PARTE E

Network Address Translation

Modulo 1: Motivazioni NAT

Indirizzi IP: sufficienti?

- **Attualmente ciascun indirizzo IP (IPv4) consiste di 4 byte (32 bit)**
- **In IPv4 vi sono 3.758.096.384 indirizzi IP utilizzabili per l'indirizzamento degli host**
- **Ci sono un miliardo di host connessi**

Eppure gli indirizzi stanno (quasi!) esaurendo…
Perché?

Perché gli indirizzi IP esauriscono

Evoluzione della tecnologia e dei dispositivi che consentono connessione a Internet:

- Siamo partiti con un indirizzo IP (computer) per molte persone
- Stiamo arrivando a un indirizzo IP per persona
- Arriveremo ad aver bisogno di più indirizzi IP per persona

Gli indirizzi IP non sono e non possono essere distribuiti uniformemente

- Classe A: 128 reti, 16M host
- Classe B: 16K reti, 64K host
- Classe C: 2M reti, 256 host

Possibili soluzioni

- ***Presente* (la soluzione "a breve termine")**
  - DHCP → indirizzi occupati solo per gli host connessi
  - Tecniche di **Network Address Translation** ("*Natting*") → *creazione di spazi di indirizzamento privati*

- ***Futuro* (la soluzione "definitiva"):**
  - Passaggio al nuovo standard **IPv6** → aumentare il numero di indirizzi disponibili (2^{128} >>> 2^{32})

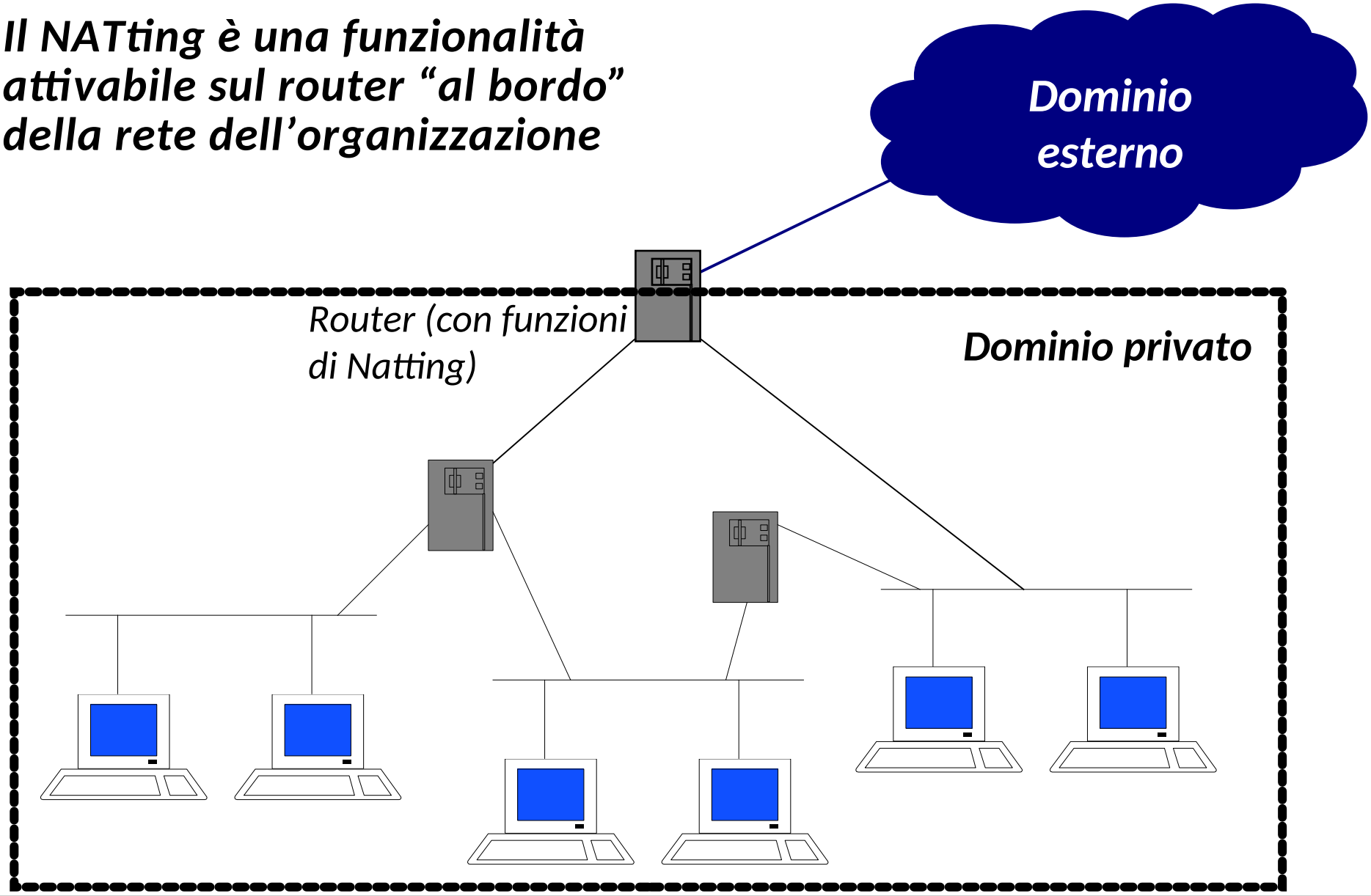
Reti private e semi-private

- **Per alcune (poche) organizzazioni è importante avere reti private in senso stretto:**
  - nessun nessun pacchetto esce da una rete *privata* e nessun pacchetto entra in una rete *privata*
  - indirizzi univoci solo all'interno della rete *privata*

- **Per molte altre organizzazioni è importante avere reti semi-private con tre categorie di host:**
  - nessun accesso da/a host fuori "dall'organizzazione" (molti host)
  - accesso parziale (host che possono raggiungere l'esterno ma non sono raggiungibili dall'esterno)
  - accesso completo (pochi host, es. server Web)

NATting per reti semi-private

Il NATting è una funzionalità attivabile sul router "al bordo" della rete dell'organizzazione



Indirizzi non routable

Poiché per molte organizzazioni non è necessario che tutti i loro indirizzi siano visibili globalmente, per evitare di sprecare indirizzi, la IANA ha definito delle *reti private*, ossia:

- **non uniche a livello mondiale (RFC 1918)**
- **con *indirizzi IANA privati* (Non-Internet Routable IP Addresses)**
- **gli indirizzi "non routable" si possono utilizzare senza richiedere autorizzazione, purché si garantisca che il traffico e gli indirizzi siano limitati alla rete interna**
  - 192.168.0.0/16 (256 reti classe C)
  - 172.16.0.0/12 (16 reti classe B)
  - 10.0.0.0/8 (1 rete classe A)

Indirizzi IP privati per Intranet

In questo modo, un'organizzazione tipicamente ha la possibilità di progettare una rete che:

- **include host visibili da Internet (*host pubblici*)**
- **altri host che non sono visibili (*host privati*)**

Gli *host privati* possono scambiare pacchetti:

- **solo con altri host privati all'interno della stessa rete senza intermediari**
- **con host pubblici mediante:**
  - **application gateway (*proxy*) sugli host pubblici**
  - **Network Address Translation (NAT)**

NAT router

Il *NAT router* (un router con funzionalità di NATting) si interpone tra la rete locale di una organizzazione e Internet con i seguenti compiti:

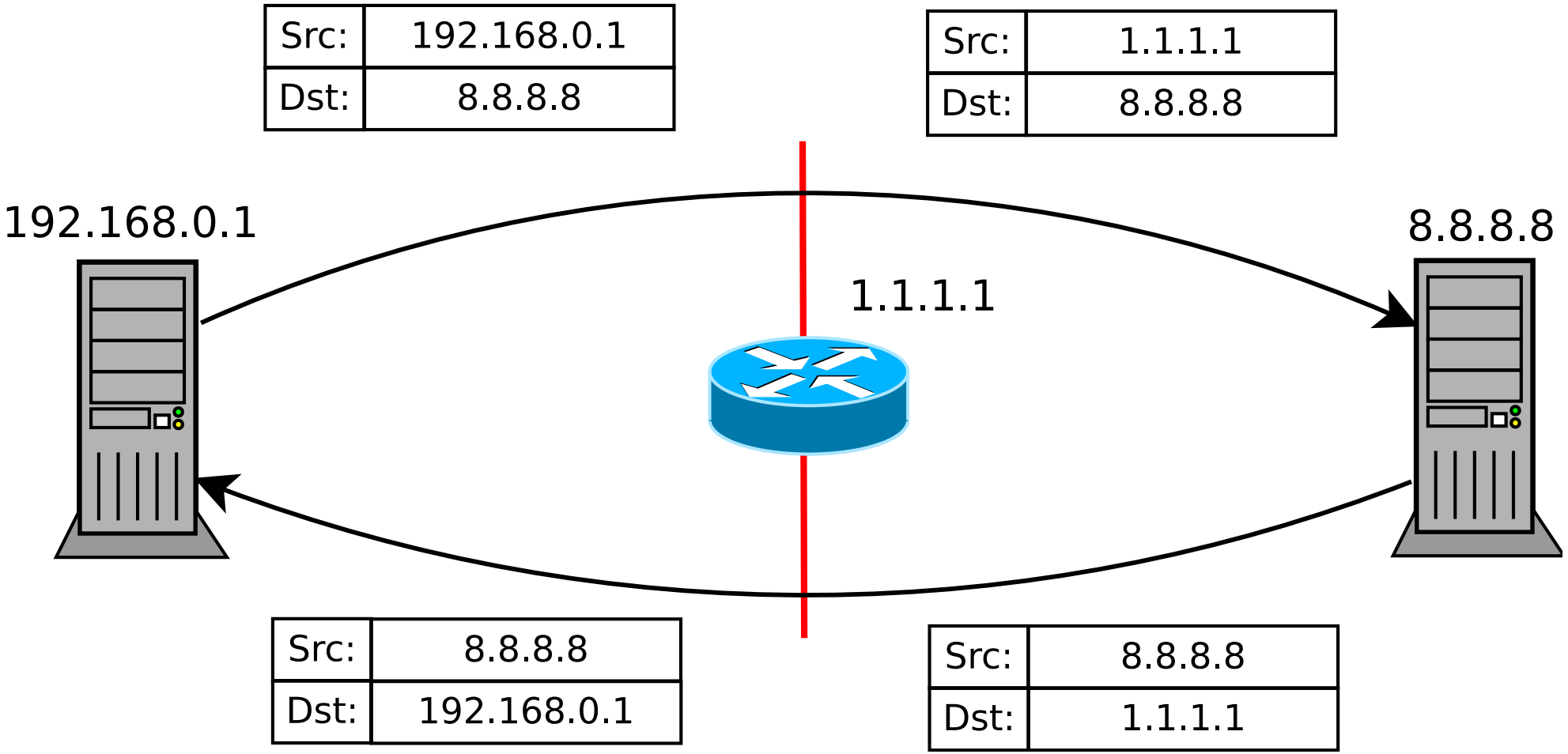
- **Mappa gli indirizzi IP tra due domini (interno-esterno)**

  indirizzi locali* ← : → *indirizzi IP globali

- **Garantisce la trasparenza del routing tra gli *end system***
- **“Moltiplica” le possibilità di interconnessioni di host di una organizzazione (nel caso in cui l’organizzazione abbia a disposizione un numero di indirizzi IP inferiore al numero di host)**
- **Aumenta la sicurezza evitando di rendere visibili all’esterno alcuni computer di una organizzazione**

Modulo 2: Funzionamento

NAT: esempio


Src: 192.168.0.1
Dst: 8.8.8.8
Src: 1.1.1.1
Dst: 8.8.8.8
192.168.0.1
8.8.8.8
1.1.1.1
Src: 8.8.8.8
Dst: 192.168.0.1
Src: 8.8.8.8
Dst: 1.1.1.1

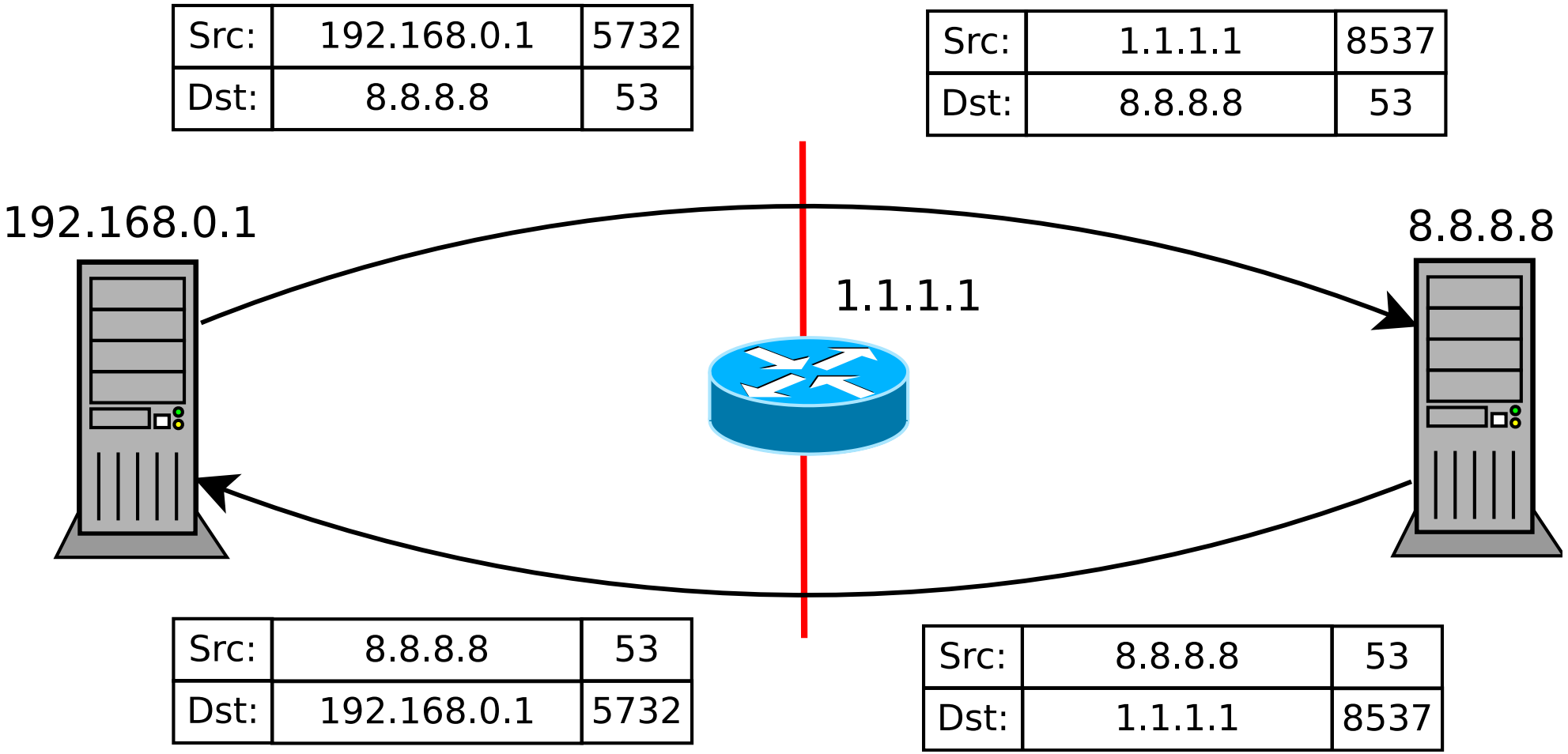
PAT: Port address translation

- **Necessario per condividere pochi indirizzi IP pubblici (spesso uno solo) fra tanti host dotati di indirizzi IP privati;**

- **Il protocollo agisce sugli header del livello 4 (trasporto) per estende il mapping *(binding)* del NAT *da coppie di indirizzi IP a coppie IP:porta*;**

- C'è confusione nei termini: alcuni usano termini alternativi, e *comunemente si usa il termine NAT per riferirsi sia a PAT che a NAT (anche durante questo corso useremo il termine PAT solo quando vogliamo indicare specificamente).*

NAT + PAT: esempio



Binding degli indirizzi

Il router gestisce una corrispondenza (*binding*) tra gli indirizzi dei due domini tramite una TABELLA che mantiene una riga per ciascuna "connessione aperta":

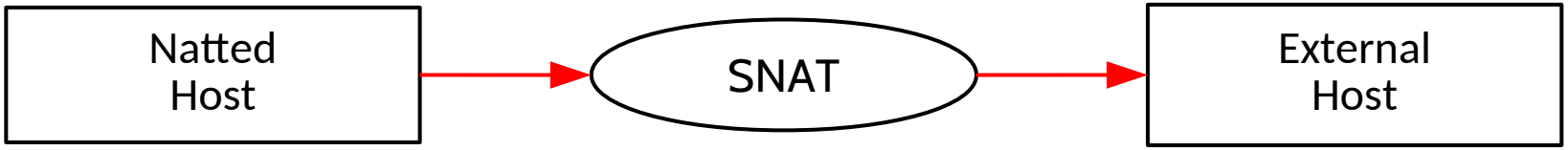
- **binding statico**
  - la corrispondenza viene configurata manualmente con l'indirizzo IP del router o con uno degli indirizzi pubblici di un pool di indirizzi
  - Il pool potrebbe essere piccolo rispetto alla rete locale: questo potrebbe determinare il numero massimo di connessioni contemporanee che l'organizzazione accetta verso Internet
- **binding dinamico**
  - la corrispondenza indirizzo privato-pubblico viene calcolata dinamicamente a seconda del traffico e dell'host che fa richiesta

Nel caso di più connessioni che condividono lo stesso IP pubblico, la tabella deve conservare altre informazioni relative alla sessione

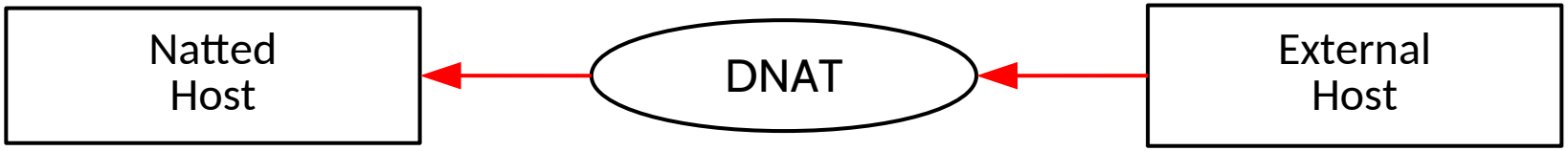
Source Natting e Destination Natting

Esistono due scenari da considerare per permettere la comunicazione fra reti divise da un router nat:

1) **un host *nattato* alla rete vuole iniziare a comunicare con un host esterno;**



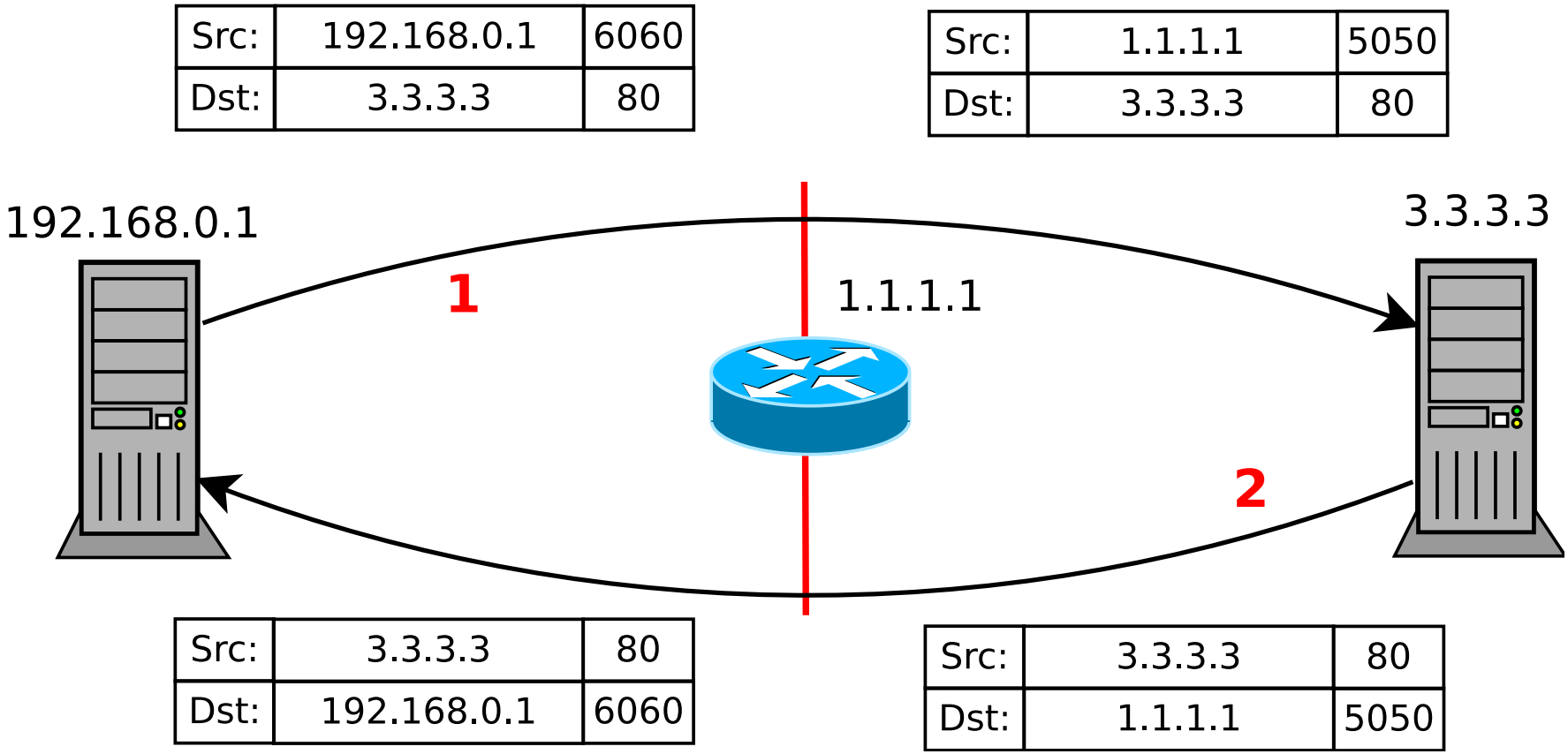
2) **un host esterno alla rete vuole iniziare a comunicare con un host *nattato*.**



Usiamo i termini Source Natting (SNAT) e Destination Natting (DNAT) per riferirci a protocolli che operano rispettivamente nel primo e nel secondo scenario.

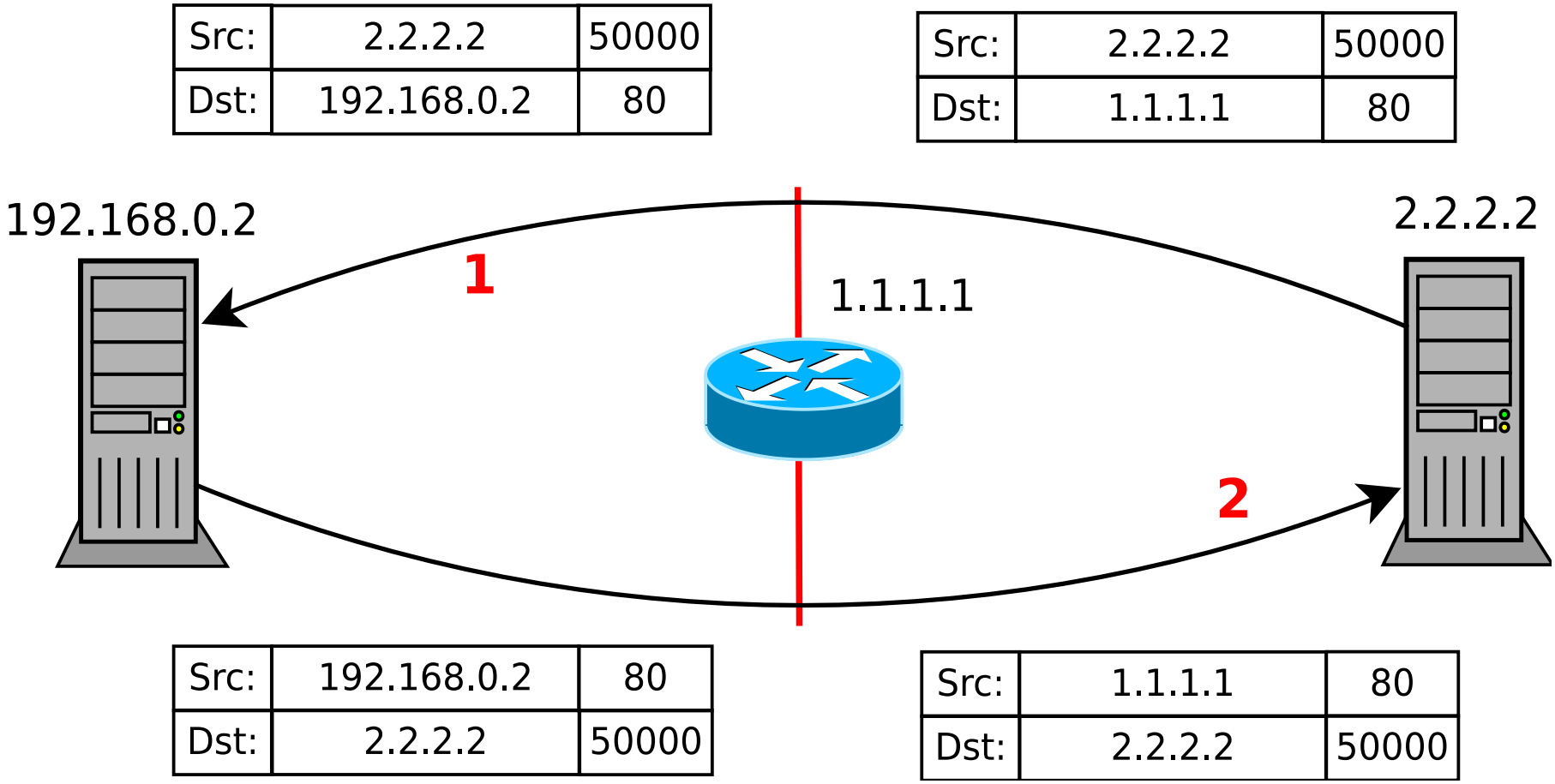
SNAT: esempio

Consideriamo lo scenario precedente, in cui un client interno alla rete private si connette a un server esterno alla rete.



DNAT: esempio

Consideriamo uno scenario in cui un client esterno si connette a un server interno *con ip privato*.

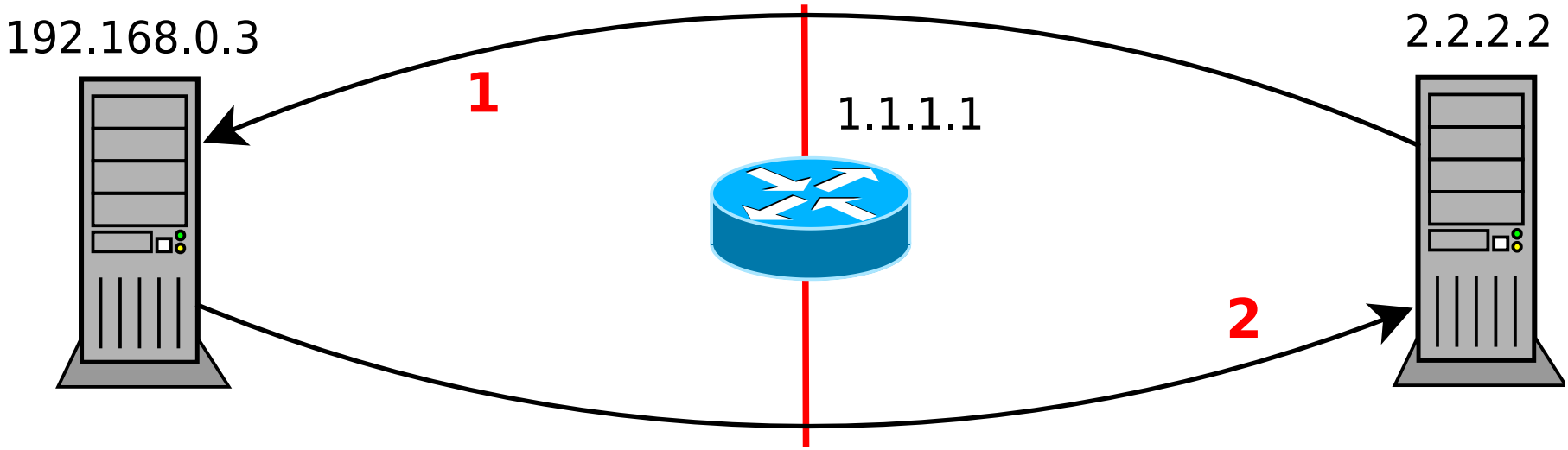


DNAT: esempio

Consideriamo un secondo scenario in cui un client esterno si connette a un server interno *con ip privato*, e la porta utilizzata dal server interno è diversa da quella resa disponibile dal gateway della rete.

Src:	2.2.2.2	50000
Dst:	192.168.0.3	8888

Src:	2.2.2.2	50000
Dst:	1.1.1.1	8080

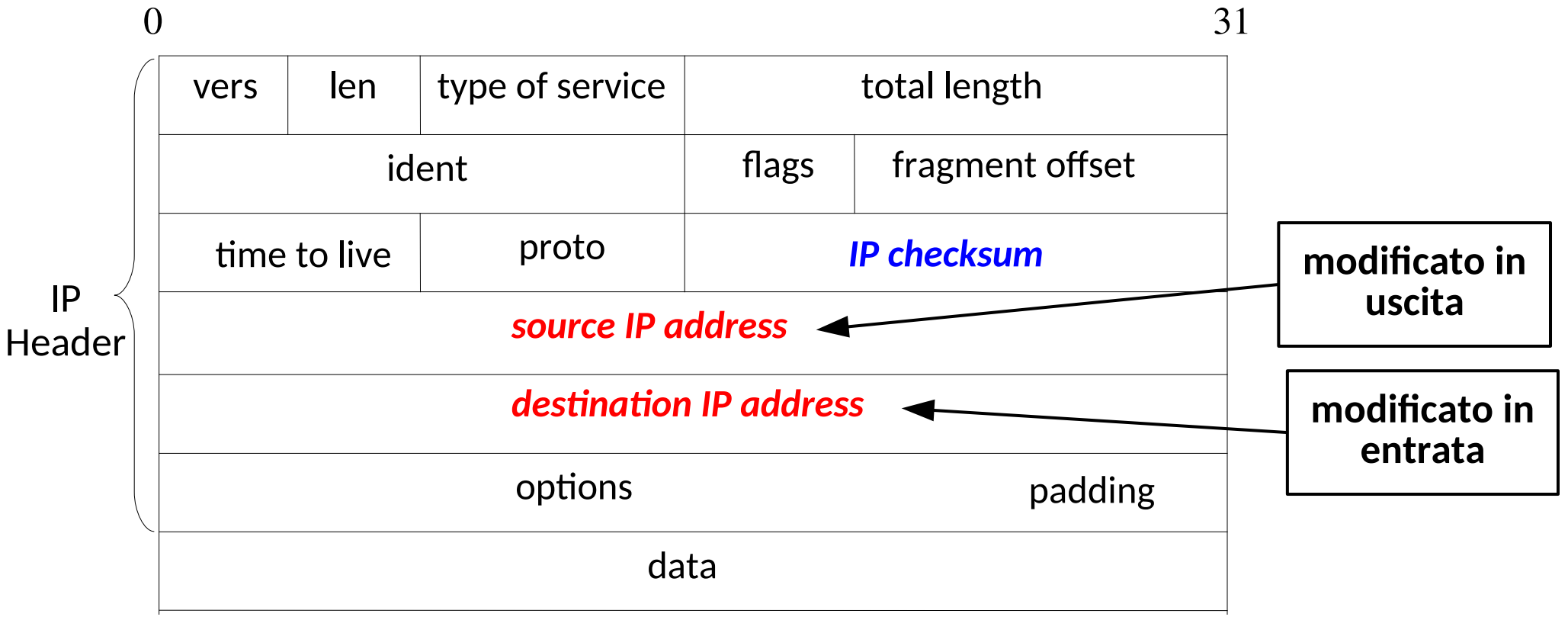


Src:	192.168.0.3	8888
Dst:	2.2.2.2	50000

Src:	1.1.1.1	8080
Dst:	2.2.2.2	50000

NAT: modifica del datagram IP

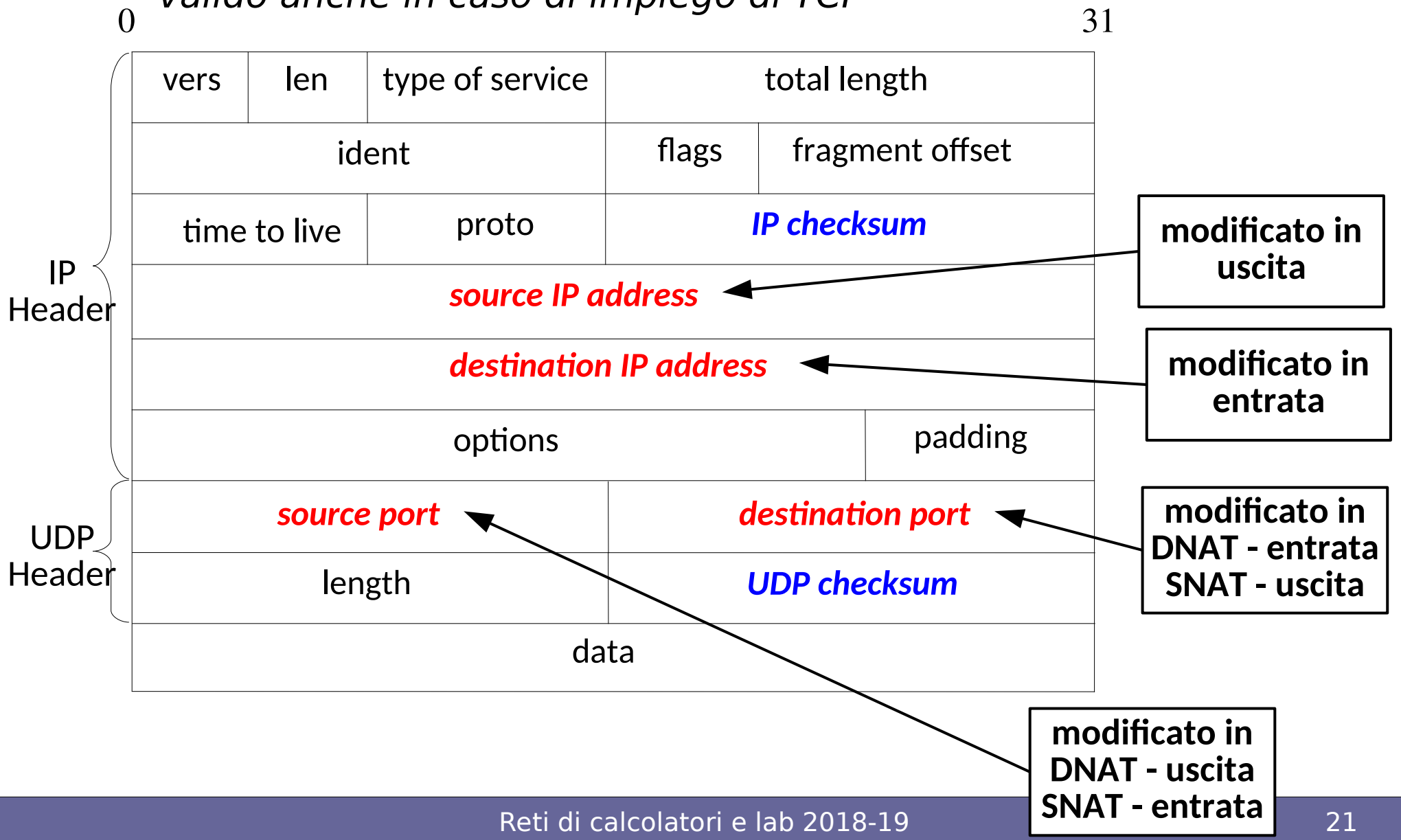
Nota: *Entrata/Uscita sono riferite al flusso di inizio della comunicazione*



Modificando l'header del pacchetto IP, il checksum va aggiornato sia per i pacchetti in entrata sia per i pacchetti in uscita

NAT + PAT: modifica del datagramma

Questo esempio considera l'header UDP, ma è valido anche in caso di impiego di TCP



Natting: contro

- **Distrugge la semantica della comunicazione *end-to-end* in quanto gli IP impiegati dai partecipanti non vengono mantenuti immutati nella trasmissione dei pacchetti su internet:**
  - gli host interni non sono raggiungibili dall'esterno: sono necessarie configurazioni ad hoc per garantire la raggiungibilità dei server;
  - conflitti fra host che implementano servizi dello stesso tipo (e.g., conflitto fra porte su cui raggiungere i server)
  - ispezione e modifiche sui pacchetti a livello *applicativo* per permettere il funzionamento di alcune applicazioni (e.g., ftp) e protocolli (e.g., p2p) che divergono dal paradigma client-server;

Natting: pro

- **Distrugge la semantica della comunicazione *end-to-end* in quanto gli IP impiegati dai partecipanti non vengono mantenuti immutati nella trasmissione dei pacchetti su internet:**
  - gli host interni non sono raggiungibili dall'esterno:
    ottimo dal punto di vista della sicurezza della rete
- **Soluzione economica, relativamente facile e veloce**
- **Consente massima flessibilità nella gestione interna degli indirizzi senza richiedere alcun permesso al proprio ISP**

Modulo 3: iptables

IPtables

- **IPtables è un software per implementare funzionalità di Packet Filtering, di Inspection, di NAT e di marking dei pacchetti.**
- **Presente in tutte le maggiori distribuzioni Linux a partire dal Kernel 2.4.**
- **È il successore di IPchains (Kernel 2.2.x)**
- **Qualora non fosse presente, i sorgenti sono reperibili all'URL** *http://ftp.netfilter.org/pub/iptables*

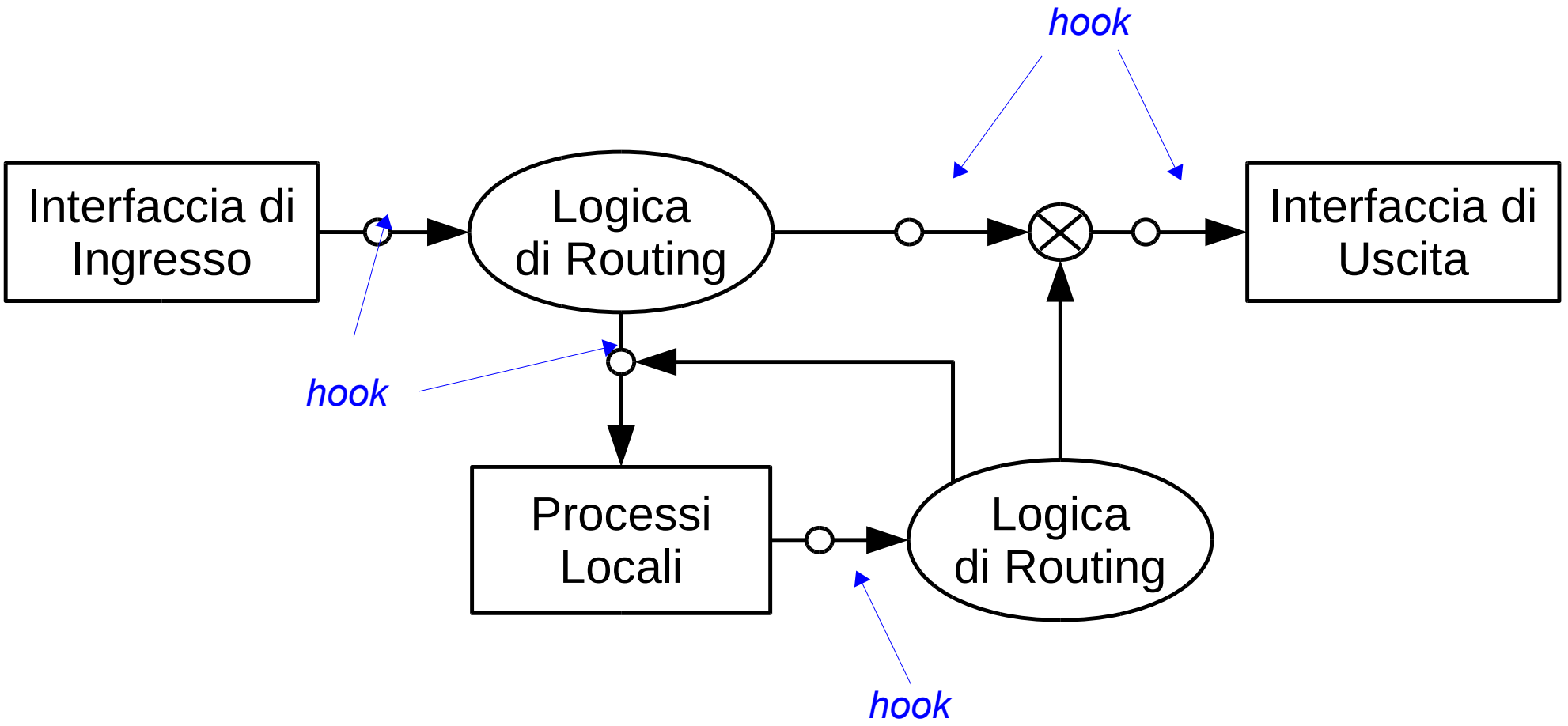
IPtables

- **IPtables consente la realizzazione di *regole*** per eseguire diversi tipi di operazioni sui pacchetti.
- Lo stack TCP/IP è gestito dal sistema operativo, quindi IPtables deve potersi interfacciare con il Kernel Linux.
- L'interfacciamento con il Kernel Linux, IPtables sfrutta il modulo Netfiler.
- **Tale modulo opera fornendo agganci (*hooks*)** al sistema operativo utilizzabili per intercettare i pacchetti in transito.

IPtables

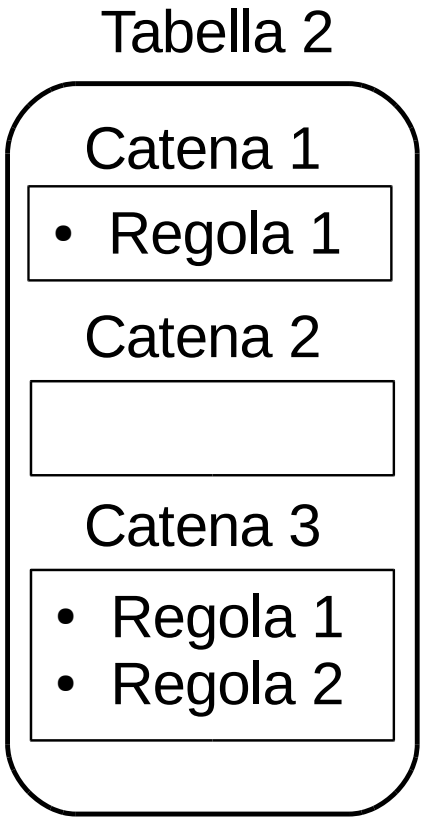
- **Ogni volta che un pacchetto attraversa un *hook***, Netfilter controlla se a quel determinato punto è stata assegnata una *funzione di gestione*:
  - se sì, il pacchetto viene passato alla funzione;
  - se no, il pacchetto passa all'hook successivo

IPtables


hook
Interfaccia di Ingresso
Logica di Routing
Interfaccia di Uscita
hook
Processi Locali
Logica di Routing
hook

IPtables

- **In sintesi, ogni *regola*:**
  - viene applicata in una certa fase di gestione dei pacchetti in base all'***hook*** su cui agisce
  - definisce su **quali** pacchetti deve essere applicata (e.g. IP, porta, iface)
  - definisce **come** modificare i pacchetti
- Le regole sono raggruppate in ***tabelle*** in base alle funzionalità alle quali si riferiscono.



- Regole appartenenti a una stessa tabella che “insistono” sullo stesso hook appartengono a una stessa ***catena***, e vengono eseguite in una seguenza ordinata.

NAT e PAT con IPtables

Sono presenti tre *tabelle:*

- **Filter**: per operazioni di filtraggio.
- **Mangle**: per le funzionalità di *marking* dei pacchetti e per effettuare modifiche ai campi TOS e TTL.
- **Nat**: per le funzioni di *Masquerading*, *Port Forwarding* e *Transparent Proxy*.

NAT e PAT con IPtables

La tabella nat prevede tre ***chain*** di default:

PREROUTING:

- **DNAT** pacchetti provenienti dall'esterno

OUTPUT:

- **DNAT** pacchetti generati localmente

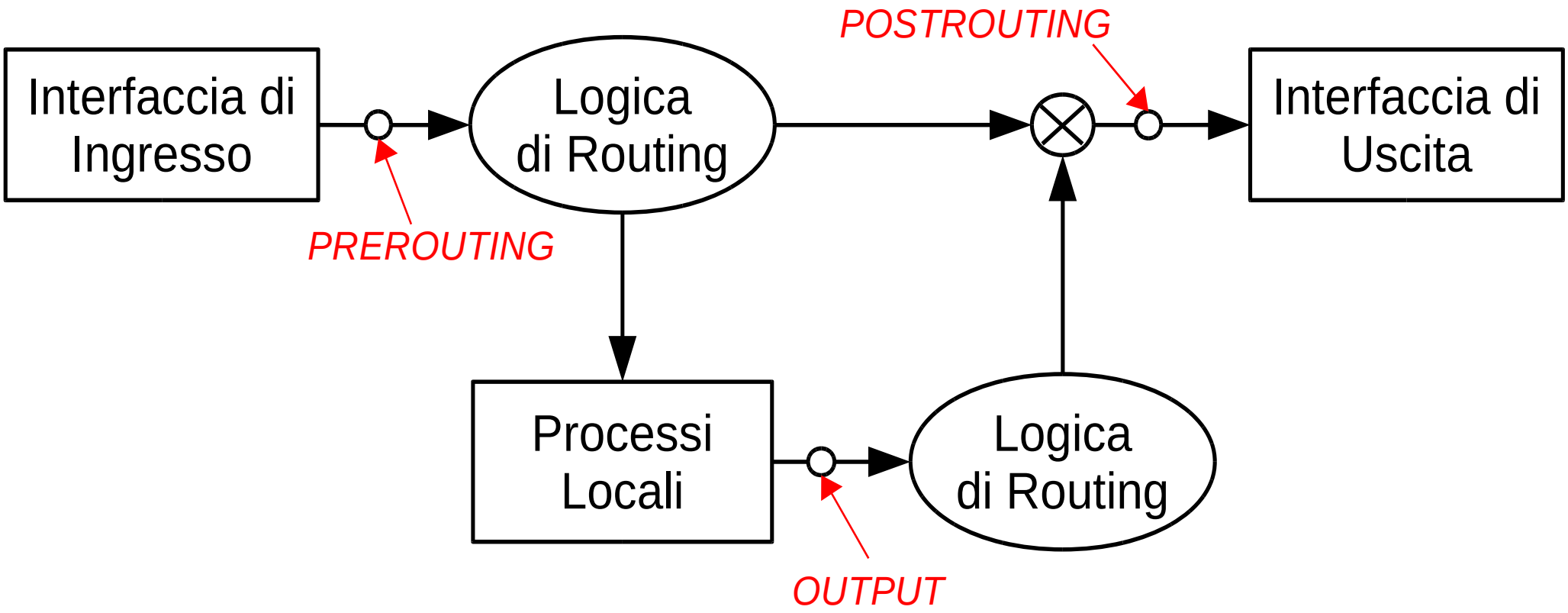
POSTROUTING:

- **SNAT** su tutti i pacchetti

Le regole impostate in ciascuna chain agiscono su uno specifico ***hook*** di netfilter

NAT e PAT con IPtables

Ogni chain agisce su un diverso hook di Netfilter



La tabella *nat* definisce tre chain di default:

PREROUTING, POSTROUTING e OUTPUT

NAT e PAT con IPtables

Scrivere regole di NAT e PAT richiede l'applicazione delle seguenti operazioni:

- **Source Network Address Translation (SNAT): alterazione della sorgente dei pacchetti, da applicare dopo l'applicazione delle regole di routing, ovvero su *POSTROUTING***
- **Destination Network Address Translation (DNAT): alterazione della destinazione dei pacchetti, da applicare prima dell'applicazione delle regole di routing, ovvero:**
  - su ***OUTPUT*** per pacchetti provenienti da processi locali;
  - su ***PREROUTING*** per quelli provenienti da interfacce di rete

Visualizzare le regole nat

 # iptables -t nat [-v] [-n] -L [<chain>] [--line-numbers]

Esempio:

```
root@r1:~# iptables -t nat -L -v -n

Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in   out  source      destination
    5   420 DNAT    all  --  eth1 *    0.0.0.0/0   0.0.0.0/0
to:192.168.1.1


Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in   out  source      destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in   out  source      destination


Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in   out  source      destination
    0     0 SNAT    all  --  *    eth1 0.0.0.0/0   0.0.0.0/0    to:1.2.3.4
```

Eliminare regole dalla tabella nat

Rimuovere regole:

```
# iptables -t nat -D <chain> <rule_number>
```

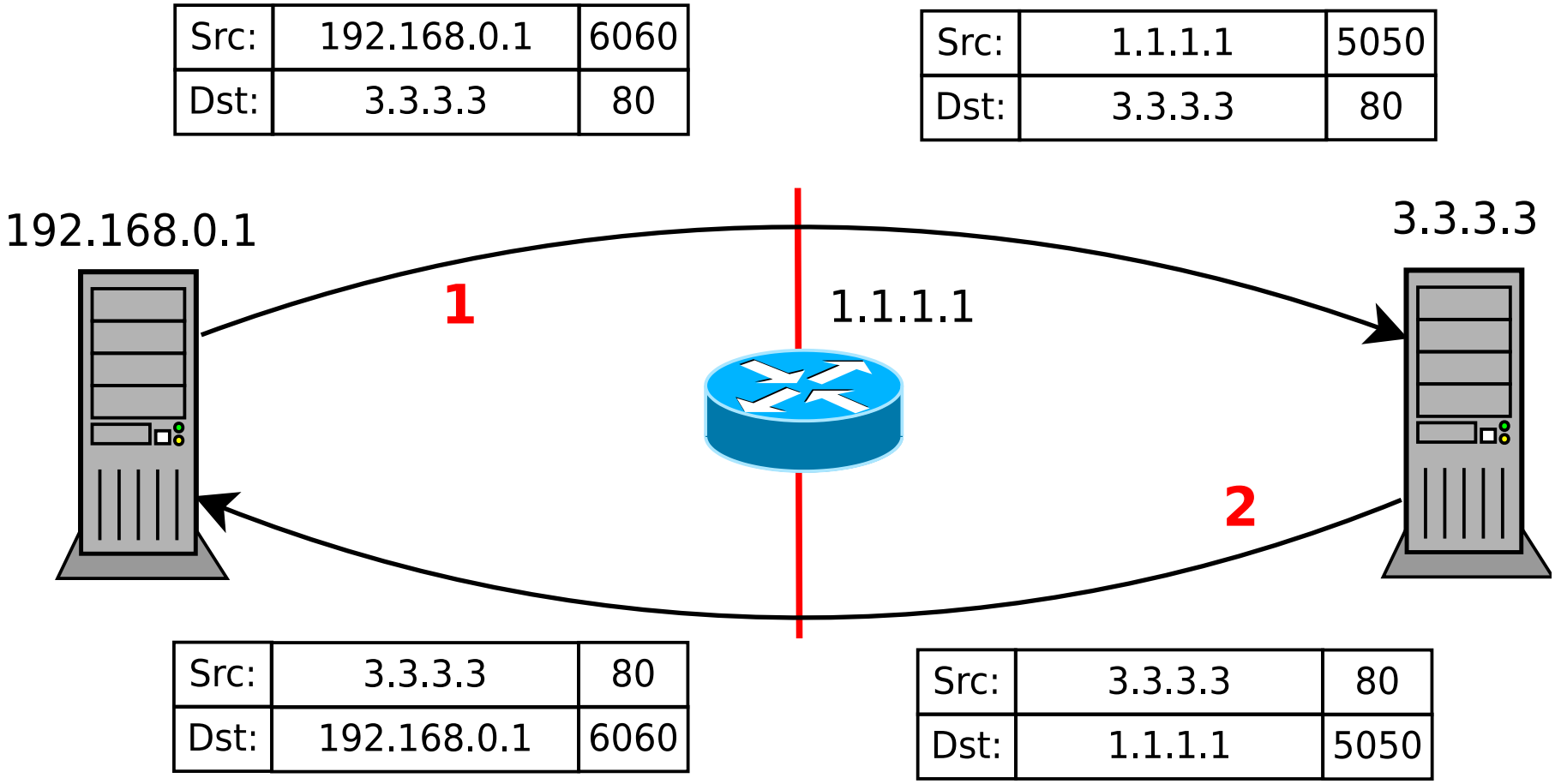
Esempio:

```
# iptables -t nat -D POSTROUTING 1
```

Elimina la prima regola di iptables della chain *POSTROUTING* nella tabella nat

SNAT: esempio

Consideriamo client interno alla rete private che si connette a un server esterno alla rete.



SNAT con IPtables: indirizzo statico

Aggiungere una regola di SNAT:

```
# iptables -t nat -A POSTROUTING \
      -o <output_iface> -j <action>
```

Esempio SNAT:

```
# iptables -t nat -A POSTROUTING \
      -o eth1 -j SNAT --to-source 1.2.3.4
```

Modifica tutti i pacchetti in uscita dall'interfaccia eth1 sostituendo l'IP sorgente con l'indirizzo IP 1.2.3.4.

SNAT con IPtables: indirizzo dinamico

Nel caso in cui l'indirizzo IP dell'interfaccia di rete da configurare sia dinamico, impiegare l'obiettivo MASQUERADE

```
# iptables -t nat -A POSTROUTING \
     -o <output_iface> -j MASQUERADE
```

Esempio SNAT MASQUERADE:

```
# iptables -t nat -A POSTROUTING \
     -o eth1 -j MASQUERADE
```

Masquerade modifica tutti i pacchetti in uscita dall'interfaccia eth1 sostituendo l'IP sorgente con l'indirizzo IP attualmente associato all'interfaccia di rete.

SNAT con IPtables: indirizzi multipli

Nel caso in cui si debba creare una corrispondenza fra multipli indirizzi pubblici e privati, è necessario specificare esplicitamente quali trasformare e come:

```
# iptables -t nat -A POSTROUTING \
      -o <output_iface> -s <private-ip> -j <action>
```

Esempio SNAT con indirizzi multipli:

```
# iptables -t nat -A POSTROUTING -o eth1 \
      -s 192.168.1.1 -j SNAT --to-source 1.2.3.4
# iptables -t nat -A POSTROUTING -o eth1 \
      -s 192.168.1.2 -j SNAT --to-source 1.2.3.5
```

Esegue le trasformazioni:

192.168.1.1 → 1.2.3.4
192.168.1.2 → 1.2.3.5

SNAT e DNAT con IPtables

IPtables gestisce i pacchetti IP e le connessione TCP in modo *stateful*, per cui ogni singola regola impostata in realtà può comportare molte altre azioni da parte del software.

Nel caso di SNAT, ad esempio, IPtables ritrasforma automaticamente anche gli *indirizzi IP di destinazione* dei pacchetti di risposta a pacchetti su cui precedentemente era stato effettuato SNAT***!***

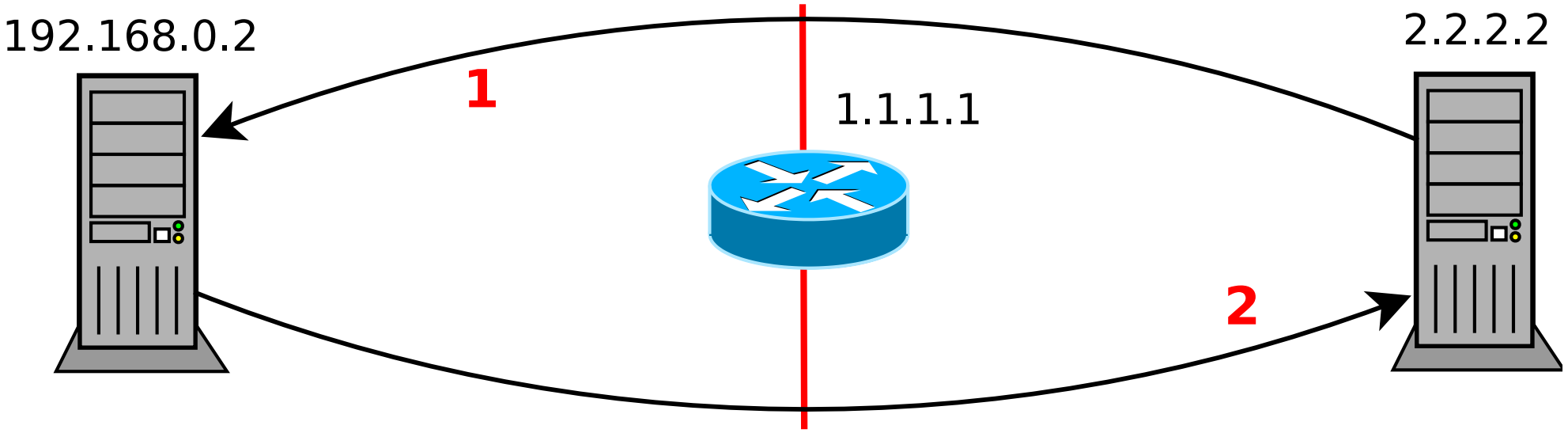
Nel caso in cui un pacchetto venga inviato dall'esterno, IPtables non può applicare alcuna regola legata a SNAT, bensì è necessario ricorrere a regole di **DNAT**, ad esempio per applicare politiche di ***port forwarding***.

DNAT: esempio

Consideriamo uno scenario in cui un client esterno si connette a un server interno *con ip privato*.

Src:	2.2.2.2	50000
Dst:	192.168.0.2	80

Src:	2.2.2.2	50000
Dst:	1.1.1.1	80



Src:	192.168.0.2	80
Dst:	2.2.2.2	50000

Src:	1.1.1.1	80
Dst:	2.2.2.2	50000

Address Translation con IPtables

Aggiungere una regola di DNAT:

```
# iptables -t nat -A PREROUTING -j DNAT \
      -i <iface> -d <public_ip> \
      --to-destination <private_ip>
```

Esempio DNAT:

```
# iptables -t nat -A PREROUTING -j DNAT \
      -i eth1 -d 1.2.3.4 \
      --to-destination 192.168.1.30
```

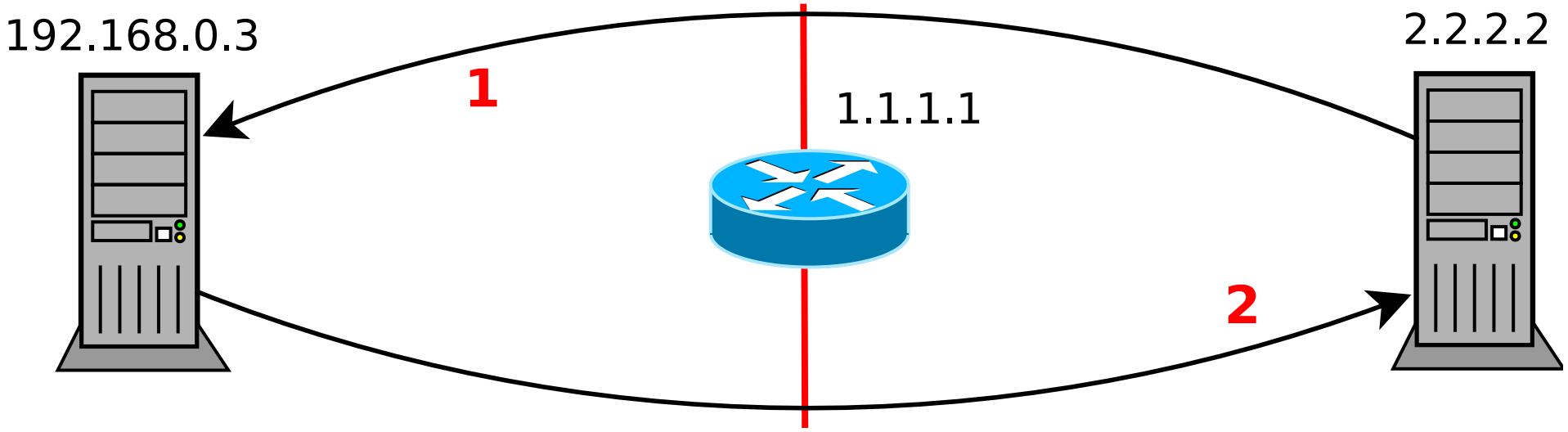
Modifica i pacchetti in entrata dall'interfaccia eth1 e diretti all'indirizzo IP 1.2.3.4 sostituendo l'IP destinazione con l'indirizzo IP 192.168.1.30

DNAT: esempio

Consideriamo un secondo scenario in cui un client esterno si connette a un server interno *con ip privato*, e la porta utilizzata dal server interno è diversa da quella resa disponibile dal gateway della rete.

Src:	2.2.2.2	50000
Dst:	192.168.0.3	8888

Src:	2.2.2.2	50000
Dst:	1.1.1.1	8080



Src:	192.168.0.3	8888
Dst:	2.2.2.2	50000

Src:	1.1.1.1	8080
Dst:	2.2.2.2	50000

Port forwarding con IPtables

Aggiungere una regola di DNAT:

```
# iptables -t nat -A PREROUTING -j DNAT \
      -i <iface> -d <public_ip>  \
      -p <protocol> --dport <port> \
      --to-destination <private_ip:port>
```

Esempio DNAT:

```
# iptables -t nat -A PREROUTING -j DNAT \
      -i eth1 -d 1.2.3.4 \
      -p tcp --destination-port 8000 \
      --to-destination 192.168.1.30:80
```

Modifica i pacchetti in entrata dall'interfaccia eth1, diretti all'IP 1.2.3.4, con protocollo tcp e porta 8000, impostando indirizzo di destinazione 192.168.1.30 e porta di destinazione 80

Rendere permanenti le modifiche

***Opzione 1*:**

impostare le regole di IPtables nel file */etc/network/interfaces* con il comando *post-up*, alla stregua delle regole di routing

***Opzione 2*:**

usare iptables-persistent salvando le regole sul file ***/etc/iptables/rules.v4*** **(per ipv4) tramite i comandi**

iptables-save → stampa a standard output le regole di iptables attualmente configurate

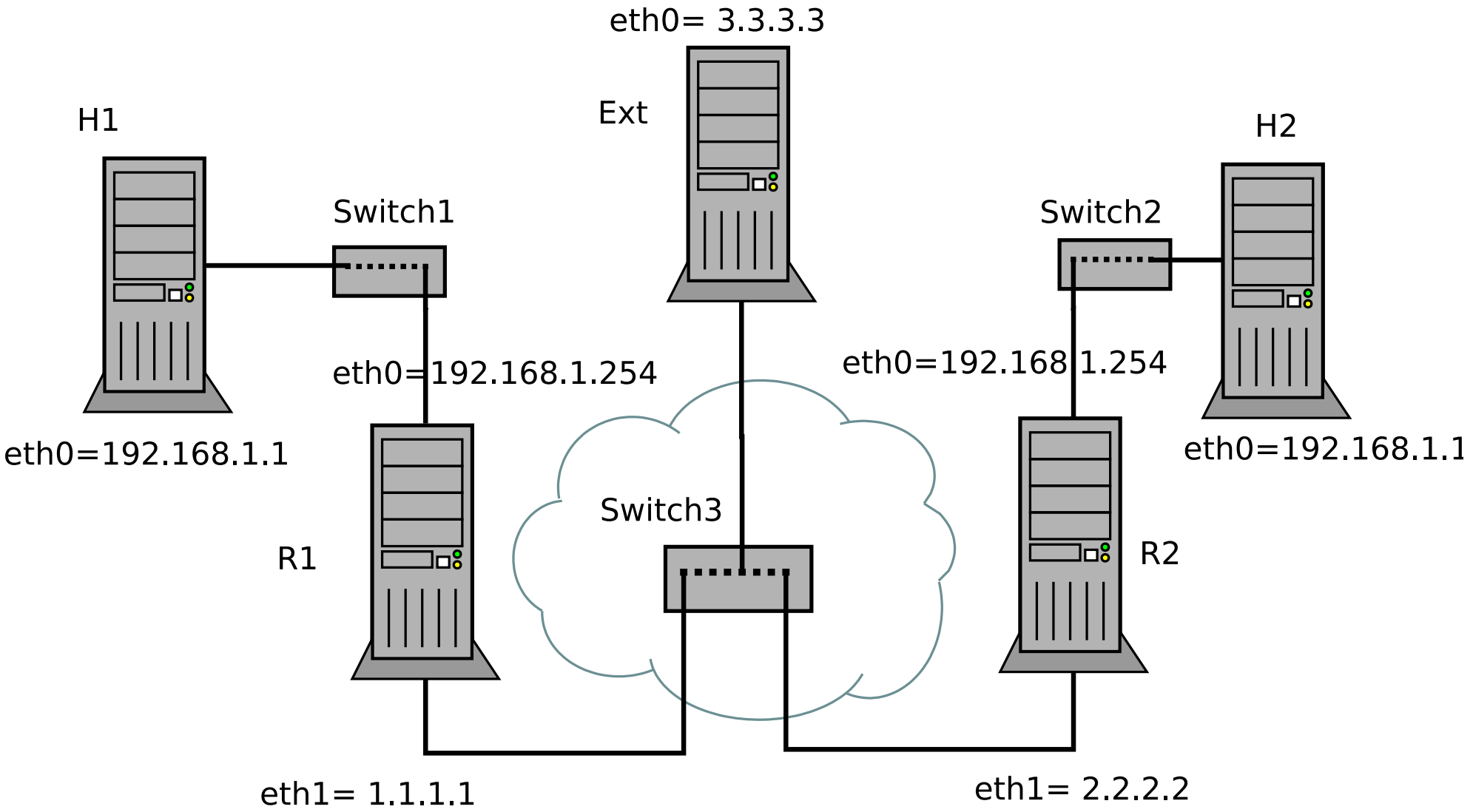
iptables-restore → ripristina le regole di iptables inviate a standard input

- Per salvare/caricare la configurazione:

```
iptables-save > /etc/iptables/rules.v4

iptables-restore < /etc/iptables/rules.v4
```

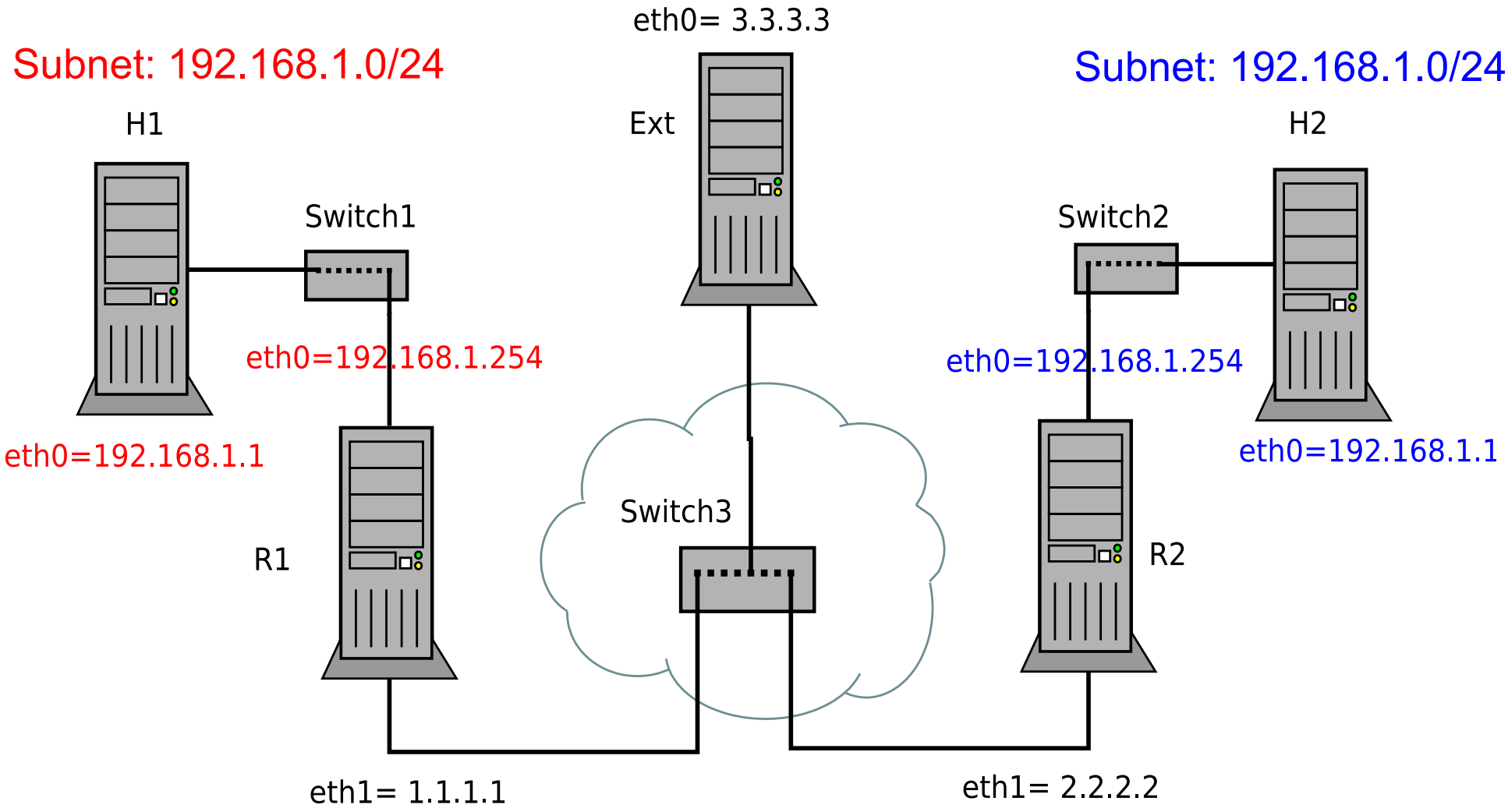
Modulo 4: Esercitazione

Esercitazione



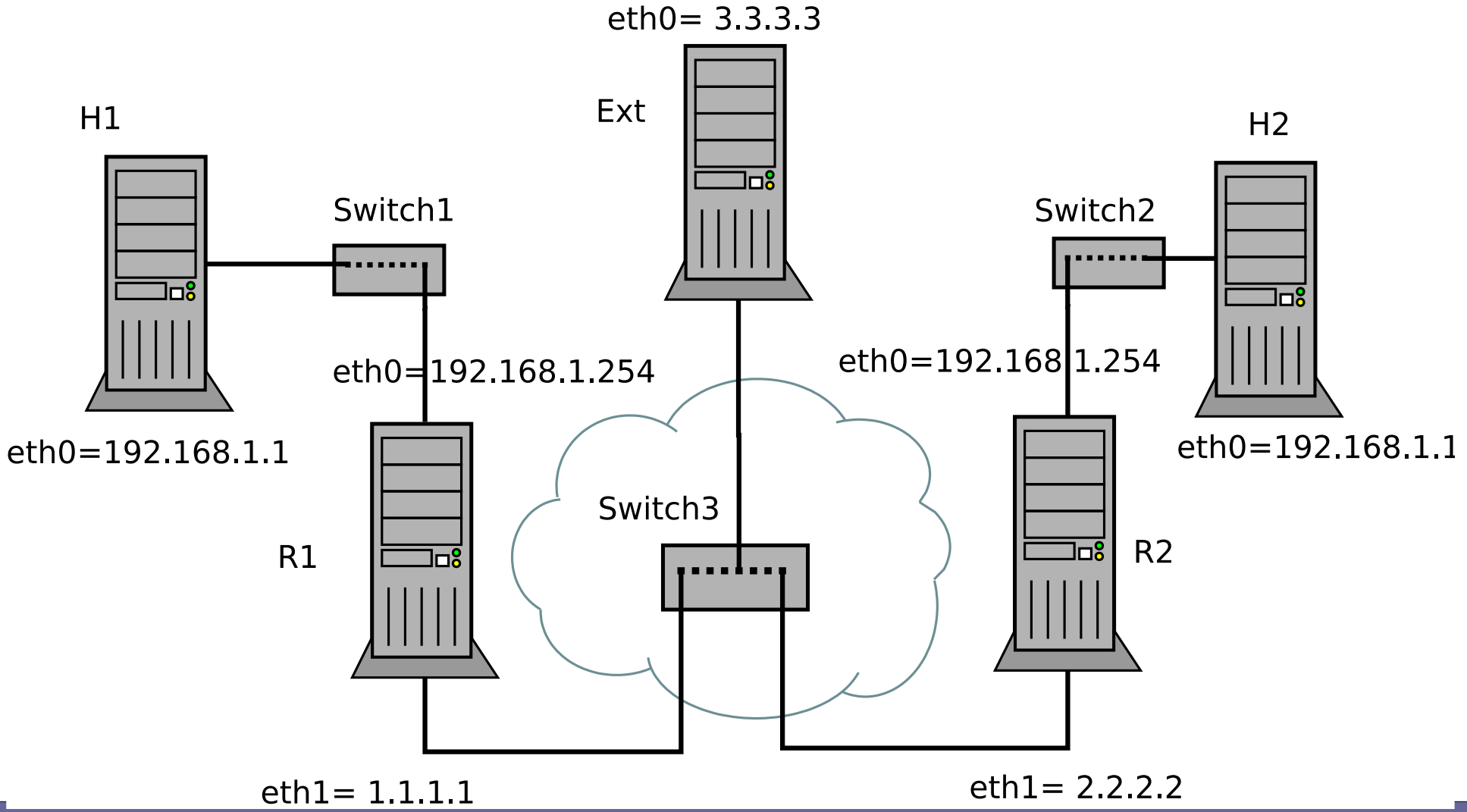
Scenario: Network Address Translation

Indirizzi IP Internet duplicati? **NO**, si impiega il "**NATting**"



Esercizio

Configurare la rete iniziale per permettere la comunicare fra i nodi, considerando che S3 simula la rete Internet e i gateway R1 ed R2 devono applicare Source Natting sul traffico in uscita dalle rispettive reti.



Suddivisione dei compiti

- **Step 1**
  - Configurazione del networking a livello IP
  - Verifica di cosa funziona
- **Step 2**
  - Configurazione NAT
  - Verifica funzionamento complessivo

File /etc/network/interfaces

- **Step 1: configurazione rete sui nodi**
- **Ext**

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 3.3.3.3/32
    post-up route add -host 1.1.1.1 dev eth0
    post-up route add -host 2.2.2.2 dev eth0
```

File /etc/network/interfaces

- **H1**

```
auto eth0
iface eth0 inet static
    address 192.168.1.1/24
    gateway 192.168.1.254
```

- **H2**

```
auto eth0
iface eth0 inet static
    address 192.168.1.1/24
    gateway 192.168.1.254
```

File /etc/network/interfaces

- **R1**

```
auto eth0
iface eth0 inet static
    address 192.168.1.254/24

auto eth1
iface eth1 inet static
    address 1.1.1.1/32
    post-up route add -host 3.3.3.3 dev eth1
    post-up route add -host 2.2.2.2 dev eth1
```

- **N.B. ricordarsi di abilitare l'IP forwarding**

```
sysctl -w net.ipv4.ip_forward=1
```

File /etc/network/interfaces

- **R2**

```
auto eth0
iface eth0 inet static
    address 192.168.1.254/24

auto eth1
iface eth1 inet static
    address 2.2.2.2/32
    post-up route add -host 3.3.3.3 dev eth1
    post-up route add -host 1.1.1.1 dev eth1
```

- **N.B. ricordarsi di abilitare l'IP forwarding**

```
sysctl -w net.ipv4.ip_forward=1
```

Verificare il lavoro fatto

- **Da H1 facciamo ping verso 3.3.3.3**
  - Non si ottiene risposta
  - Non succede davvero niente?
- **Verifica del traffico**
  - R1$ tcpdump -i eth0
  - R1$ tcpdump -i eth1
  - Ext$ tcpdump -i eth0
- **Il flusso di richieste H1 → Ext funziona**
- **Perché non arrivano le risposte?**
  - Ragionare sulla tabella di routing di Ext

- **Step 2: configurazione NAT**
- **Serve Source NAT dinamico su R1 e R2**
- **Su R1:**
  - R1$ iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
- **Su R2:**
  - R2$ iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
- **Possiamo inserire i comandi direttamente nei file di configurazione**

File /etc/network/interfaces

- **R1**

```
auto eth0
iface eth0 inet static
    address 192.168.1.254/24

auto eth1
iface eth1 inet static
    address 1.1.1.1/32
    post-up route add -host 3.3.3.3 dev eth1
    post-up route add -host 2.2.2.2 dev eth1
    post-up iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

File /etc/network/interfaces

- **R2**

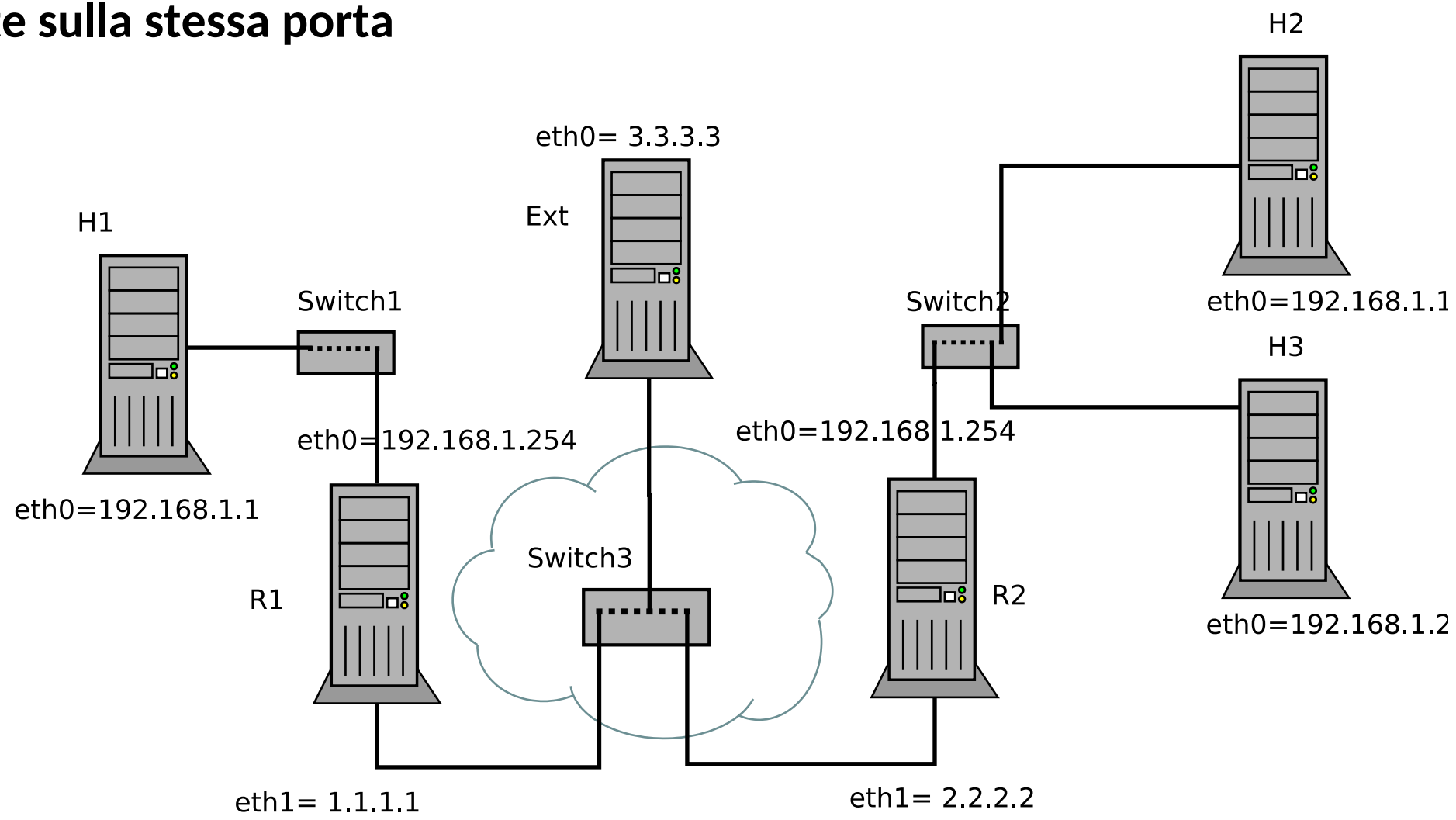
```
auto eth0
iface eth0 inet static
    address 192.168.1.254/24

auto eth1
iface eth1 inet static
    address 2.2.2.2/32
    post-up route add -host 3.3.3.3 dev eth1
    post-up route add -host 1.1.1.1 dev eth1
    post-up iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

Verificare il lavoro fatto

- **Da H1 e/o H2:**
  - Ping verso 3.3.3.3
  - Ci aspettiamo che il nodo sia contattabile
- **Su R1 e/o R2:**
  - Verifichiamo con tcpdump il traffico sull'interfaccia eth0
  - Confrontiamo con il traffico su eth1

Esercizio

Aggiungere un host H3 in cui è in esecuzione un servizio sulla porta UDP 7777. Fare in modo che il servizio sia contattabile dagli host esterni alla rete sulla stessa porta



Configurazioni

- **H3: file /etc/network/interfaces**

```
auto eth0
iface eth0 inet static
    address 192.168.1.2/24
    gateway 192.168.1.254
```

- **R2: aggiungiamo una nuova regola di NAT**

```
iptables -t nat -A PREROUTING -i eth1 -p udp\
    --dport 7777 -j DNAT --to-destination 192.168.1.2
```

Verifica funzionamento

- **Su H3 mettiamo in ascolto un server**

nc -l -u -p 7777

- **Da H1 contattiamo il server**

nc -u 2.2.2.2 7777

- **I dati inviati da H1 dovrebbero arrivare ad H3 ed essere visualizzati**

Verifica funzionamento

- **Osserviamo il traffico su R1**
  - Su interfaccia eth0
  - Su interfaccia eth1
- **Osserviamo il traffico su R2**
  - Su interfaccia eth0
  - Su interfaccia eth1
- **Si osservi la combinazione di SNAT e DNAT**