

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)

NAT router (NAT box)

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

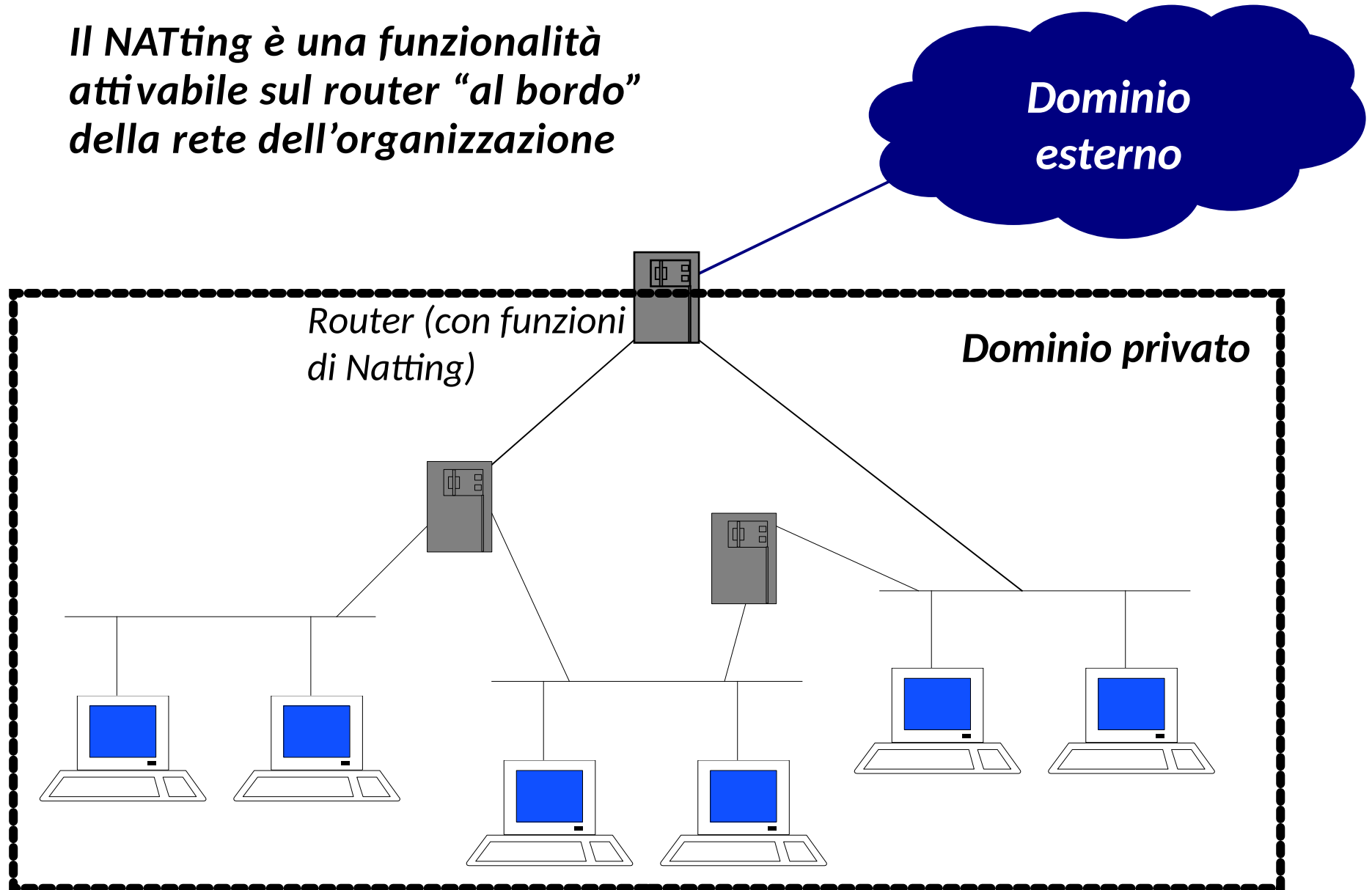
- Mappa gli indirizzi IP tra due domini (interno-esterno)

indirizzi locali \leftarrow : \rightarrow indirizzi IP globali

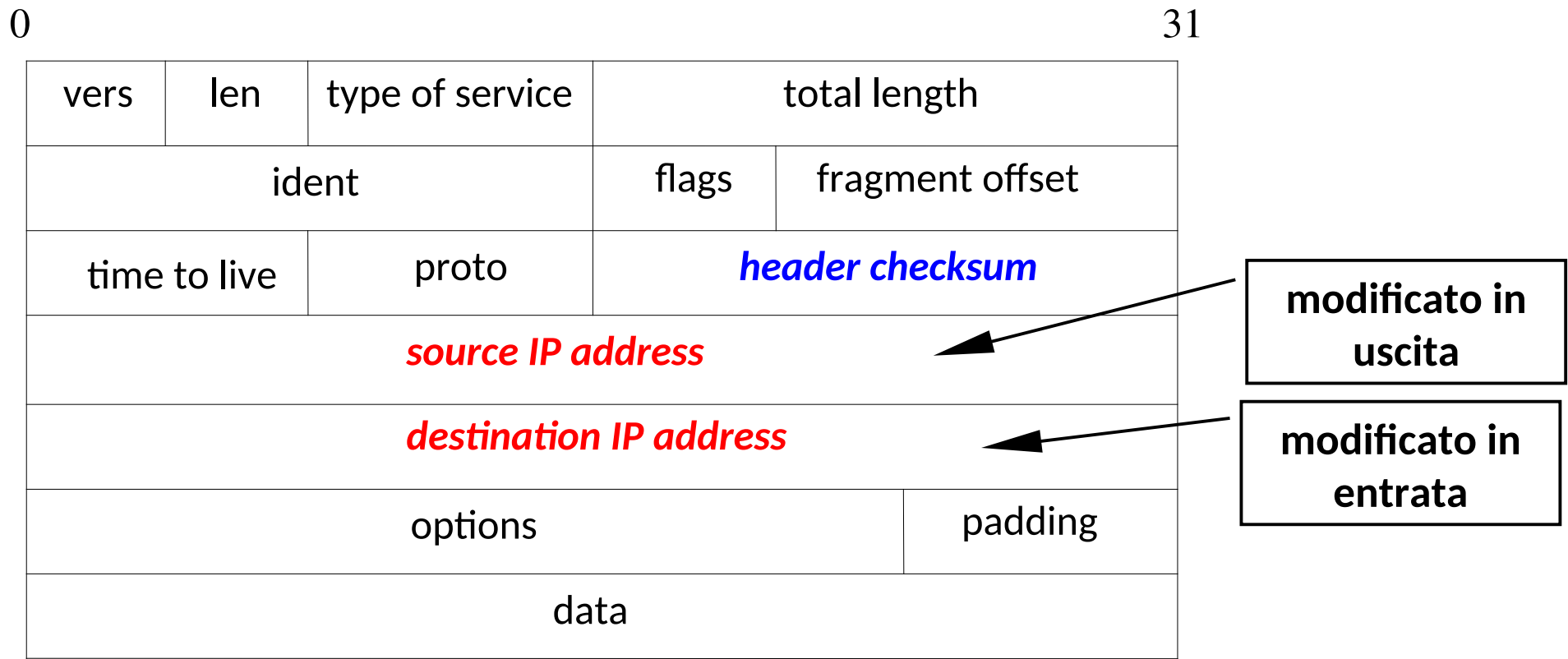
- **Nota:** *il meccanismo può essere in realtà utilizzato in tutti i contesti in cui un router mette in comunicazione due reti con **spazi di indirizzamento IP separati***
- Garantisce la trasparenza del routing tra gli *end system*
- “Moltiplica” le possibilità di interconnessioni di host di una organizzazione (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

NATting per reti semi-private

Il NATting è una funzionalità attivabile sul router “al bordo” della rete dell’organizzazione

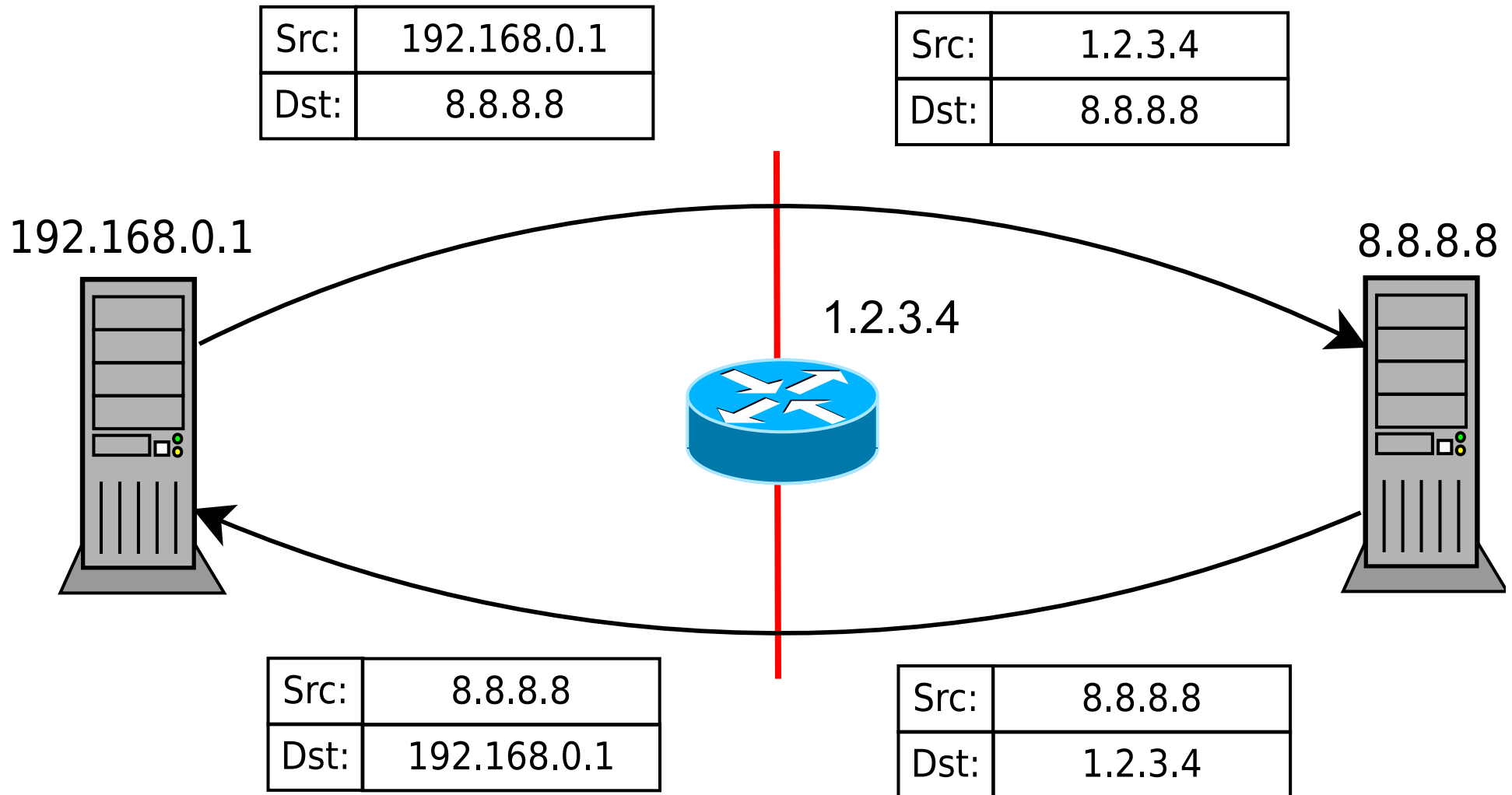


NAT: modifica del datagram IP



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

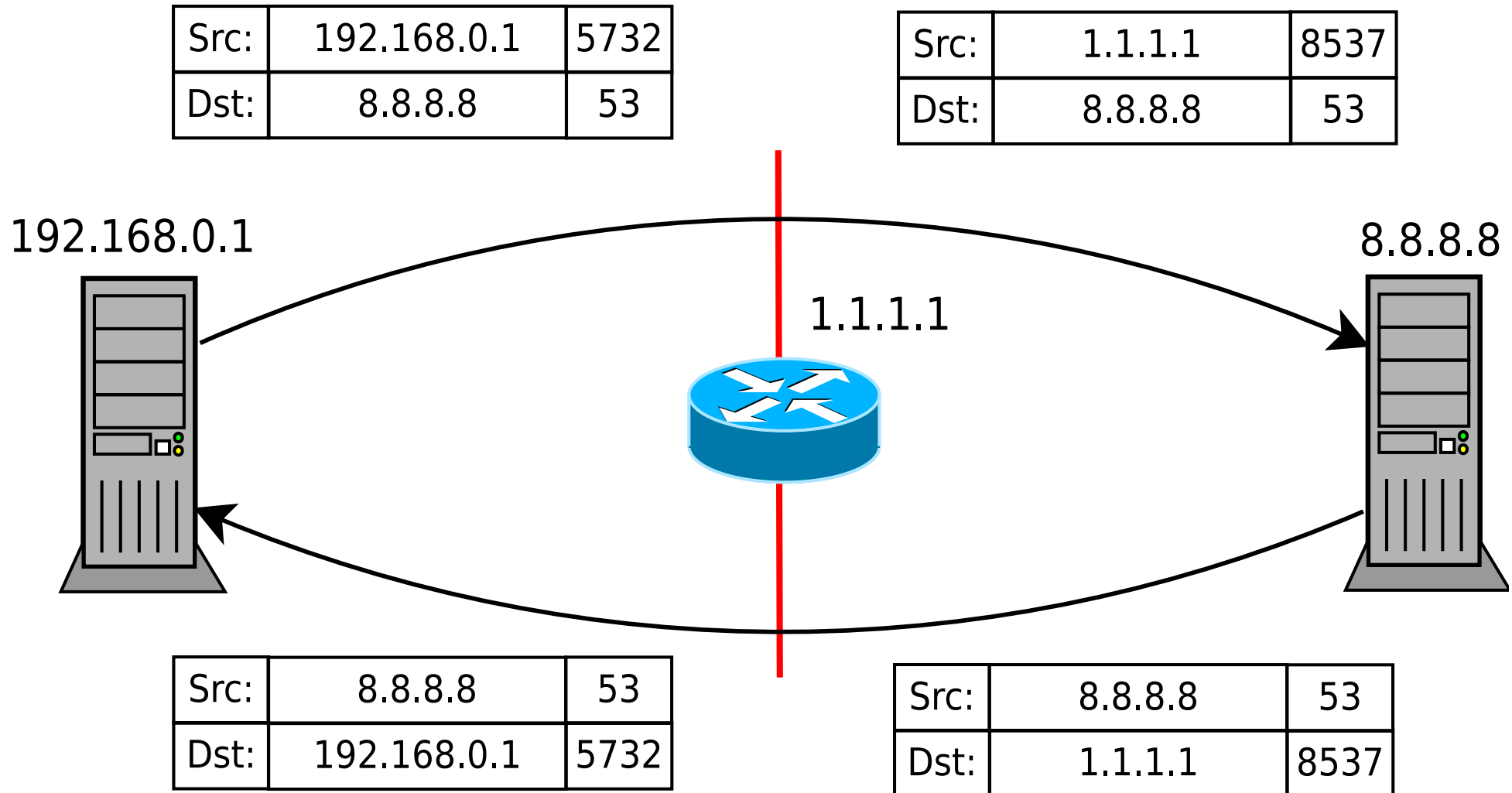
Network Address Translation (NAT): esempio



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 con Port Address Translation (PAT): esempio



PAT: modifica del datagramma (IP+UDP)

0

31

vers	len	type of service	total length	
ident			flags	fragment offset
time to live	proto		<i>header checksum</i>	
<i>source IP address</i> ←				
<i>destination IP address</i> ←				
options			padding	
<i>source port</i> ←		<i>destination port</i> ←		
length		<i>checksum</i>		
data				

modificato in uscita

modificato in entrata

modificato in DNAT

modificato in SNAT

Binding degli indirizzi

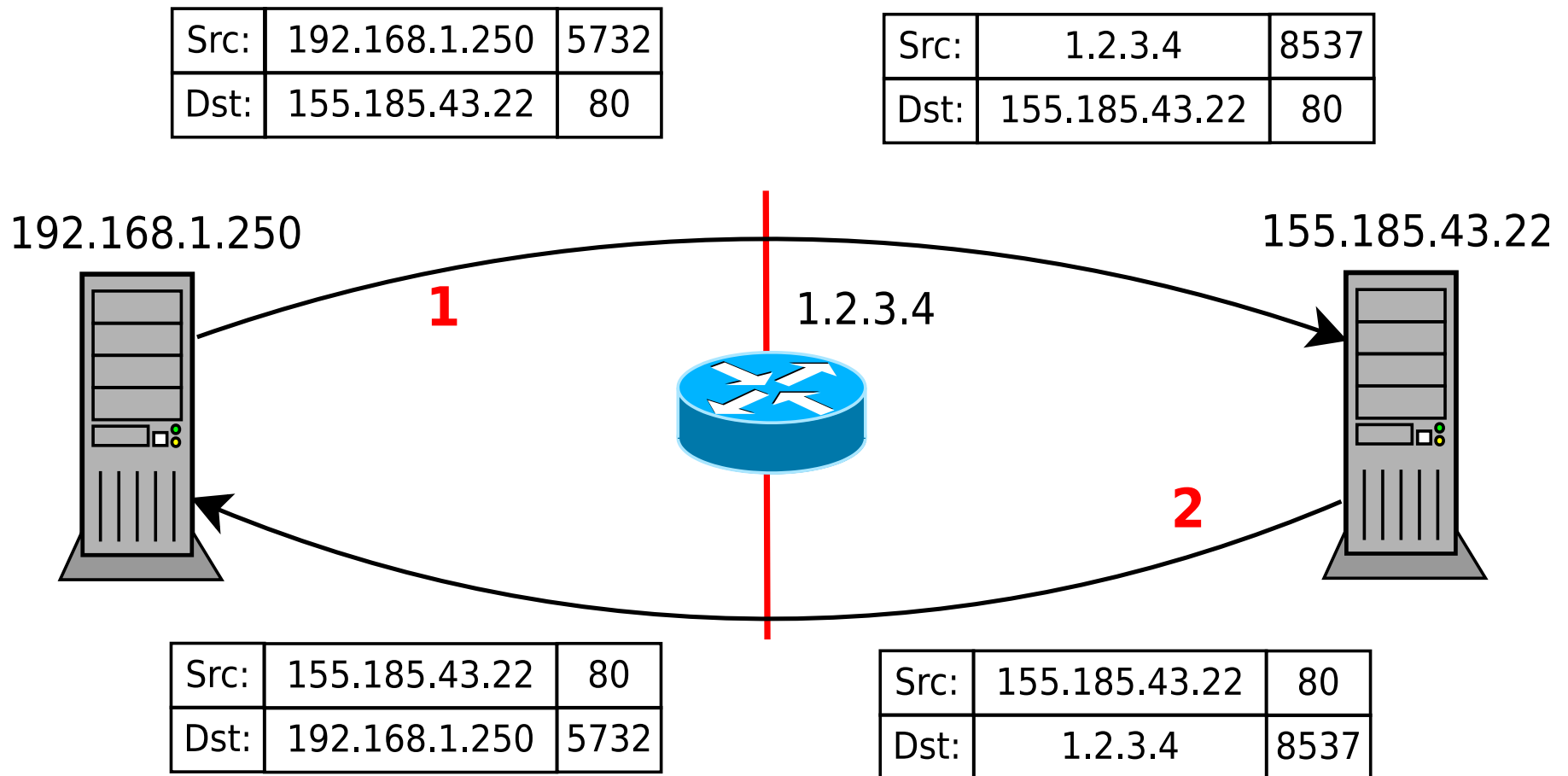
Il router gestisce una corrispondenza (*binding*) tra gli indirizzi dei due domini tramite una TABELLA che mantiene una riga per ciascuna “connessione aperta”:

- **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

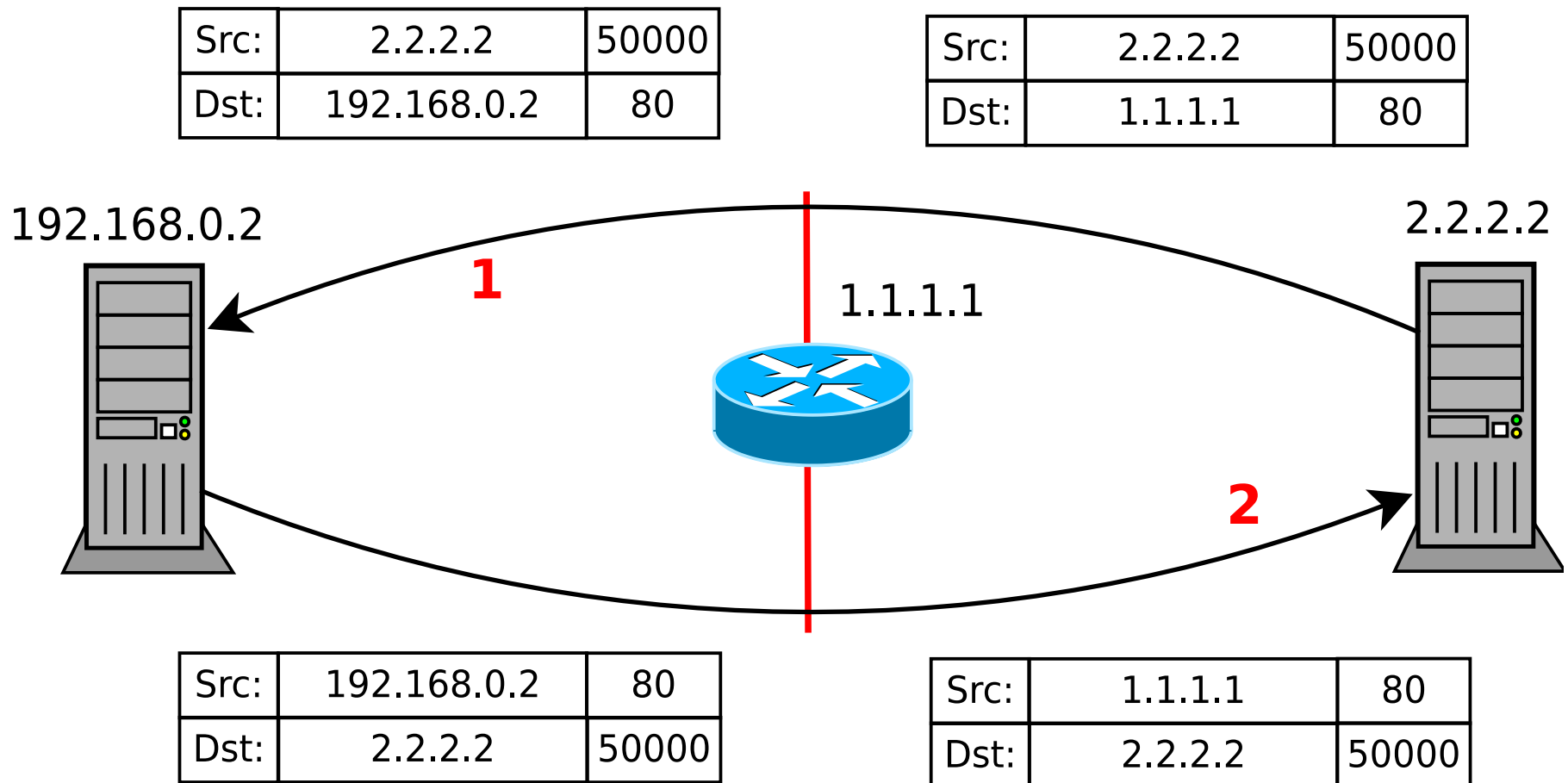
Source NAT (SNAT): esempio

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



Destination NAT (DNAT): esempio (1)

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

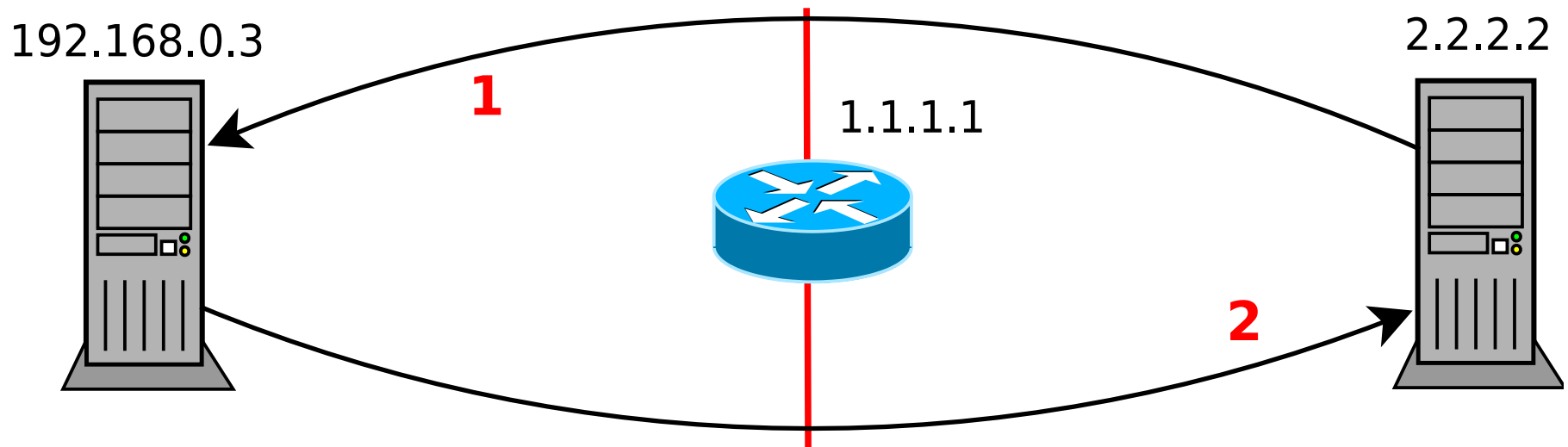


DNAT: esempio (2)

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

Natting: contro

- Distrugge la semantica della comunicazione *end-to-end* in quanto gli host interni non possono essere raggiunti dall'esterno:
 - configurazioni ad-hoc per garantire la raggiungibilità di server interni alle reti private;
 - 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 host interni non possono essere raggiunti 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

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>
- È stato succeduto da *nftables*, e attualmente entrambi i due tool sono solitamente presenti sui sistemi Linux moderni
- Alcune regole molto semplici di *nat* possono anche essere realizzate tramite *iproute2*

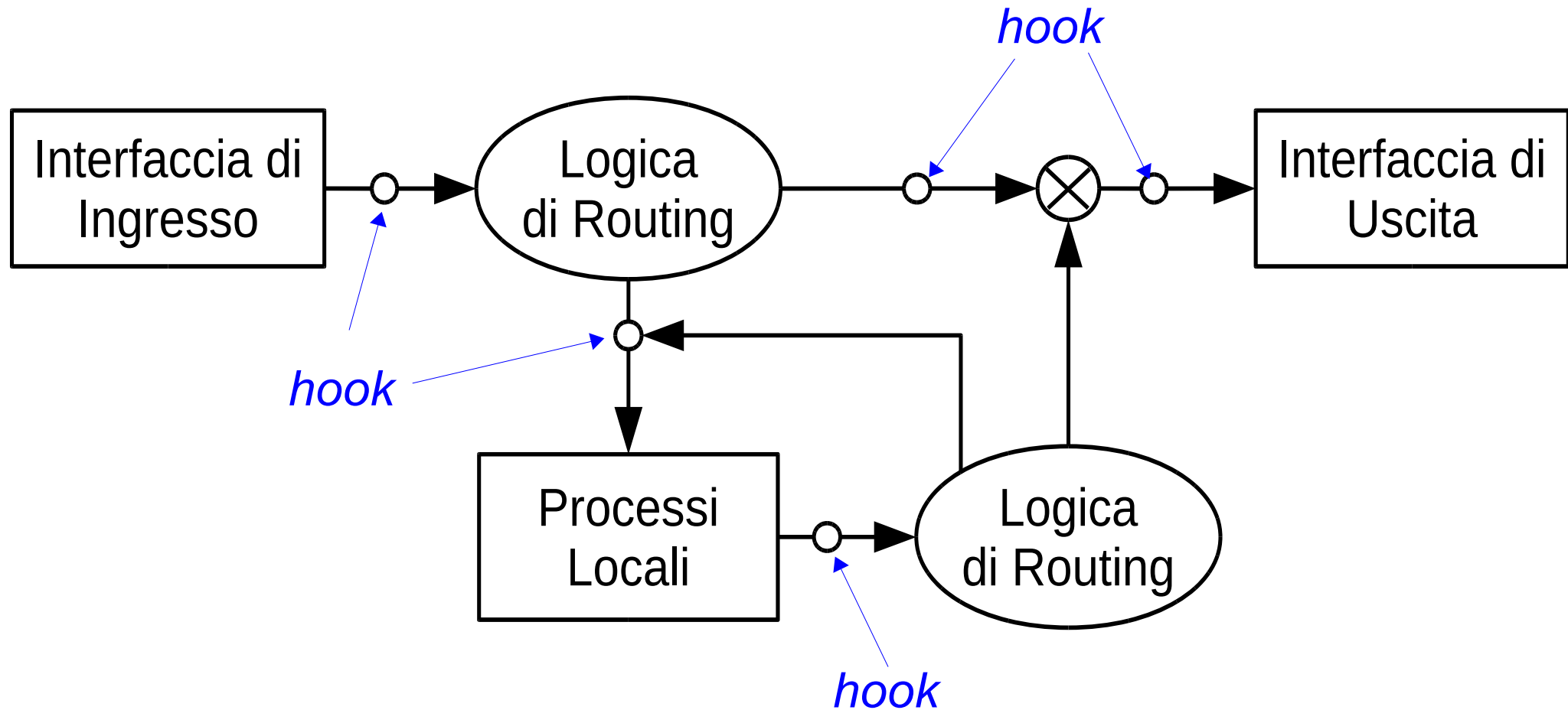
IPtables (2)

- 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 Netfilter.
- Tale modulo opera fornendo agganci (**hooks**) al sistema operativo utilizzabili per intercettare i pacchetti in transito.

IPtables (3)

- 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

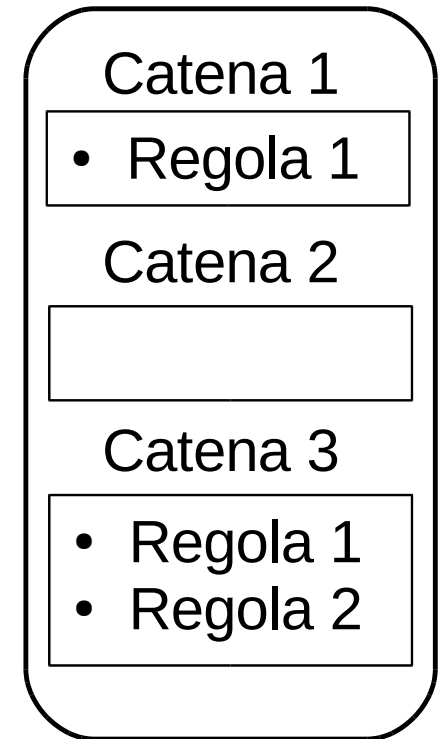
IPTables (4)



IPtables (5)

- 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 sequenza ordinata.

Tabella 2



NAT e PAT con IPtables (1)

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 (2)

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

PREROUTING:

- **DNAT** pacchetti provenienti dall'esterno

OUTPUT:

- **DNAT** pacchetti generati localmente

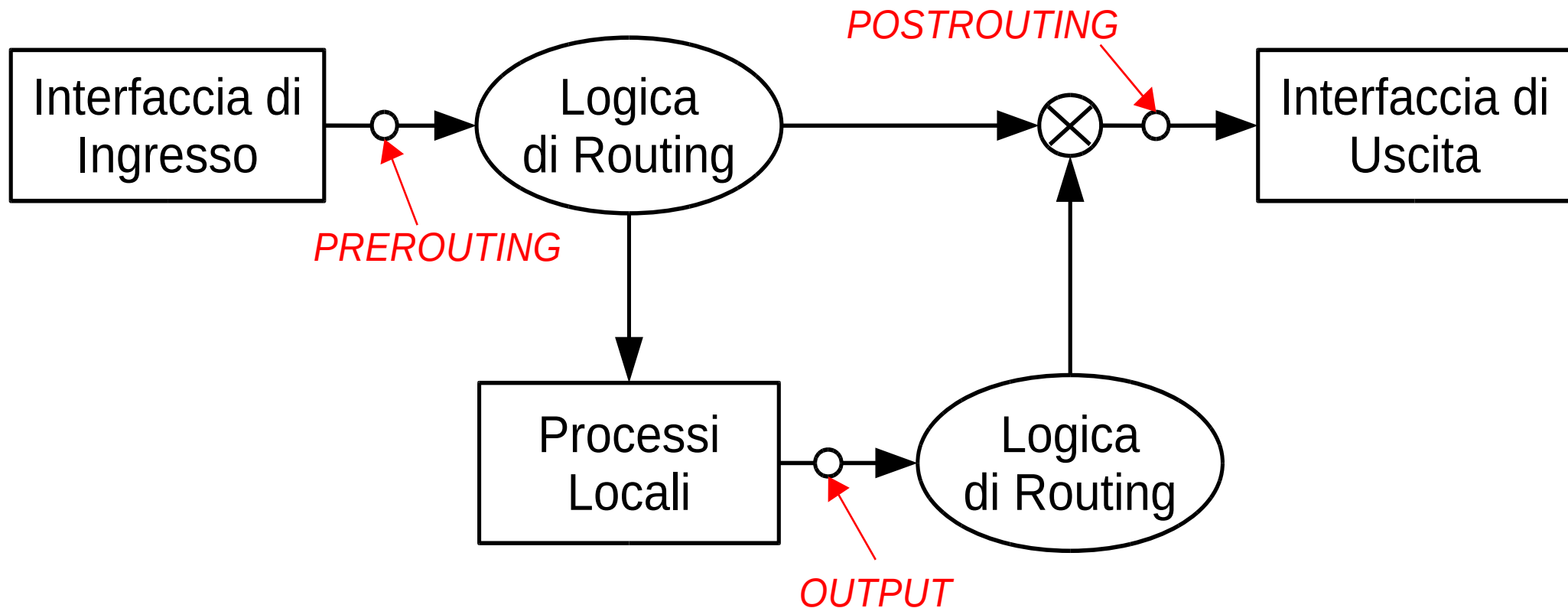
POSTROUTING:

- **SNAT** su tutti i pacchetti

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

NAT e PAT con IPtables (3)

La tabella *nat* definisce tre chain di default:
PREROUTING, *POSTROUTING* e *OUTPUT*



Ogni chain agisce su un diverso hook di Netfilter

NAT e PAT con IPtables (4)

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 della tabella nat (1)

```
# 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>
```

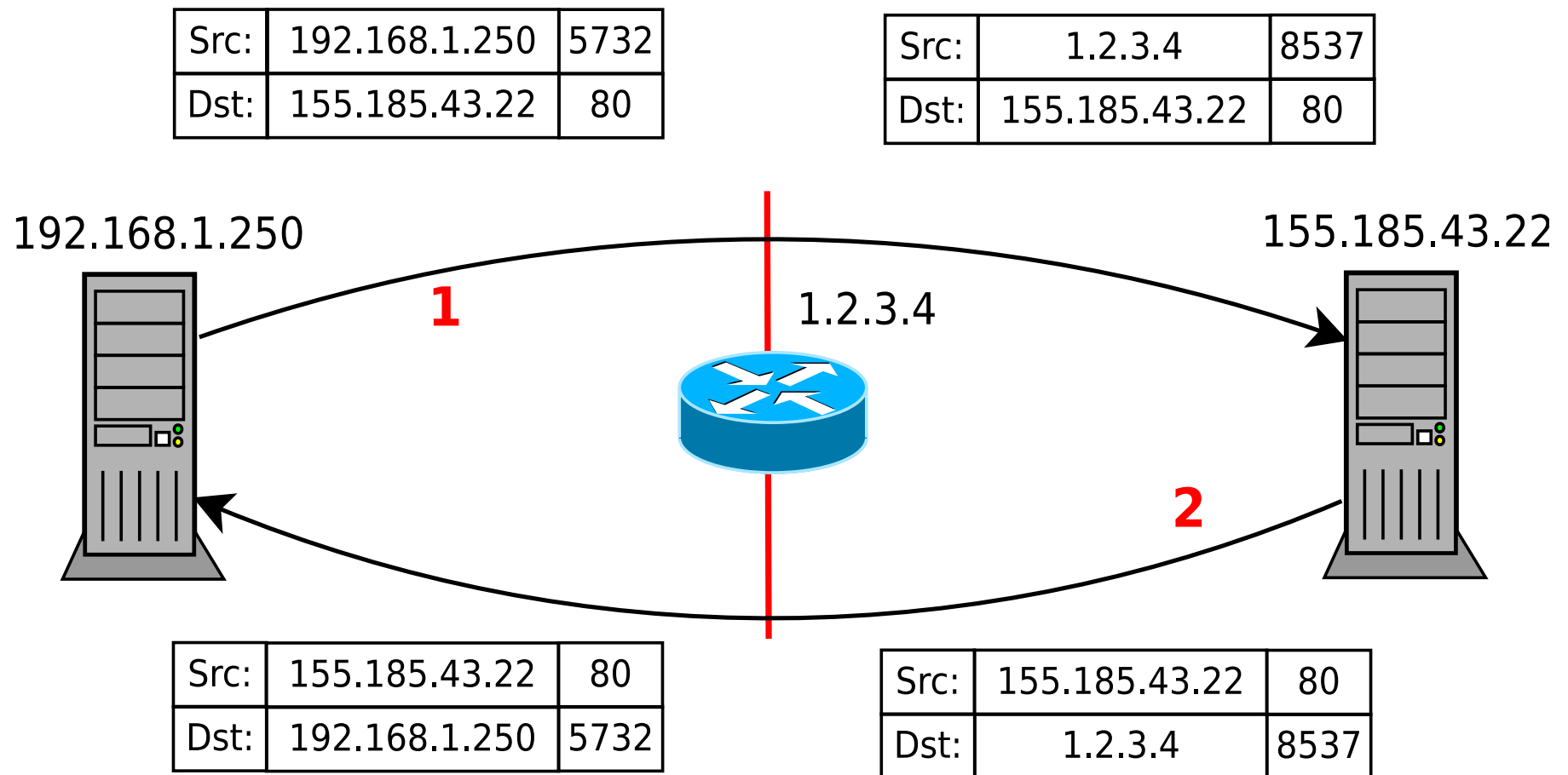
Esempio:

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

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

SNAT: esempio

Consideriamo lo scenario precedente, in cui un client interno alla rete private 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

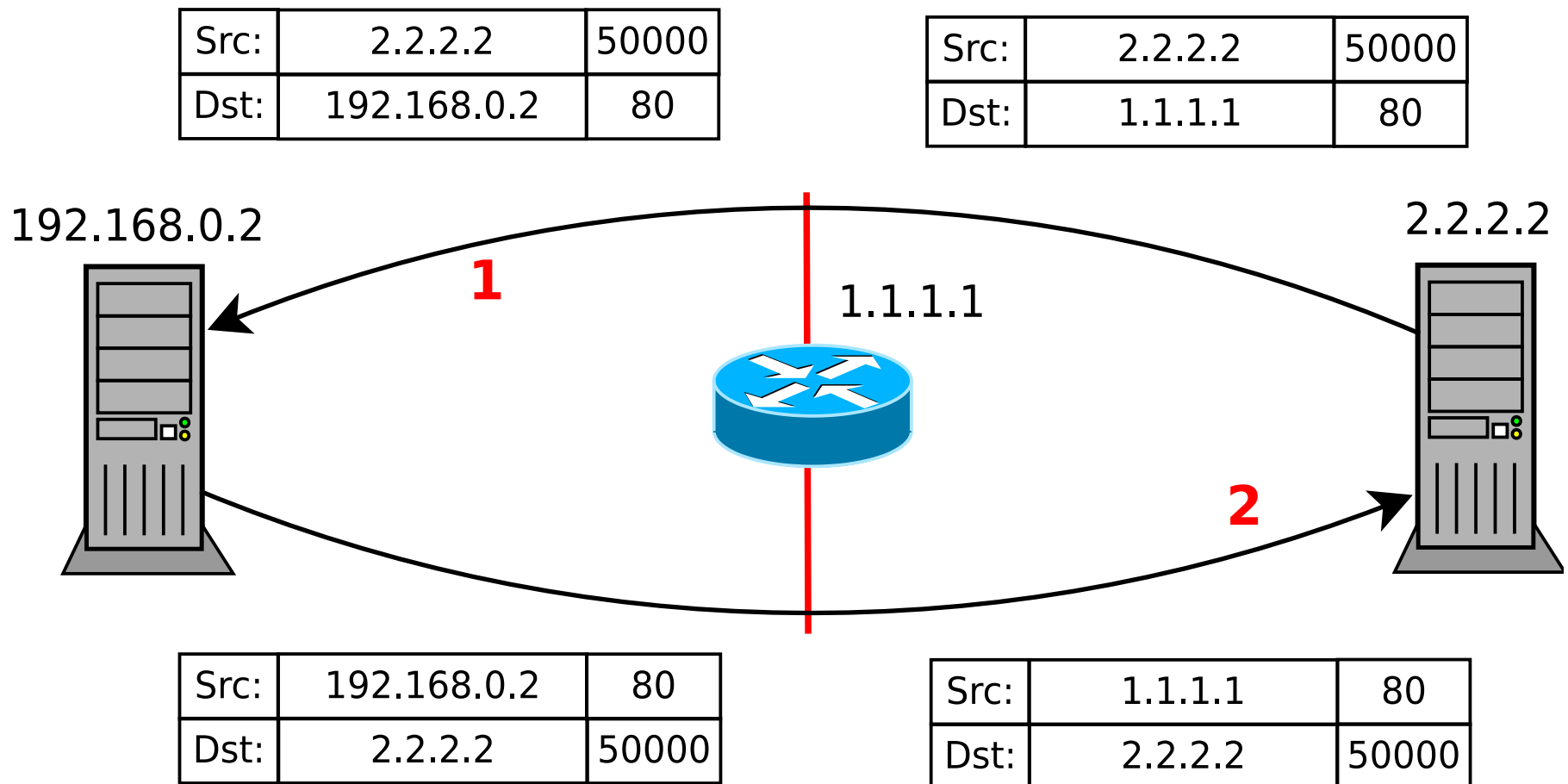
IPtables gestisce i pacchetti IP e le connessioni TCP in modo **stateful**, per cui ogni singola regola impostata in realtà può comportare molte altre azioni da parte del software.

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 (1)

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



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
```

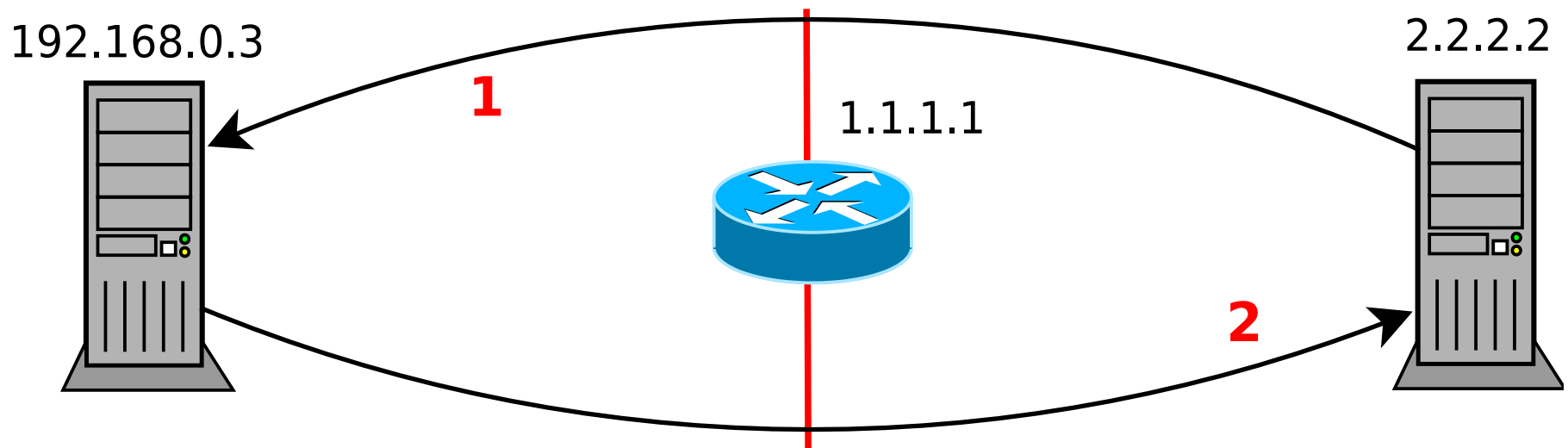
Modifica i pacchetti in entrata dall'interfaccia eth1 e diretti all'indirizzo IP 1.2.3.4 sostituendo l'IP destinazione con l'indirizzo IP 192.168.1.30

DNAT: esempio (2)

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 di IPtables

Opzione 1 (sconsigliata):

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

Opzione 2 (consigliata):

usare **netfilter-persistent**

netfilter-persistent save → memorizza le regole correnti

netfilter-persistent reload → ricarica a runtime le regole persistenti (utile per ripristinare una configurazione “stabile” in seguito ad alcuni tentativi di modifica a runtime)

- Note:
 - netfilter-persistent è un servizio di sistema (gestibile con `systemctl`)
 - le regole sono memorizzate nel file `/etc/iptables/rules.v4` e `/etc/iptables/rules.v6` rispettivamente per IPv4 e IPv6