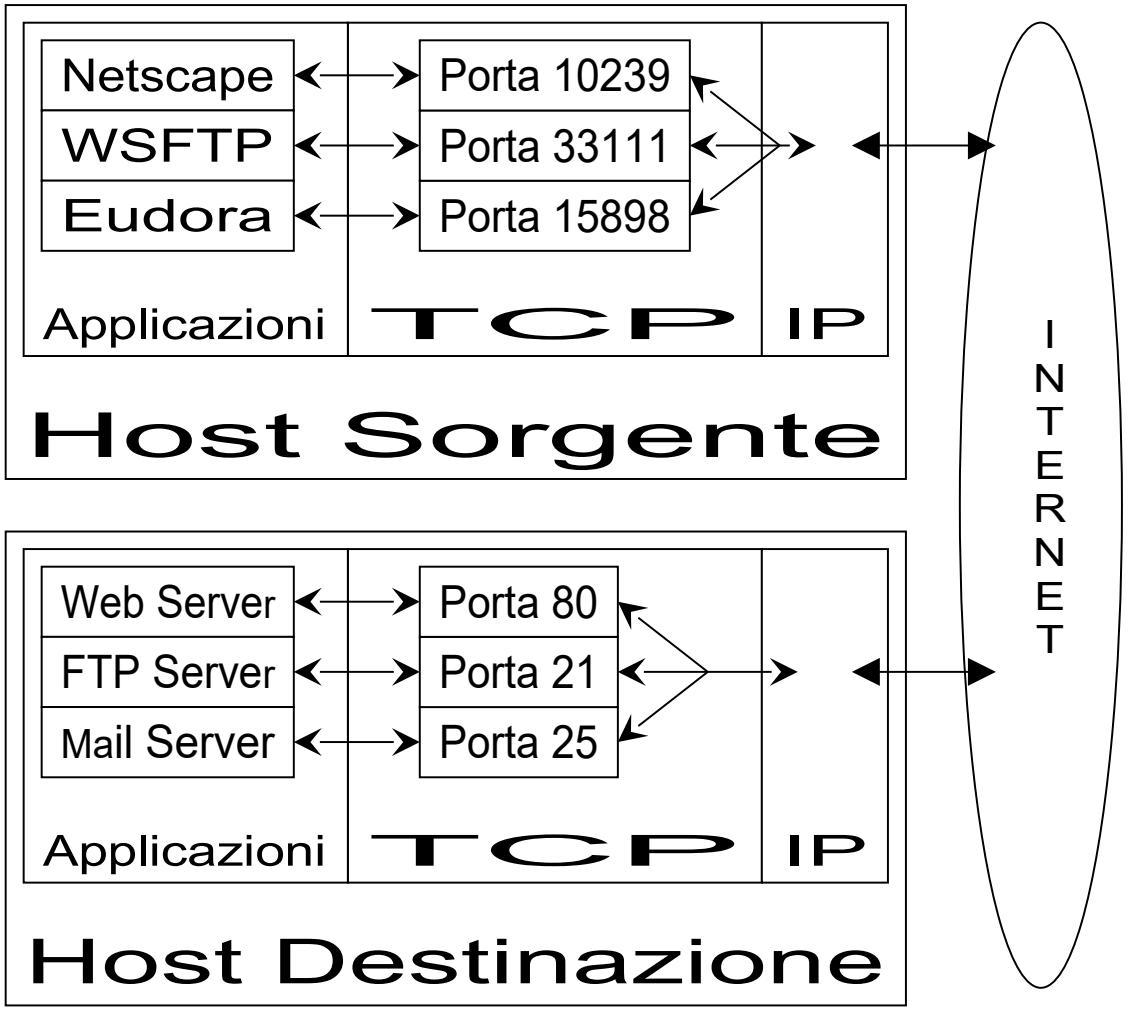
Il livello di trasporto

- TCP = Transmission Control Protocol
- UDP = User Datagram Protocol
- Le funzioni svolte da questi protocolli sono classificabili nell'ambito dello strato 4 del modello OSI. Conseguentemente, operano su di un canale di comunicazione dei dati end-to-end, fornito da IP che nasconde i dettagli della gestione dei pacchetti.
  - IP non permette di distinguere diverse applicazioni sorgente o destinazione dei pacchetti su di uno stesso host
  - IP non fornisce garanzie sulla consegna dei pacchetti

Multiplexing

- Il multiplexing è la funzione svolta sia da TCP che da UDP è per consentire l'indirizzamento di applicazioni specifiche all'interno di un host
  - IP fornisce un canale trasmissivo tra due host, TCP ed UDP permettono a più applicazioni di usare questo canale contemporaneamente-
  - Questo servizio viene offerto associando ogni applicazione ad una *porta*.
    - Una porta è indicata da un numero compreso tra 0 e 65535.
    - In ogni istante, su di un host, ciascuna porta non può essere utilizzata da più di un'applicazione.
- Per raggiungere un'applicazione su di un host, bisogna conoscere la porta associata. Per questo i servizi di ampio interesse sono collocati su porte standard, dette *well-known ports*

Multiplexing



Il livello fisico

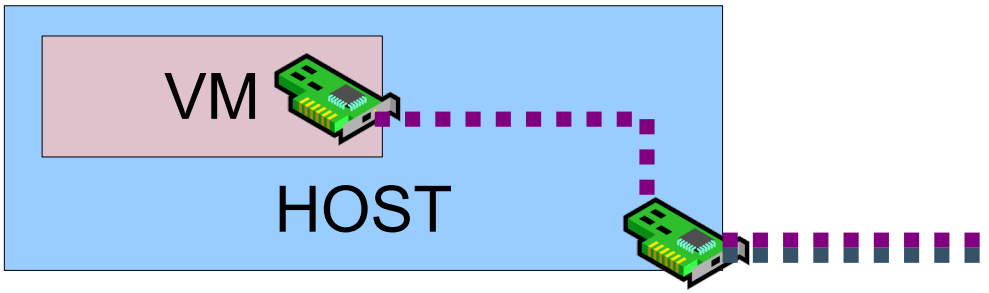
- Sul “ferro”:
  - l’interfaccia di rete è un vero e proprio dispositivo elettronico
  - il MAC address è tipicamente cablato nel dispositivo
  - la sua connessione via cavo oppure la sua associazione a una rete wireless ne definiscono l’appartenenza a una LAN
  - l’indirizzo IP va configurato (vedi seguito)
- In una VM:
  - l’interfaccia di rete è un artefatto gestito dall’hypervisor
  - il sistema operativo guest la percepisce come un dispositivo
  - l’hypervisor
    - può impostare il MAC address
    - definisce a quale segmento di rete è connessa
    - può gestire a volte l’indirizzamento logico e modalità particolari di consegna del traffico

Il livello fisico in VirtualBox

- VirtualBox permette di attivare molteplici interfacce per ogni VM
- Ogni interfaccia può essere connessa in una modalità specifica; quelle principali sono
  - NAT
  - Bridged
  - Host-only
  - Internal
- A default, è attiva una sola interfaccia in modalità NAT
  - permette di navigare attraverso l'host
  - vedremo meglio cosa significa dopo aver introdotto i concetti di instradamento

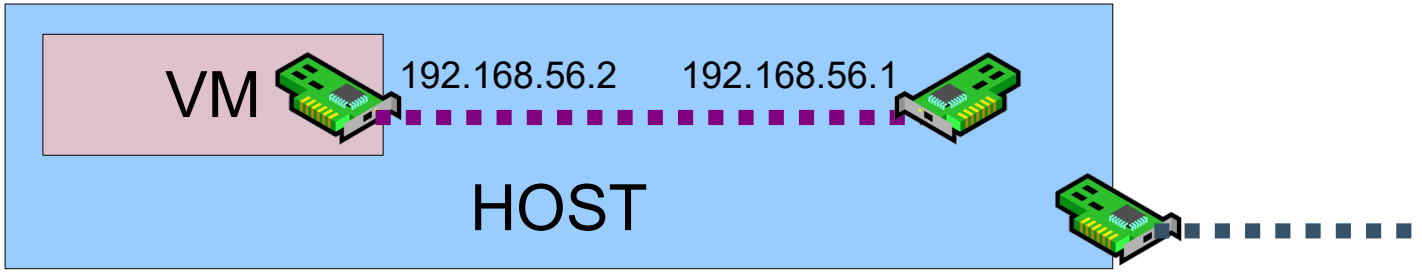
Il livello fisico in VirtualBox

- Interfacce *bridged*
  - si comportano come se fossero connesse all'interfaccia dell'host attraverso un bridge/hub, quindi è come se fossero attestate sulla stessa LAN dell'host



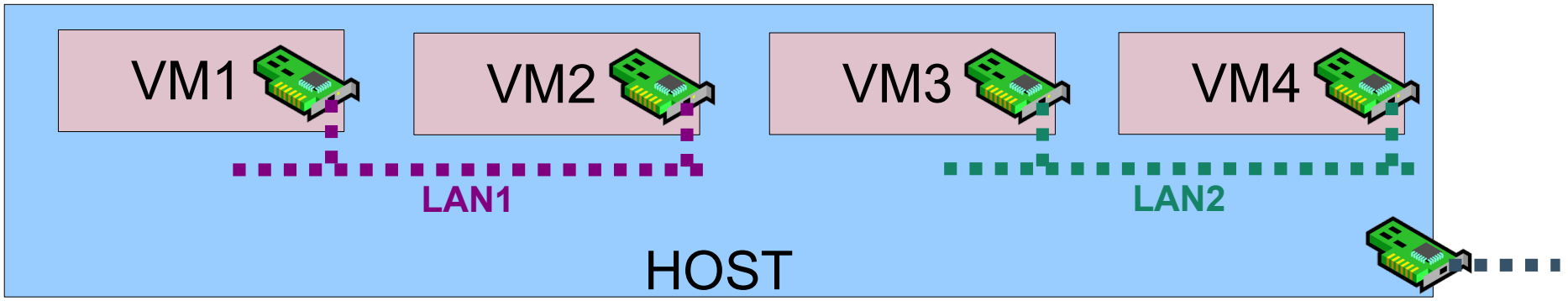
Il livello fisico in VirtualBox

- Interfacce *host-only*
  - l'hypervisor
    - viene configurato per utilizzare una specifica subnet IP
    - genera un'interfaccia virtuale sull'host e le assegna un IP della subnet
    - connette l'interfaccia della VM alla corrispondente LAN virtuale in modo che la VM possa comunicare unicamente con l'host



Il livello fisico in VirtualBox

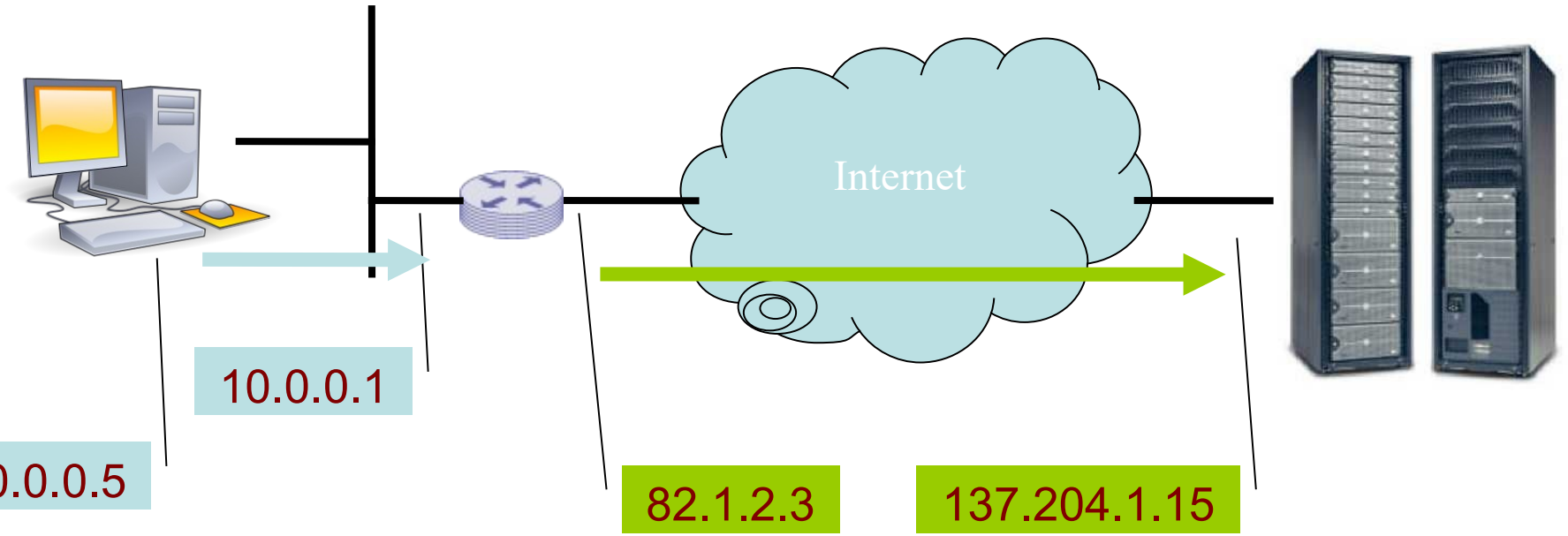
- Interfacce *internal*
  - l'hypervisor le assegna a una LAN totalmente virtuale, e fa in modo che solo le interfacce appartenenti alla stessa internal network possano comunicare tra loro



Network Address Translation (NAT)

- La prospettiva di un esaurimento degli IP disponibili ha fatto esplodere l'utilizzo di una tecnica che consente di utilizzare un solo indirizzo pubblico, senza rinunciare alla possibilità di realizzare in tecnologia TCP/IP una rete anche di grandi dimensioni e di connetterla ad Internet.
- L'efficacia della tecnica si basa sull'osservazione che la gran parte degli host è *client* e non *server*
  - non necessitano di essere raggiunti da richieste
  - originano richieste e devono poter essere raggiunti dalle risposte

Network Address Translation (NAT)



Richiesta

Source	10.0.0.5	: 34567
Destination	137.204.1.15	: 80

Default gateway: 10.0.0.1

Richiesta traslata

Source	82.1.2.3	: 34567
Destination	137.204.1.15	: 80

Network Address Translation (NAT)

- La quintupla
  (protocollo, ip_sorgente, porta_sorgente, ip_destinazione, porta_destinazione)
  identifica univocamente una connessione
- Nel NAT molti IP sorgente vengono sostituiti dall'unico IP pubblico del router
  - possibilità di modificare la porta sorgente per disambiguare le connessioni originate con tutti i parametri identici a parte l'IP sorgente
  - memorizzazione delle traslazioni per poter riconoscere il destinatario delle risposte

Source IP	Source port	Router IP	Router port	Dest. IP	Dest. port
10.0.0.5	11111	82.1.2.3	11111	137.204.1.15	80
10.0.0.9	11111	82.1.2.3	11111	137.204.1.15	80

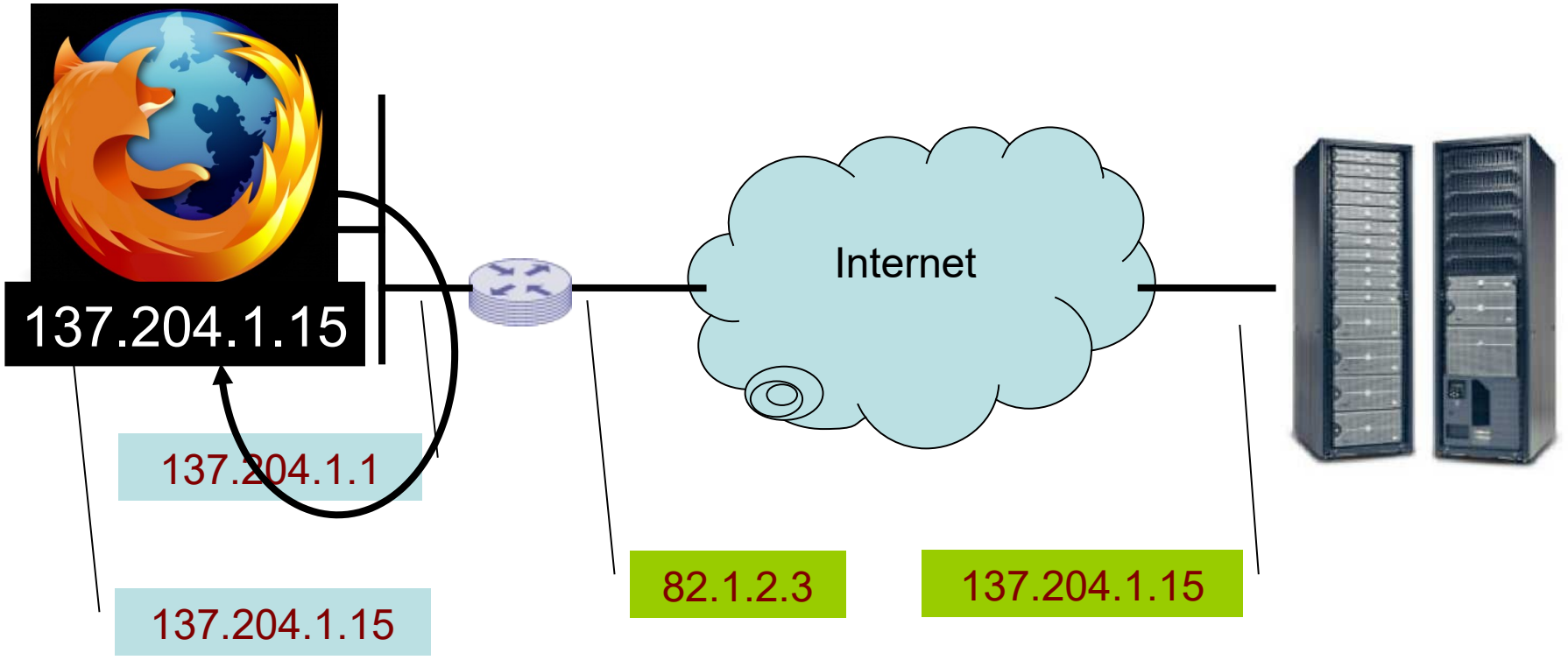
Network Address Translation (NAT)

- La quintupla
  (protocollo, ip_sorgente, porta_sorgente, ip_destinazione, porta_destinazione)
  identifica univocamente una connessione
- Nel NAT molti IP sorgente vengono sostituiti dall'unico IP pubblico del router
  - possibilità di modificare la porta sorgente per disambiguare le connessioni originate con tutti i parametri identici a parte l'IP sorgente
  - memorizzazione delle traslazioni per poter riconoscere il destinatario delle risposte

Source IP	Source port	Router IP	Router port	Dest. IP	Dest. port
10.0.0.5	11111	82.1.2.3	11111	137.204.1.15	80
10.0.0.9	11111	82.1.2.3	22222	137.204.1.15	80

Network Address Translation (NAT)

- Gli IP della rete interna risultano del tutto nascosti
- Cosa capiterebbe scegliendoli arbitrariamente?



IP privati (RFC 1918)

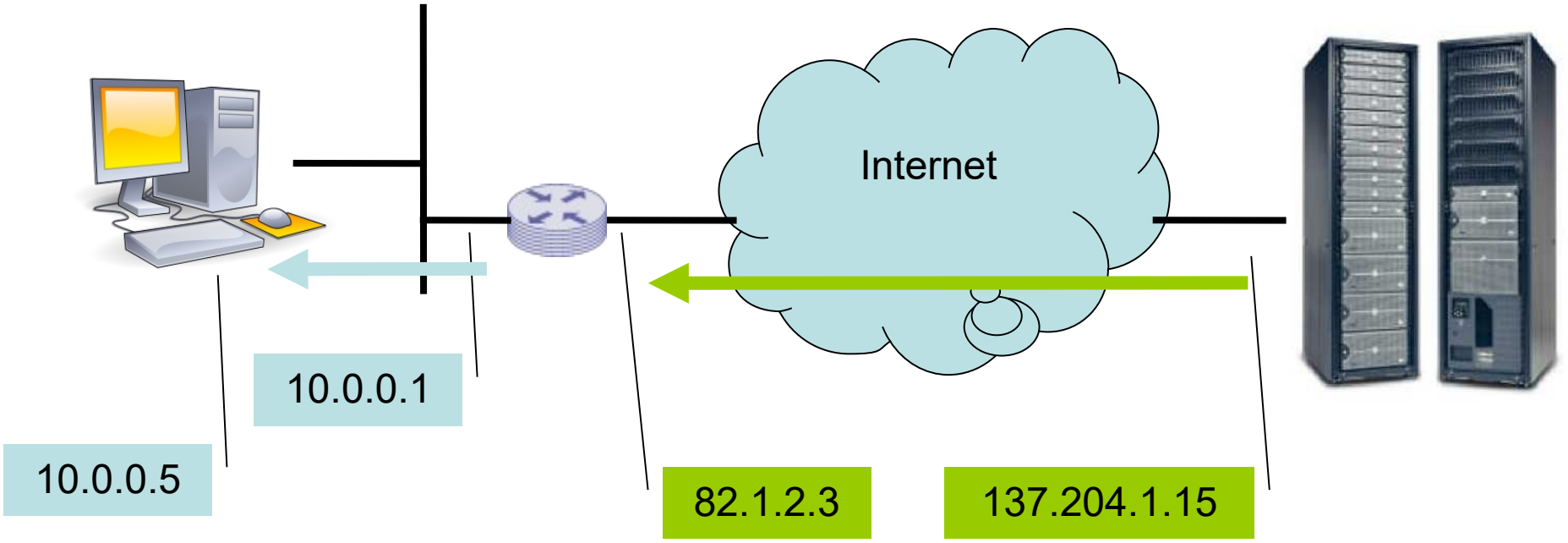
- Per evitare il problema del possibile "oscuramento" di IP validi, uno standard definisce alcuni intervalli di indirizzi che non possono essere utilizzati su Internet
  - `10.0.0.0/8`
  - `172.16.0.0/16  -- 172.31.0.0/16`
  - `192.168.0.0/24 -- 192.168.255.0/24`

SNAT / DNAT

- Per poter utilizzare una rete di client con un solo IP pubblico si modifica l'IP **sorgente**
  - Source NAT (SNAT)
  - il traffico fluisce spontaneamente attraverso il default gateway, che lo maschera: comportamento **trasparente ed automatico**

- Lo stesso dispositivo di instradamento consente anche di rendere raggiungibili dall'esterno alcuni host della rete privata, modificando l'indirizzo di destinazione quando riceve richieste su di una specifica porta del proprio IP pubblico
  - Destination NAT (DNAT)
  - la mappatura tra porta (servizio) di destinazione ed host interno a cui inoltrare la richiesta va **esplicitamente configurata**

DNAT

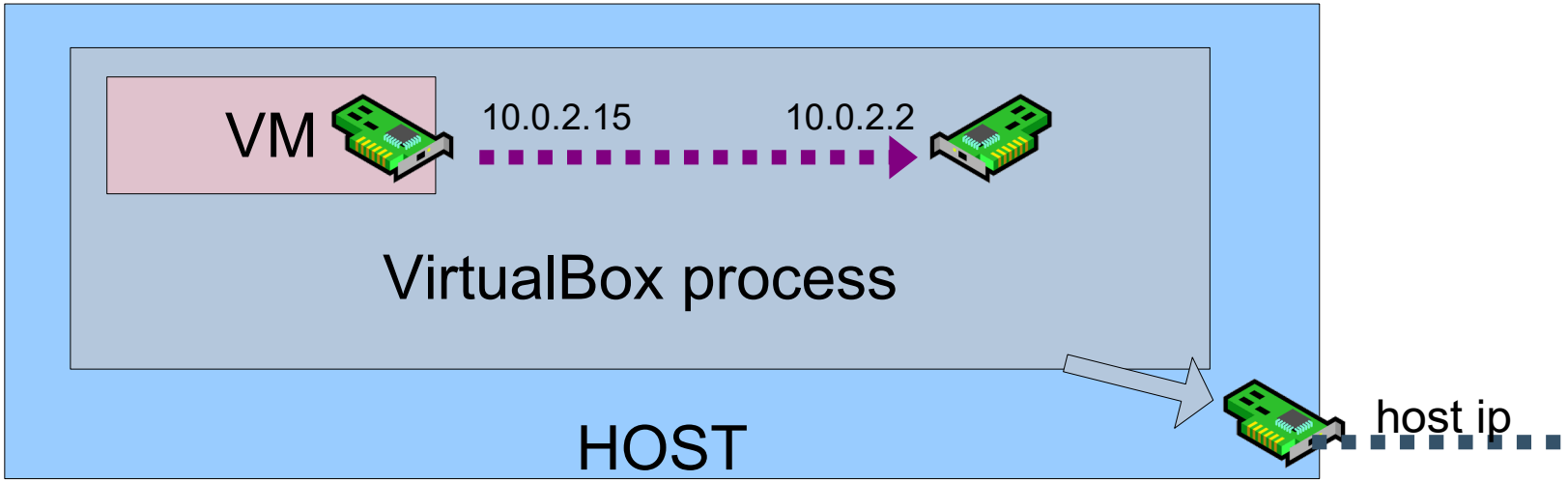


Richiesta traslata	
Source	137.204.1.15 : 34567
Destination	10.0.0.5 : 80

Richiesta	
Source	137.204.1.15 : 34567
Destination	82.1.2.3 : 80

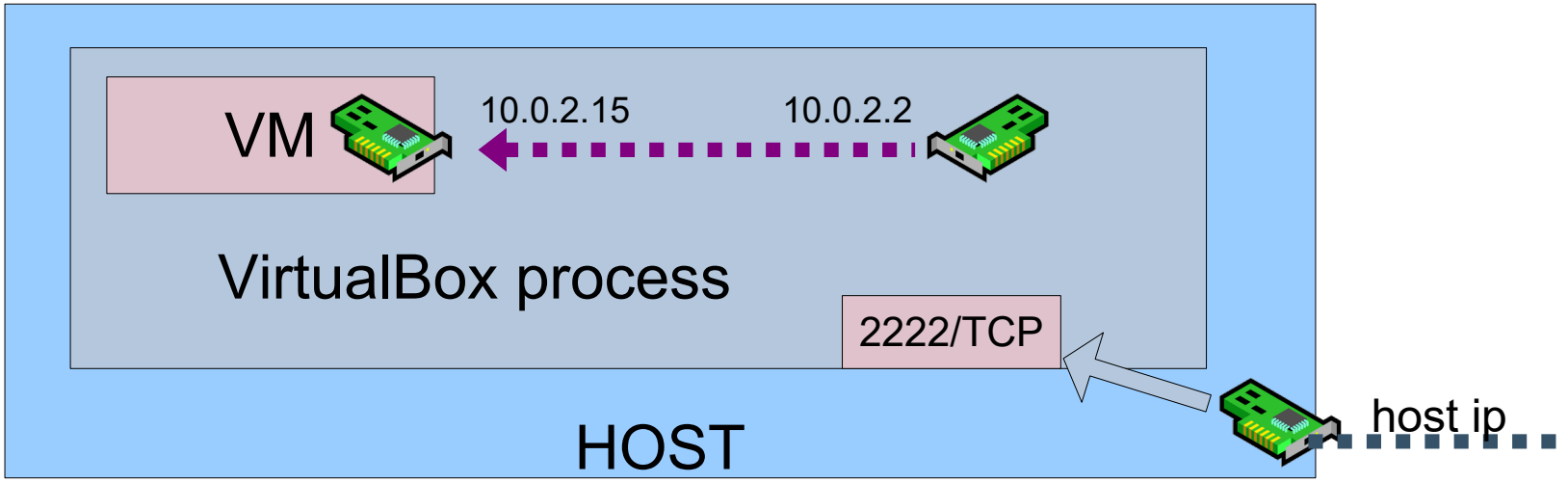
SNAT in VirtualBox

- Come anticipato, le VM VirtualBox nascono con un'interfaccia di tipo NAT
- VirtualBox stesso fa implicitamente da SNAT-Router
  - la VM viene configurata per usare come default gateway una interfaccia virtuale che consegna i pacchetti al processo VirtualBox
  - VirtualBox li propaga all'host OS come se li avesse generati lui stesso, e quindi vengono etichettati con l'IP sorgente dell'host



DNAT in VirtualBox

- È possibile accedere a una VM “NATtata” configurando VirtualBox perchè si comporti anche da DNAT-Router
  - il processo VirtualBox si mette in ascolto su di una porta TCP o UDP dell’host
  - il traffico entrante viene modificato assegnando come destinazione l’IP della scheda virtuale NAT del guest



Informazioni di sistema

- La configurazione di un'interfaccia di rete richiede come minimo
  - indirizzo IP
  - netmask
- più, se la rete locale è interconnessa ad altre
  - gateway specifici
  - default gateway
- e, se è disponibile un sistema di risoluzione di nomi
  - indirizzi dei server DNS
  - domini di default per costruire i FQDN

Metodi di configurazione

- Le informazioni possono essere assegnate
  - manualmente
  - da un server DHCP
  - localmente in modo automatico
- Attenzione alla dicotomia tipica runtime/persistenza
  - comandi per cambiare istantaneamente la configurazione
  - configurazione da applicare all'avvio
- Attenzione alle interfacce utente di configurazione
  - metodo classico: editing file di testo
    - segue l'approccio standard di qualsiasi servizio
    - modifiche non applicate fino a `systemctl restart networking` (o equivalenti)
  - metodo di default in molte distribuzioni di Linux: **NetworkManager**
    - se presente, può essere pilotato da GUI o `nmcli`
    - può sovrascrivere le modifiche runtime in qualsiasi momento

Configurazione runtime

- suite iproute2
  - comando ip + sottocomandi, sostituisce ifconfig e route
  - controllo completo di tutti gli aspetti più avanzati (link layer, interfacce virtuali, tunnel, VXLAN, policy-based routing, ...)
- sottocomando address (a)
  - visualizzazione `ip a`
  - assegnazione `ip a add <address>/<mask> dev <interface>`
  - rimozione `ip a del <address>/<mask> dev <interface>`
- sottocomando route (r)
  - visualizzazione `ip r`
  - routing via gw `ip r add <dst_net>/<mask> via <gw_addr>`
  - routing via dev `ip r add <dst_net>/<mask> dev <interface>`
  - rimozione `ip a del <address>/<mask>`

Configurazione "classica" Debian

- file `/etc/network/interfaces`
- *snippet* nella cartella `/etc/network/interfaces.d/`
- esempio tipico

```
auto eth0                          # attiva con ifup -a
iface eth0 inet static             # con dhcp al posto di static,
                                   # non serve altro
address 192.168.56.203
netmask 255.255.255.0                 # se omesso, class-based
```

  - opzionalmente

```
gateway 192.168.56.1               # uno solo, non per interfaccia
up /path/to/command arguments      # eseguito dopo configurazione
```

- `man 5 interfaces`

Configurazione "classica" RedHat

- directory `/etc/sysconfig/network-scripts`
  - contiene un file `ifcfg-`*`nome`* per ogni interfaccia

- Esempio
  - file `ifcfg-eth0`

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
PREFIX=24
IPADDR=10.0.1.27
```

https://access.redhat.com/documentation/it-it/red_hat_enterprise_linux/7/html/networking_guide/sec-configuring_ip_networking_with_ifcg_files

NetworkManager

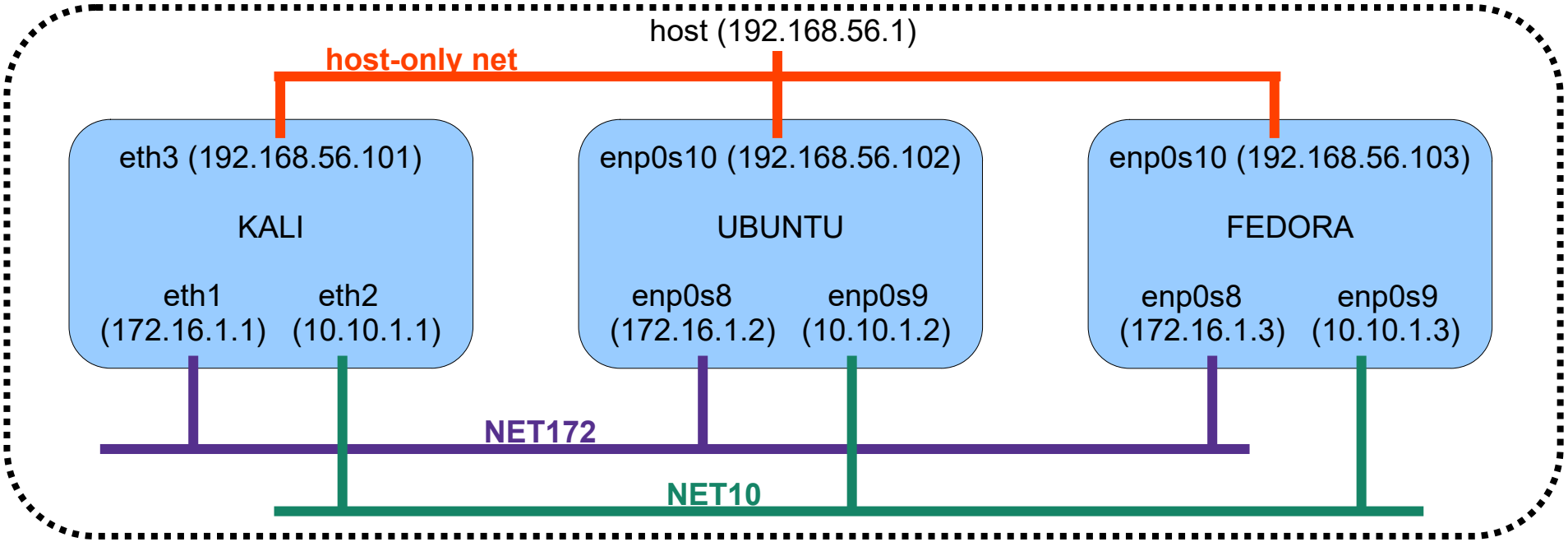
- Nelle distribuzioni che utilizzano systemd, la configurazione è tipicamente gestita da NetworkManager
  - (per ora) se trova configurazioni legacy come descritte nelle slide precedenti, le utilizza
  - volendo ripristinare il sistema più datato e semplice su Ubuntu:

```
systemctl disable --now NetworkManager
systemctl enable --now networking
```

- È un *demone* che conserva le configurazioni in file di testo in file nella directory `/etc/NetworkManager/system-connections`
- Può essere controllato dinamicamente col comando `nmcli`

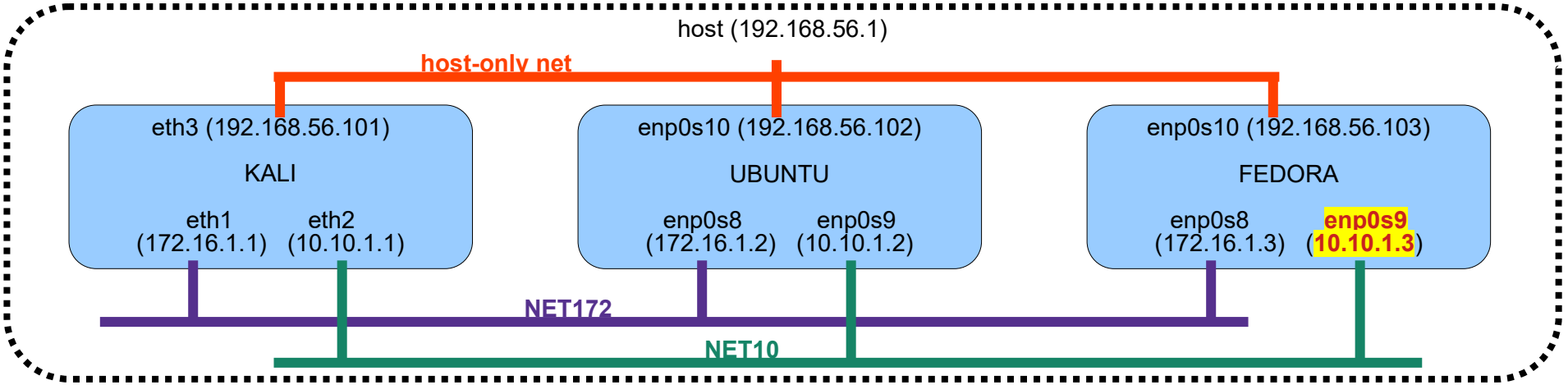
Laboratorio

- Obiettivo: far comunicare le nostre VM tra loro in questo modo
  - i nomi delle interfacce possono variare! adeguate a quel che appare sui vostri sistemi
  - l'interfaccia eth0 o enp0s3 è la scheda 1 predefinita di V.Box, connessa a internet



1) Impostiamo nuove schede di rete su VirtualBox
2) Configuriamo i sistemi (prossima slide)

Laboratorio



- **Esempio (comando tutto su una riga):**

etichetta della configurazione, può essere una stringa qualsiasi (univoca) che individua un modo di configurare **questa interfaccia**

```
sudo nmcli con add con-name net10 type ethernet ifname enp0s9 ipv4.method manual ipv4.address 10.10.1.3/24
```

in alternativa può essere *auto*, se sulla rete c'è un server DHCP che assegna automaticamente gli indirizzi (è quel che succede sull'interfaccia predefinita di VirtualBox.

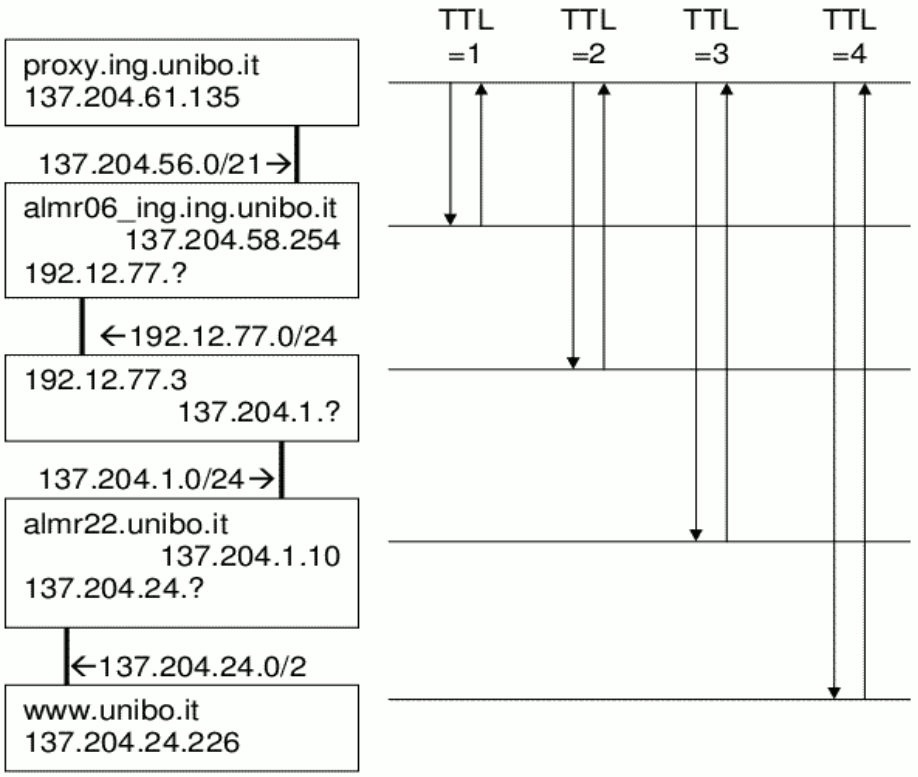
da specificare solo se il metodo è manual – è l'indirizzo che si vuole assegnare all'interfaccia, con l'indicazione breve della netmask (ampiezza della rete di cui fa parte)

Guide rapide a NetworkManager:

- **https://www.golinuxcloud.com/nmcli-command-examples-cheatsheet-centos-rhel/**
- **https://www.thegeekdiary.com/nmcli-command-examples-in-linux/**
- **https://opensource.com/article/22/4/networkmanager-linux**

Tool di monitoraggio

- Verifica di base della connettività
  - `ping <IP>`
- Verifica del percorso dei pacchetti
  - `traceroute <IP>`




```
[proxy@proxy proxy]$
     /usr/sbin/traceroute www.unibo.it

traceroute      to      cesias.alma.unibo.it
(137.204.24.226),  30  hops  max,  40  byte
packets

 1  almr06_ing (137.204.58.254)  1.578 ms
 2  192.12.77.3 (192.12.77.3)  4.246 ms
 3  almr22.unibo.it (137.204.1.10)  3.504 ms
 4  cesias.alma.unibo.it (137.204.24.226)
                                  3.108 ms
[proxy@proxy proxy]$
```

Modelli di interazione

Il modello client-server

- Con il termine *server* indichiamo un'applicazione che rende disponibile, mediante un'interfaccia standard, un servizio.
- Con il termine *client* indichiamo un'applicazione in grado di utilizzare i servizi messi a disposizione da un server.

Attenzione: a volte gli stessi termini vengono usati impropriamente per indicare l'host che ospita l'applicazione

- Il modello CS centralizza dati e metodi di elaborazione per un particolare servizio in un sito
  - evita duplicazioni, incoerenze di versione, proliferazione di interfacce
  - scarica sul client il compito di reperire i diversi servizi necessari a realizzare un'operazione complessa e distribuita

Il modello client-server

- Modello molti (client) a uno (server)
- Tipicamente sincrono e bloccante
  - Se il server non risponde il client non può avanzare
  - Si implementa nel client come reagire a fronte di un timeout (inferire un guasto?)
- Tipicamente con binding dinamico
  - Ad ogni invocazione il client può scegliere il server
  - Se il server non è disponibile dove atteso, la rete restituisce un errore
- Il server può essere progettato per offrire diversi tipi di prestazione
  - Servizio sequenziale o parallelo
  - Connessione persistente con notifiche push
  - Autenticazione e autorizzazione
  - ...

Modelli ad accoppiamento debole

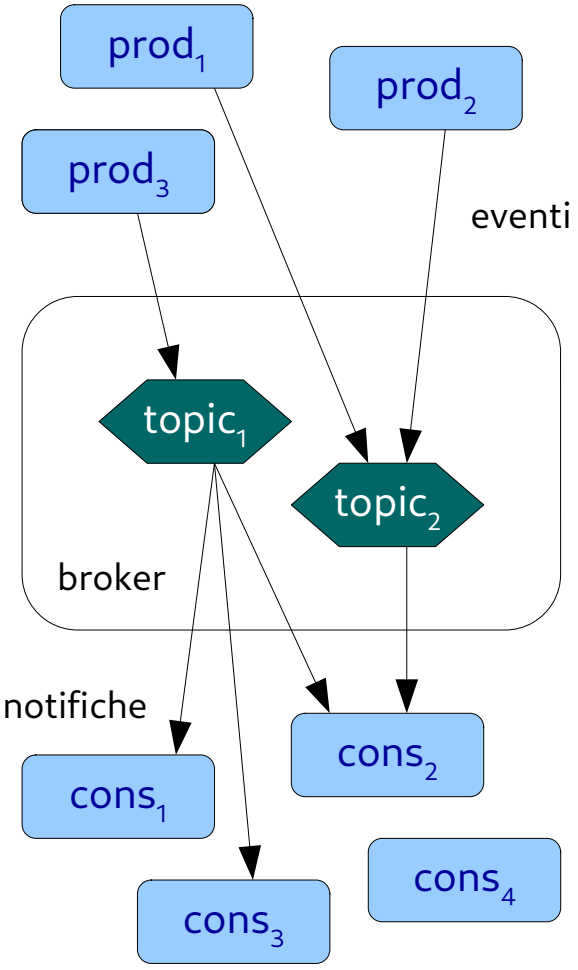
- Vincoli limitanti del modello client/server (ad accoppiamento forte):
  - contemporanea presenza degli attori di una comunicazione
  - impossibilità di comunicare molti a molti (broadcast, multicast)
- Modelli alternativi basati su **scambio di messaggi asincroni**
  - Sender / Receiver
  - Publish / Subscribe
- Questi modelli
  - richiedono e prevedono l'esistenza di un intermediario
  - sono più flessibili ma meno facili da usare (es.: conferma di ricezione?)
  - permettono ogni combinazione molti a molti

Sender / Receiver

- L'infrastruttura prevede almeno un intermediario a cui il sender affida il messaggio e da cui il receiver può leggerlo
- Possono essere presenti più intermediari
  - Per trasportare i messaggi da "vicino al sender" a "vicino al receiver"
  - Per implementare politiche di alta disponibilità
- L'enfasi non è sul messaggio ma sul receiver
  - Il sender e l'infrastruttura devono conoscerlo
  - Il receiver deve attivarsi per controllare la presenza di nuovi messaggi
  - Il multicast è possibile ma "mimato" con una replica del messaggio per ogni receiver esplicitamente indicato dal sender
- Es: posta elettronica

Publish / Subscribe (PubSub)

- Enfasi sui messaggi
  - *Pubblicati* da un *produttore*
  - *Notificati* a un *consumatore*
- L'infrastruttura (detta *message broker* o *event bus)* disaccoppia i due tipi di attori e non ha bisogno di conoscerli a priori, poichè
  - I produttori si manifestano inviando messaggi (eventi) identificati da un argomento (topic)
  - I consumatori si manifestano *iscrivendosi* a un argomento e ricevono le notifiche di nuovi messaggi
- Svantaggi: richiede una definizione precisa dei tipi di messaggi supportati
- Es: GUI, XMPP, MQTT, RSS, DDS



Servizi e protocolli

Domain Name Service (DNS)

- Gli host sono individuati da indirizzi IP, con cui gli apparati lavorano bene perché sono numeri binari.
- Gli esseri umani faticano a ricordare dei numeri, preferiscono usare un nome significativo della funzione dell'host.
- É utile disporre di un servizio che associ un nome all'indirizzo
- Ogni host viene identificato per mezzo del proprio nome (*hostname*) e del nome del proprio dominio (*domain name*).
- I dominî sono definiti in maniera gerarchica.
- I componenti della gerarchia sono separati con un punto
  es.: www.deis.unibo.it

Top Level Domain (TLD)

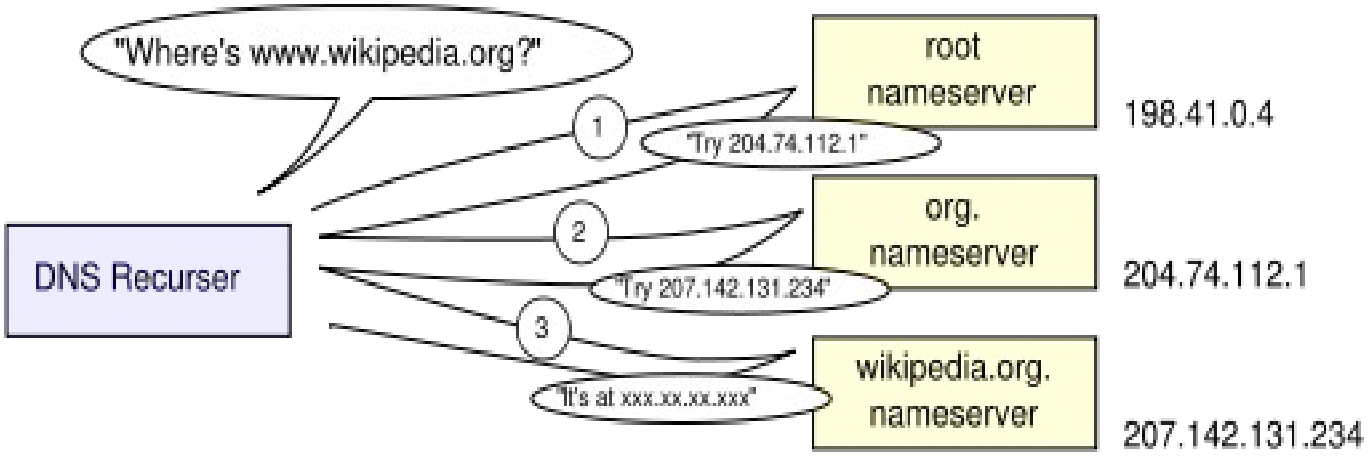
- Il primo componente a destra (la parte più elevata della gerarchia) è detto TLD (Top Level Domain) e deve essere scelto essenzialmente:
  - fra i TLD nazionali:
    - sigle di due lettere concesse solo alle nazioni come definite e riconosciute dall'ONU
    - es.: .it, .uk, .fr, .de, .tv, …
  - fra le sigle identificative di determinate categorie (gTLD):
    - dal 1985 esistono: .com, .org, .net, .edu, .gov, .mil, .int
    - dal 2001 anche: .aero, .biz, .coop, .info, .name, .museum, .pro,
    - dal 2005 anche: .jobs, .travel, .cat
    - dal 2006 anche: .mobi
    - Poi .asia, .tel, .xxx, ma altri lasciati in attesa di decisione per anni
  - La critica dell'arbitrarietà delle procedure per ottenere l'attivazione dii un gTLD ha infine portato alla possibilità di richiedere gTLD qualsiasi (decisione 2008, avvio 2012, attivazione 2013)
    - 185.000$ + 25.000$/anno

Aspetti gestionali

- La gestione dei TLD è in carico allo IANA.
  - http://www.iana.org/domain-names.htm
- All'atto pratico la gestione di un dominio viene esercitata per mezzo di deleghe ai gestori dei sottodomini:
  - sponsor o operatori dei TLD generici
    - .com --> Verisign, .aero --> SITA, ...
  - enti nazionali responsabili dei TLD geografici
    - .it --> IIT-CNR, ...
  - qualsiasi ente o persona responsabile dei domini al di sotto dei TLD (domini di 2° livello)

Risoluzione dei nomi in indirizzi

- Il DNS è un sistema **client-server**.
  - Il client prende il nome di *resolver*, e rivolge le proprie interrogazioni (*query*) tramite il protocollo UDP ad un server in ascolto sulla porta 53.
  - Il server può farsi carico del reperimento dell'indirizzo richiesto, o solo di fornire un puntatore da cui proseguire le ricerche



Risoluzione inversa

- Per la mappatura inversa (da IP a nome) si usa lo stesso meccanismo con un “trucco”: si sfrutta la gerarchia degli IP e si definisce un spazio dei nomi con origine in-addr.arpa. Si noti che
  - gli IP sono scritti dal byte che rappresenta la rete più ampia a quello che rappresenta la parte più specifica dell'indirizzo
    - Es: 137.204.57.1 == RIPE --> UNIBO --> DEIS --> PROMET1
  - i nomi sono scritti a partire dal nome specifico dell'host verso il TLD che rappresenta il dominio più ampio nella gerarchia
    - www.deis.unibo.it
- quindi per usare la stessa infrastruttura la ricerca di 137.204.57.1 viene effettuata con una query del nome 1.57.204.137.in-addr.arpa
- NOTA: le deleghe ai diversi livelli del “nome” non sono più arbitrarie, ma devono seguire l’assegnazione della sottorete IP corrispondente (cioè ad esempio il RIR che gestisce la rete 137.0.0.0/8 è l’unico server autorevole per il “dominio” 137.in-addr.arpa)

Identificazione dei domini e degli IP

- I registri degli indirizzi IP e dei nomi di dominio sono pubblicamente consultabili per poter risalire ai titolari
  - utilizzo delle query DNS per associare nome <--> indirizzo
  - utilizzo dei servizi *whois* per identificare i responsabili tecnici e legali del domino o del blocco di indirizzi
    - i registri non sono integrati a livello globale
    - utilizzo dei servizi whois del RIR appropriato per l'indirizzo
    - utilizzo dei servizi whois del registrar appropriato per il TLD

Posta elettronica

La posta elettronica (e-mail) è stato uno dei primi servizi introdotti su ARPAnet, ed è tuttora uno dei più utilizzati su Internet.

I problemi affrontati per la ricezione dei messaggi sono:

- È richiesta l'identificazione univoca del destinatario
  - indirizzo costruito in modo gerarchico: utente@host.dominio
- È necessario poter depositare i messaggi fino alla effettiva lettura
  - casella postale, ovvero spazio su di un server di posta
- È utile un sistema di trasporto che tolleri la temporanea irraggiungibilità del server di destinazione
  - ridondanza, per ogni destinazione sono indicati punti di appoggio alternativi (inizialmente una vera "rete sulla rete")

Posta elettronica

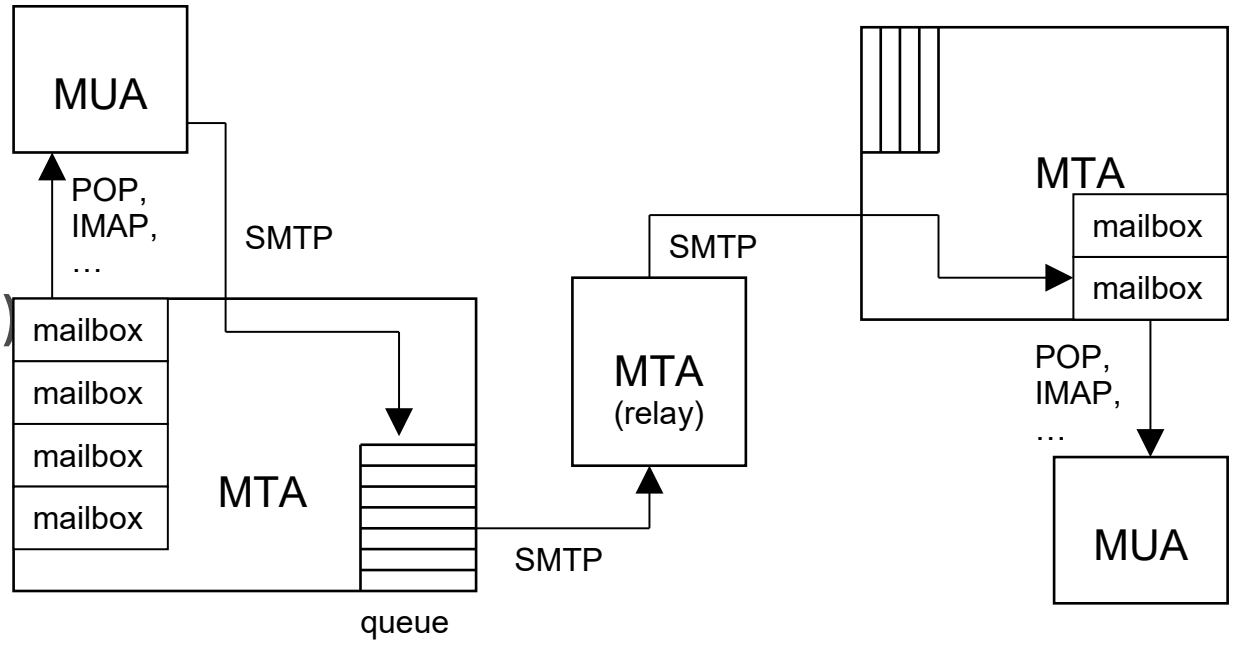
Il reperimento delle informazioni per individuare i server di posta di un dominio è affidato al DNS - stretta integrazione tra i sistemi

I problemi affrontati per la spedizione dei messaggi sono:

- È desiderabile che l'utente veda un'interazione veloce
  - queuing, il server accetta i messaggi e li accoda per spedirli
- L'utente non deve essere obbligato a mantenere la connessione al server durante la lettura e la composizione dei messaggi
  - client di posta elettronica e protocolli di colloquio col server
  - complessivamente implementano un'architettura sender/receiver

Posta elettronica

- **Applicazioni:**
  - **MUA (Mail User Agent): Client (es. Eudora)**
  - **MTA (Mail Transport Agent): Server di transito e smistamento**



- **Protocolli:**
  - **SMTP (Simple Mail Transfer Protocol), via TCP alla porta 25**
  - **POP (Post Office Protocol),), via TCP alla porta 110**
  - **IMAP (Internet Mail Access Protocol), via TCP alla porta 143**

Posta elettronica

- Originariamente pensata per trasferire solo testi
  - *header*, contenente le informazioni per la consegna del messaggio (destinatario, mittente, soggetto, data, …)
  - *body*, contenente il testo del messaggio.
- Presto riconosciuta come utile per trasferire dati di ogni genere
  - estendere le possibilità di codifica del body con lo standard MIME (Multipurpose Internet Mail Extension)
  - Il corpo della mail rimane in formato ascii, ma in esso sono codificate le informazioni binarie originali (attach).
    - Possibilità di spedire più dati distinti in un solo messaggio
    - Possibilità di spedire una indicazione del tipo di dato platform-independent
    - Consente di definire nuovi tipi di dato
  - del tutto trasparente per i server

Formato di un messaggio

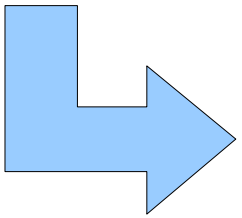
```
From luca.ghedini@mail.ing.unibo.it Wed Nov 12 23:06:49 1997
Received: (from ghedo@localhost)
     by mail.ing.unibo.it (8.8.8/8.8.5) id XAA04169
     for luca.ghedini; Wed, 12 Nov 1997 23:06:49 +0100 (MET)
Date: Wed, 12 Nov 1997 23:06:49 +0100 (MET)
From: Luca Ghedini <luca.ghedini@mail.ing.unibo.it>
To: luca.ghedini
Subject: Saluti
Mime-Version: 1.0
Content-Type: multipart/mixed;
boundary=6e57_3bae-6e62_7e55-1cf_41
Content-Length: 590

--6e57_3bae-6e62_7e55-1cf_41
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Content-MD5: DjPyOvmsowwWAIx7HqbxRg==
X-Sun-Data-Type: text

Ciao, ti mando le cose che mi avevi chiesto
--6e57_3bae-6e62_7e55-1cf_41
Content-Type: image/gif; name=2torri.gif
Content-Transfer-Encoding: base64
Content-MD5: LcOMV2MEqHKz/Ufm58vvlQ==
Content-Description: 2torri.gif
Content-Disposition: attachment; filename=2torri.gif
X-Sun-Data-Type: gif-file

R0lGODlh2AA0AfcAACAgQCAgAg/yBAQCBAwCBA/yBgQCBggCBgwCBg/yCAQCCA
cmtzIEluYy4NDVVzZSBubyBob29rcwA7
--6e57_3bae-6e62_7e55-1cf_41--
```

World Wide Web

Il problema: recuperare informazioni in un contesto altamente eterogeneo.

Cosa esisteva prima del web ?

- programmi specializzati per accedere a informazioni codificate in differenti maniere.
- difficoltà di integrazione delle informazioni provenienti
  da fonti non omogenee
- interfacce di accesso non consistenti



- Costi elevati
- Difficoltà di apprendimento
- Rapida obsolescenza dei programmi
- Necessità di “aggiustamenti” manuali

Scopi del WWW (CERN 1989)

- differenti rappresentazioni delle informazioni (es. per le immagini : GIF, JPEG, BMP,...)
- differenti modalità di recupero delle informazioni (database query, ftp site, ...)
- differenti modalità di autenticazione (password, chiavi pubbliche, DES,...)
- trasparenza accesso e allocazione
- presentazione multimediale
- unicità di interfaccia utente

Ipertesti

L'elemento base del www è l'ipertesto.

- Un ipertesto e' un testo arricchito di immagini, suoni, e riferimenti ad altre informazioni (link)
- Ciascun link punta ad un'altra informazione che può essere ovunque (World Wide)
- L'insieme dei link forma una rete (Web) in cui si incrociano i percorsi logici secondo cui l'informazione può essere esplorata.
- Il risultato è un ipertesto distribuito su una rete di calcolatori
- Accesso secondo il modello client/server

Componenti del WWW

- Browser
  - accetta le richieste dell'utente e gli presenta i risultati della esplorazione
- Server:
  - Immagazzina gli ipertesti, elabora le richieste ricevute dai browser ed invia loro i risultati della elaborazione
- Helper
  - moduli per aggiungere ai browser particolari funzionalità
- Common Gateway Interface/Server Side Scripting
  - permettono agli utenti di eseguire programmi sui server (esecuzione remota).

Standard relativi al WWW

Occorrono degli standard:

- per identificare le risorse in rete
  - Uniform Resource Identifier/Locator (URI/URL)
- per far comunicare server e browser
  - HyperText Transfer Protocol (HTTP)
- per permettere al browser di presentare i documenti all'utente
  - HyperText Markup Language (HTML)
  - Cascading Style Sheets (CSS)
- per gestire l'esecuzione lato server
  - Common Gateway Interface (CGI)

Standard relativi al WWW

Helper app (Java, Flash, reader, ...)

browser

Server

Processo esterno (es. query DB)

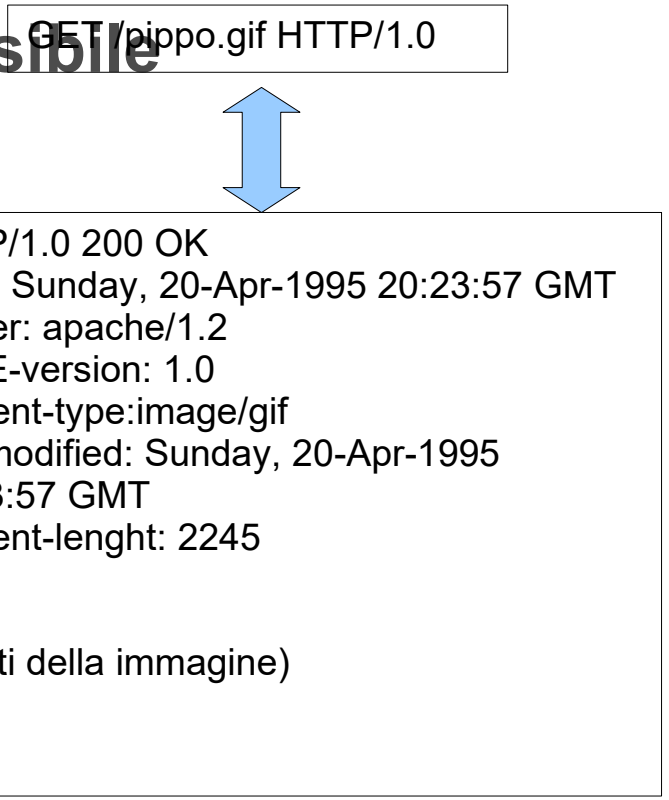
I documenti sono in HTML

Browser e Server si parlano in HTTP

Server e programmi da eseguire si parlano in CGI

Transazione HTTP

- Il protocollo è il più semplice possibile
  - V. 1.1 (RFC 2068)
  - REQUEST/RESPONSE
    - nessuna memoria del passato influenza le risposte successive
  - Capacità di negoziazione
    - il browser indica che tipi di dato, lingue, codifiche è in grado di gestire
    - è possibile elencare alternative in ordine di preferenza
  - Riutilizzo degli standard
    - messaggi codificati secondo MIME



HTTP Request

- 2 parti separate da una riga vuota:
  - header <metodo> <URL> <versione>
    <opzioni>
  - body (opzionale) dati relativi alla richiesta

- Metodo: Es. GET, HEAD, POST, PUT
- URL solo la parte relativa (tolti protocollo, host, porta)
- Versione HTTP/1.0 o HTTP/1.1
- Opzioni Es. Accept:…, If-modified-since:…, Referer:…,
  User-Agent:…,

HTTP Response

- 2 parti separate da una riga vuota:
  - header <status code> <status description>
    <opzioni>
  - body <risposta>
- Status: 1xx Information
  2xx Success
  3xx Redirection
  4xx Client Error
  5xx Server Error
- Opzioni Es. Content-type:…, Expires:…,
  Content-encoding:…,

Web Server

- Il web server di base è un *mappatore di URL su risorse locali del server*
  - Risorse locali *statiche* (HTML, CSS, immagini ed altri media, programmi da eseguire lato client, …)
    - Può effettuare operazioni sofisticate di negoziazione della versione da erogare, compressione, definizione degli header, ecc..., ma di base deve semplicemente trasformare il path della URL in un path del filesystem
    - Es: http://www.example.com/docs/faq.html → /var/www/sites/example.com/pages/docs/faq.html
  - Risorse locali *dinamiche*
    - vengono generate su richiesta
    - il risultato può essere un qualunque formato supportato dal browser, esattamente come per le risorse statiche
    - In questo caso la mappatura è tra URL e programma
      - Serve un metodo per codificare i parametri tra browser e server
      - Serve un metodo per passare i parametri tra server e programma

Lab connessioni web

- Installiamo e avviamo il webserver apache
- Verifica dello stato delle connessioni
  - `ss`
    - `-t / -u` TCP/UDP only
    - `-l / -a` stato LISTEN (il default è ESTABLISHED) / ALL
    - `-n` non risolvere gli indirizzi/porte in nomi simbolici
    - `-p` mostra processi che usano la socket
- Intercettazione del contenuto dei pacchetti
  - `tcpdump`
  - `wireshark`

  → man pages e documentazione

REST

- REpresentational State Transfer
- Uno dei metodi più usati per il passaggio di parametri client-server
- REST NON è
  - Un framework
  - Una tecnologia
  - Uno standard
- bensì uno *stile architetturale*

  *"An architecture style is a coordinated set of architectural **constraints** that restricts the roles and features of architectural elements, and the allowed relationships between those elements, within any architecture that conforms to that style" – Roy Fielding*

REST – vincoli architetturali principali

1) REST usa il modello client/server
   - Separa presentazione da memorizzazione ed elaborazione dei dati
   - Migliora la portabilità delle interfacce tra piattaforme
   - Migliora la scalabilità semplificando i server
   - Permette ai vari componenti di evolvere indipendentemente
2) REST è senza stato
   - Ogni richiesta dal client deve essere comprensibile senza trarre vantaggio dalla conoscenza che altre operazioni sono state svolte in precedenza. Dove serve, lo stato deve essere conservato dal client.
   - Più robusto dopo interruzioni, più scalabile per il server,
   - Meno performante sulla rete, meno verificabile lato server.

REST – vincoli architetturali principali

3) Le risposte REST specificano le politiche di caching
   - Se una risposta è “cacheable” il client ha diritto di usarla per successive richieste equivalenti
   - Migliora l’efficienza
   - Riduce l’affidabilità
4) REST ha un’interfaccia uniforme
   - l’implementazione dei servizi è disaccoppiata dal modo in cui sono esposti, che deve rispettare quattro vincoli
     1) Identificazione delle risorse
     2) Manipolazione delle risorse attraverso rappresentazioni
     3) Messaggi auto-descrittivi
     4) Hypermedia come motore dello stato dell’applicazione
   - Permette lo sviluppo indipendente di diverse implementazioni
   - Peggiora l’efficienza perché non permette ottimizzazioni

REST - elementi

- Trasporto: HTTP
- Identificazione delle risorse: URL
- Esempi
  - Mappatura della tabella *utenti* del database *servizio* collocato sul server *rdb01*
    `http://rdb01/servizio/utenti`
  - Mappatura di query sul database
    `http://rdb01/servizio/utenti[nome="Gian%"]`
- Operazioni REST = richieste HTTP; corrispondenza con i metodi CRUD:
  - POST → Create
  - GET → Read
  - PUT → Update / Create
  - DELETE → Delete

REST – esempio di uso dei metodi

- Richiesta con tutti i dati codificati nell'URL `http://rdb01/servizio/utenti[nome="Gian%"]`
  - POST → errore, deve avere dati nel body
  - GET → restituisce tutti gli utenti con nome che inizia per Gian
  - PUT → errore, deve avere dati nel body
  - DELETE → cancella tutti gli utenti con nome che inizia per Gian
- Richiesta con dati inseriti nel body HTTP `http://rdb01/servizio/utenti`
  - POST → crea un nuovo utente
  - GET → restituisce tutti gli utenti che rispettano i criteri inseriti nel body
  - PUT → crea o aggiorna un utente a seconda dei dati del body
  - DELETE → cancella tutti gli utenti che rispettano i criteri inseriti nel body
- In caso di errori, vengono usate le convenzioni di HTTP (4xx/5xx)

REST – dati

- Il contenuto delle risposte può essere in qualsiasi formato
- Tipicamente:
  - JSON
    - oggetti JavaScript
    - Se il client usa JavaScript per esporre un'interfaccia dinamica, il parsing è molto facilitato (ma i dati diventano parte del programma, con potenziali problemi di sicurezza)
    - Abbastanza compatto
  - XML
    - Dati contrassegnati da tag
    - Semantica dei tag definita da uno schema separato
    - Più pesante ma la validazione è più robusta e non c'è commistione di dati e codice

Protocolli per IoT

- HTTP non è adatto per l'internet delle cose
  - È pensato per documenti verbosi e corredati di molti metadati
  - Il peso di client e server non è adeguato per microcontrollori
  - Non prevede garanzie di successo della comunicazione
  - Il modello client/server richiede un'interazione a polling
    - Migliaia di dispositivi che fanno poco = migliaia di richieste inutili
    - Cicli di polling necessariamente rarefatti = scarsa reattività agli eventi
- Soluzione: protocolli basati sul modello PubSub
  - Modello a eventi asincroni
  - Ricchezza semantica sacrificata in nome della leggerezza
  - Contrasto alla perdita di pacchetti dovuta a canali inaffidabili
  - Numerose soluzioni (MQTT, AMQP, XMPP, CoAP, DDS, STOMP, ...)