Laboratorio di
Reti Informatiche

Corso di Laurea Triennale in Ingegneria Informatica

A.A. 2016/2017

Ing. Niccolò Iardella

niccolo.iardella@unifi.it

Esercitazione 7

Configurazione del firewall

Programma di oggi

- Concetto di firewall e packet filtering
- netfilter/iptables
- NAT/PAT

Firewall

- Necessità: proteggere le reti e i computer connessi a Internet da accessi indesiderati e *malware*
- **Firewall**: sistema di sicurezza che controlla le connessioni in ingresso e in uscita e applica delle **regole** di blocco e filtraggio.
- Può essere *hardware* o *software*
- Può operare a livello di rete (*network firewall*) o di singola macchina (*host-based firewall*)

Tipi di firewall

- Network layer (*packet filter*)
  - Operano a livello di TCP/IP, analizzando gli header IP, TCP e UDP
- Application layer
  - Operano a livello applicazione, facendo *deep packet inspection*
    - Più efficaci ma richiedono maggiori risorse computazionali
  - Efficaci anche contro malware, exploit di vulnerabilità note, comportamenti dannosi delle applicazioni, ecc.

Packet filtering

Firewall di livello network

Packet filtering

- *Stateless*: ogni pacchetto viene analizzato singolarmente, solo sulla base di campi statici come indirizzo di sorgente o destinazione.
- *Stateful*: tiene traccia delle connessioni TCP e degli scambi UDP in corso, e discrimina le connessioni legittime da quelle sospette.
  - Più efficace ma complesso e pesante rispetto al filtraggio *stateless*

Funzionamento

- Il firewall contiene una **tabella di regole**
- Ogni regola contiene:
  - Caratteristiche del pacchetto (*criteria*)
  - Azione da intraprendere (*target*)
    - Scarta (DROP) o accetta (ACCEPT)

Indice	IP sorgente	Porta sorgente	IP destinatario	Porta dest.	Azione
1	131.114.0.0/16		131.114.54.4	80	SCARTA
2	0.0.0.0	23	112.143.2.2		ACCETTA

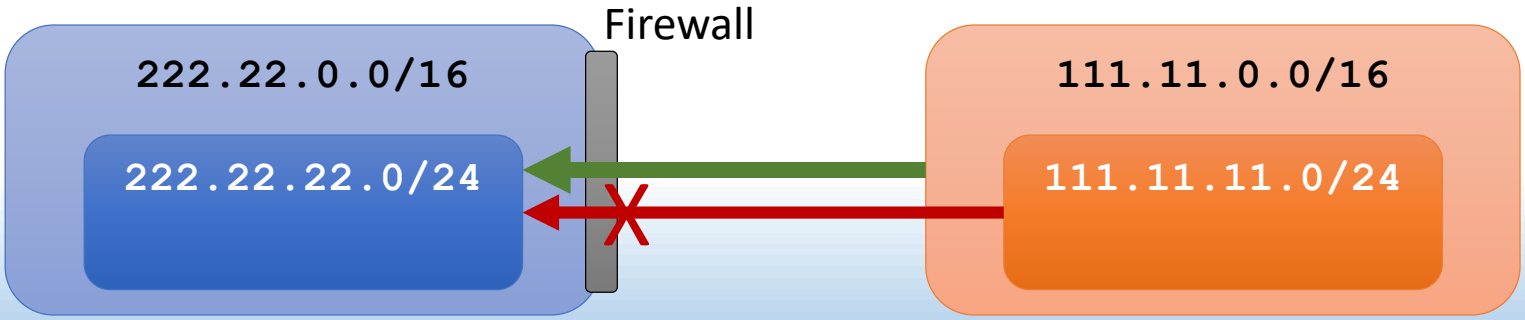
Funzionamento

- **Per ogni pacchetto**, il firewall:
  1. Analizza *l'header*
  2. Scorre la tabella delle regole
  3. *Appena* trova una regola che corrisponde alle caratteristiche del pacchetto analizzato, intraprende l'azione specificata
- Attenzione:

  Le regole sono processate nell'ordine in cui vengono inserite, e **solo la prima** corrispondenza trovata viene applicata!

Ordine delle regole

- Abbiamo una rete locale con indirizzo `222.22.0.0/16` e vogliamo:
  - Impedire l'accesso a Internet dall'interno della rete
  - Consentire l'accesso dalla rete esterna `111.11.0.0/16` alla sottorete locale `222.22.22.0/24`, ma:
  - Impedire alla sottorete esterna `111.11.11.0/24` di accedere alla sottorete locale `222.22.22.0/24`



Ordine delle regole

Indice	IP sorgente	Porta sorgente	IP destinatario	Porta dest.	Azione
1	111.11.0.0/16		222.22.22.0/24		ACCETTA
2	111.11.11.0/24		222.22.0.0/16		BLOCCA
3	0.0.0.0		0.0.0.0		BLOCCA

Indice	IP sorgente	Porta sorgente	IP destinatario	Porta dest.	Azione
1	111.11.11.0/24		222.22.0.0/16		BLOCCA
2	111.11.0.0/16		222.22.22.0/24		ACCETTA
3	0.0.0.0		0.0.0.0		BLOCCA

Regola di default

A seconda della regola di default (ultima riga della tabella), il firewall può essere:

- Inclusivo - Ultima regola: blocca tutto
  - Sicuro ma scomodo, senza definire regole non si può accedere a nulla
- Esclusivo - Ultima regola: consenti tutto
  - Comodo ma insicuro, devo prevedere e inserire manualmente tutte le regole che ritengo utili

netfilter e iptables

Packet filtering su Linux

netfilter e iptables

- **`netfilter`** è il componente del kernel di Linux che offre le funzionalità di:
  - *stateless/stateful packet filtering*
  - *NA[P]T*
  - *packet mangling* (manipolazione generica)
- **`iptables`** è il programma da linea di comando che serve per configurare le tabelle di regole

iptables

`man 8 iptables`

- iptables lavora su diverse tabelle (*tables*), ognuna specifica per una funzionalità.
  - Noi vedremo solo le tabelle **`filter`** e **`nat`**
- Ogni tabella contiene diverse catene (*chains*). Ogni catena contiene una lista di regole da applicare a una categoria di pacchetti.

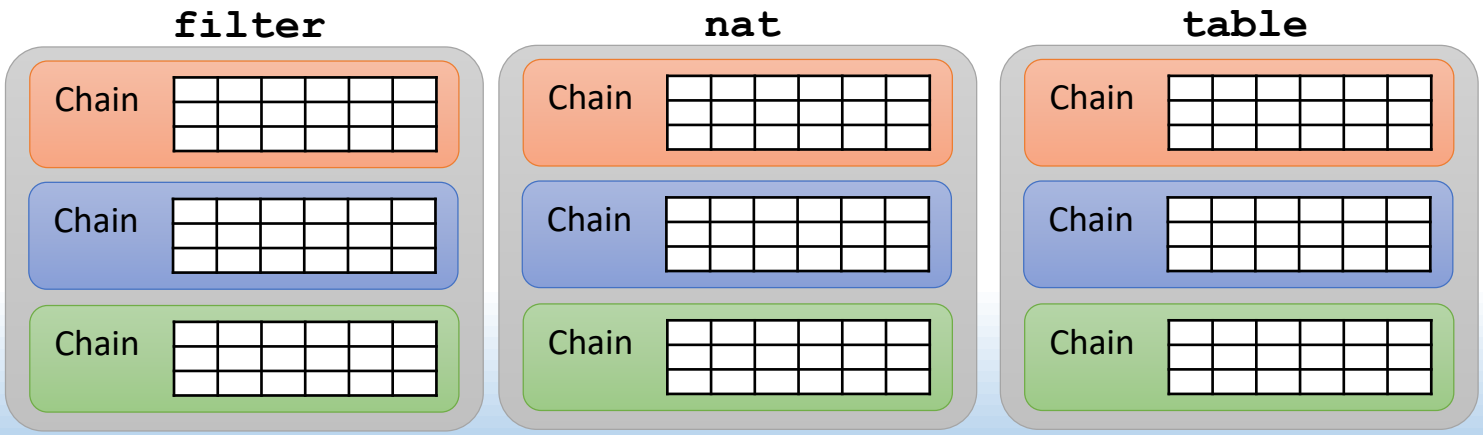
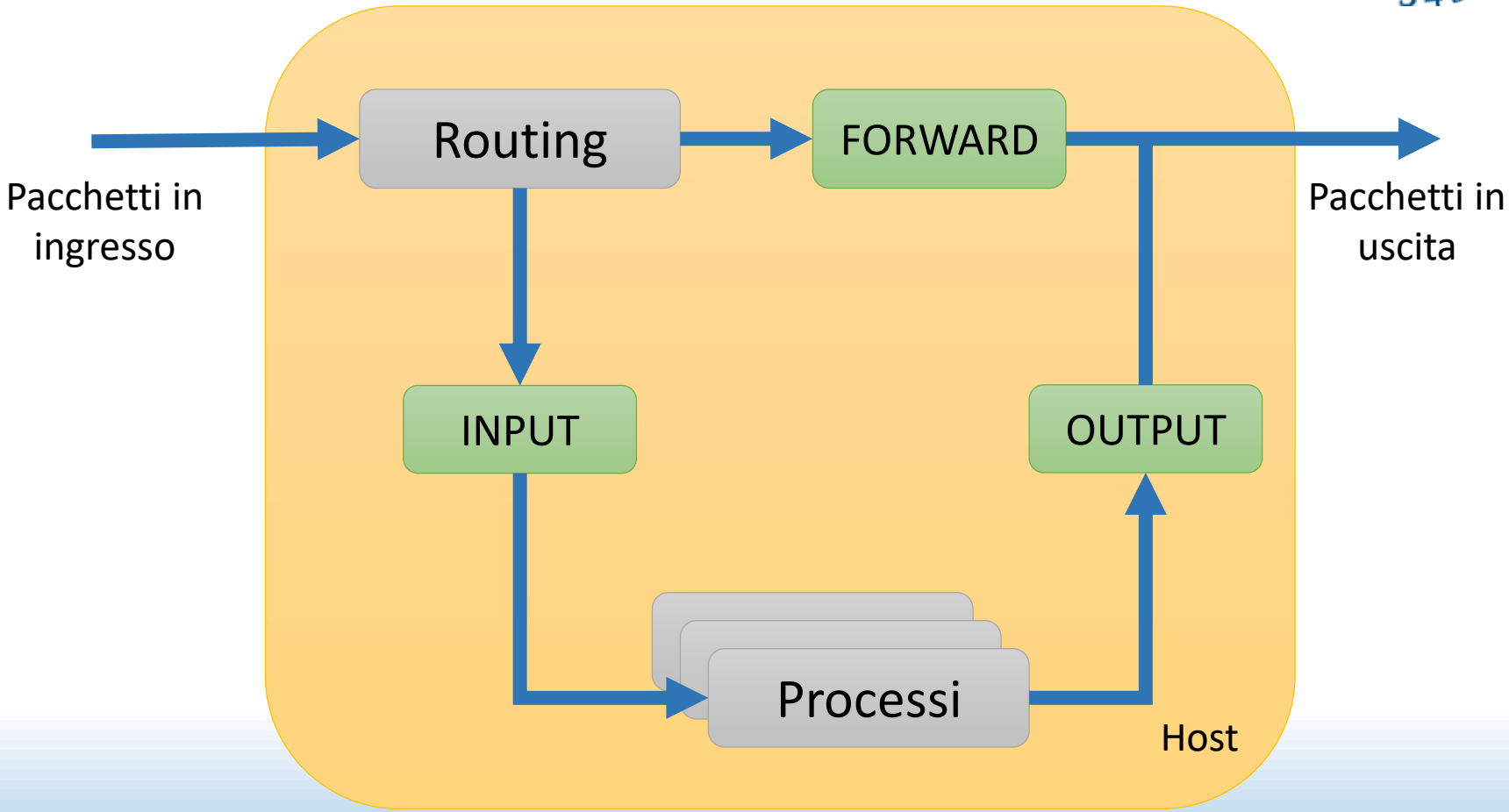


Tabella filter

La tabella **`filter`** ha 3 catene:

- **`INPUT`**, per i pacchetti in ingresso destinati ai processi locali
- **`OUTPUT`**, per i pacchetti in uscita dai processi locali
- **`FORWARD`**, per i pacchetti in transito, cioè da inoltrare ad altri host

Catene di filter



iptables

- Per visualizzare le regole:

```
# iptables [-t table] -L [chain]
```

- Se la tabella non è specificata, viene usata filter
- Se la catena non è specificata, vengono elencate tutte le catene

```
# iptables -L INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
…
# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
…
```

iptables

- Per aggiungere una regola in fondo alla catena:

```
# iptables [-t table] -A chain rule-specification
```

- Per aggiungere una regola in una posizione specifica:

```
# iptables [-t table] -I chain [num] rule-specification
```

Se num non è specificato, si usa 1, in testa alla catena

- Per rimuovere una regola dalla catena:

```
# iptables [-t table] -D chain rule-specification
# iptables [-t table] -D chain num
```

- Per rimuovere tutte le regole dalla/e catena/e:

```
# iptables [-t table] -F [chain]
```

- Per cambiare la regola di default (*policy*) DROP/ACCEPT:

```
# iptables [-t table] -P target
```

Regole

`rule-specification` è un stringa in cui possiamo specificare:

- **`-p <protocollo>`** protocollo (TCP, UDP, ICMP, …)
- **`-s <address>`** indirizzo sorgente
- **`-d <address>`** indirizzo destinazione
- **`--sport <port>`** porta sorgente
- **`--dport <port>`** porta destinazione
- **`-i <interface>`** interfaccia di ingresso
- **`-o <interface>`** interfaccia di uscita
- **`-j <target>`** azione (DROP/ACCEPT)

Regole


```
# iptables -A OUTPUT -p tcp -d 10.0.5.4 --dport 80 -j DROP
# iptables -A INPUT -p udp -s 121.0.0.0/16 -j ACCEPT
# iptables -A INPUT -p icmp -i eth0 -j DROP
…
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT     udp  --  121.0.0.0/16   anywhere
DROP       icmp --  anywhere       anywhere
Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
DROP       tcp  --  anywhere       10.0.5.4      tcp dpt:http

# iptables -D INPUT 1
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
DROP       icmp --  anywhere       anywhere
```

Salvare e caricare le regole

- Le regole non vengono salvate permanentemente, è necessario reimpostarle all'avvio
- Per salvare le regole:

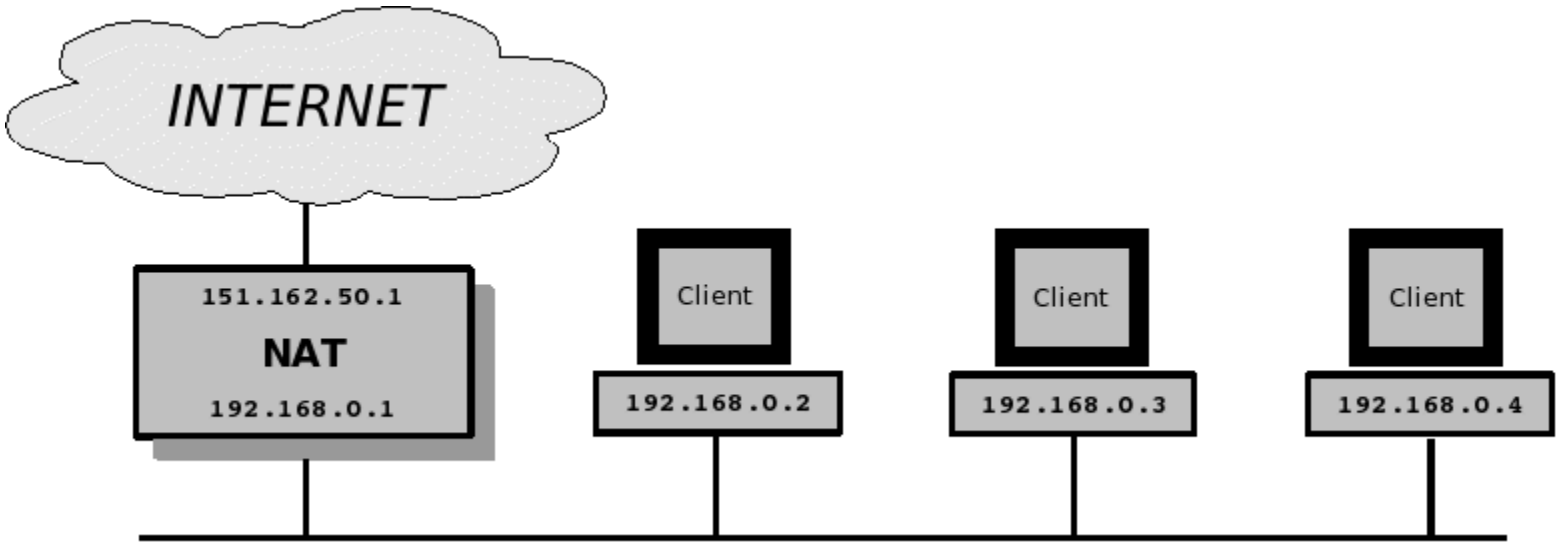
```
# iptables-save > file
```

- Per caricare le regole:

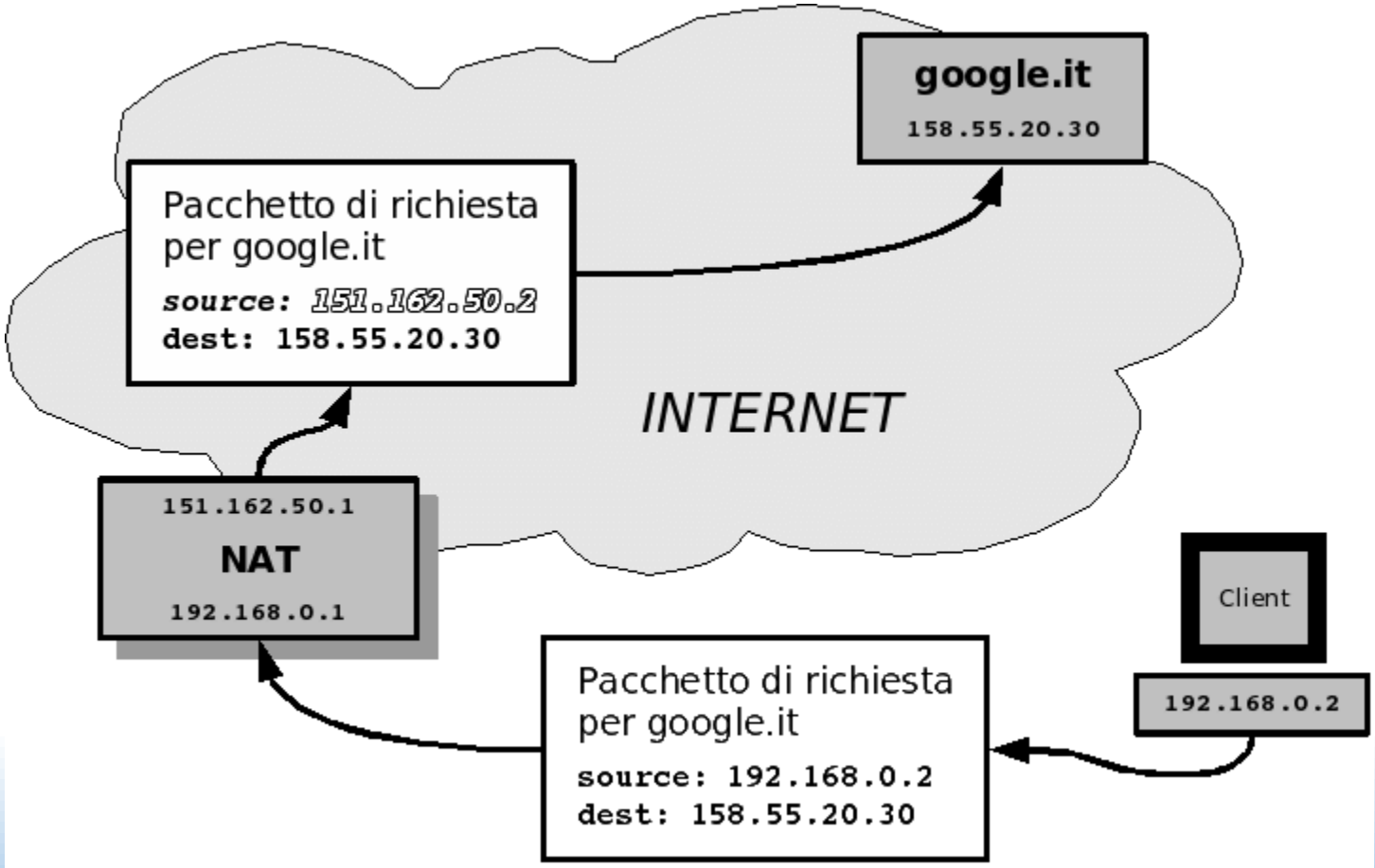
```
# iptables-restore < file
```

NAT e PAT/NAPT

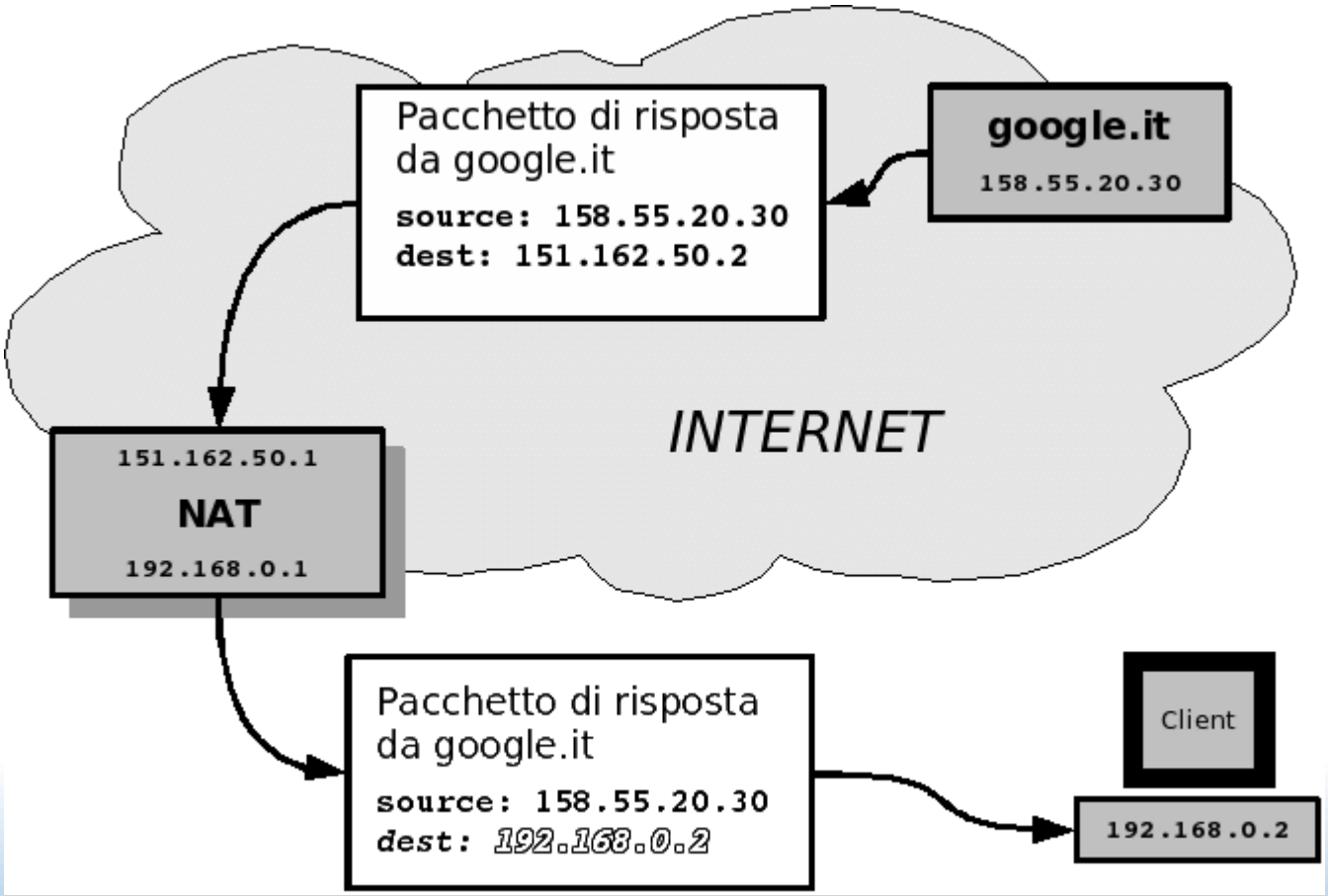
Network Address Translation



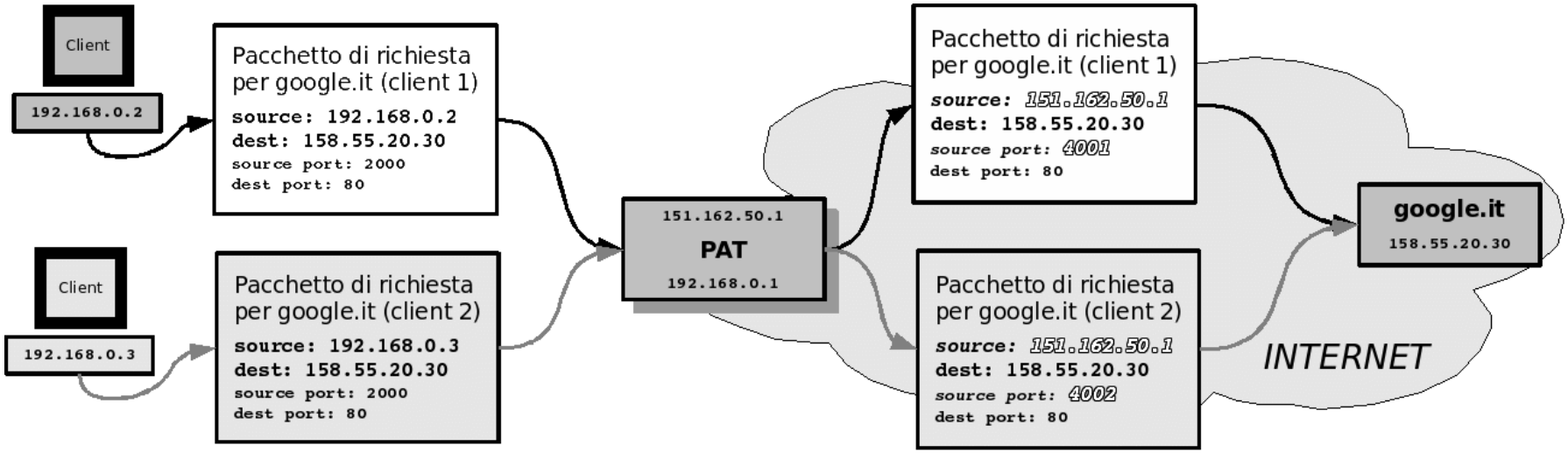
Network Address Translation



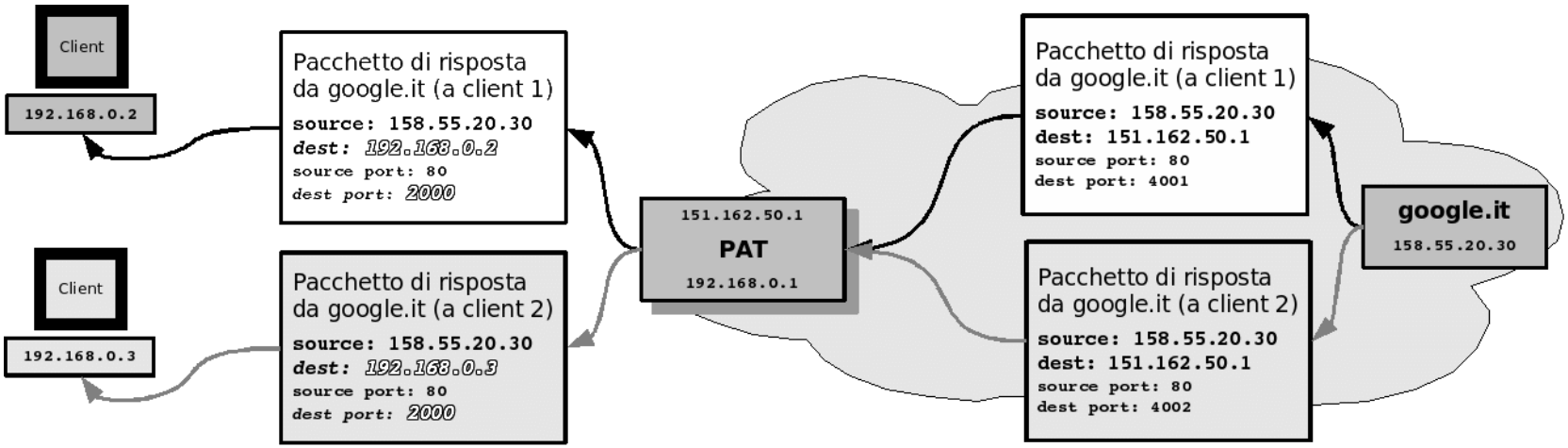
Network Address Translation



Network and Port Translation



Network and Port Translation



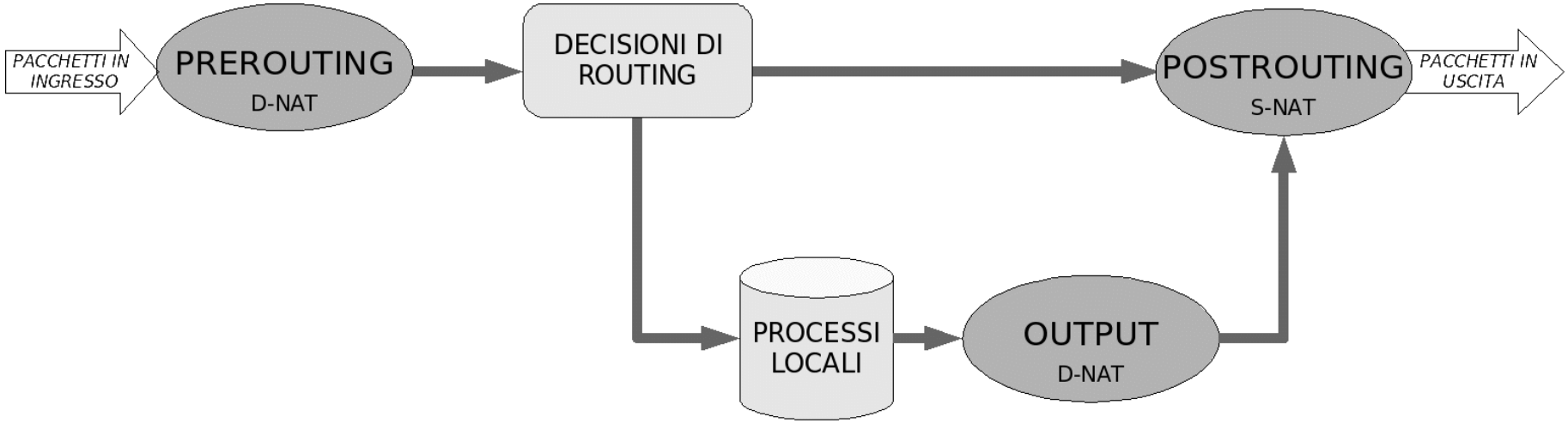
iptables e NA[P]T

- **`iptables`** gestisce il NA[P]T tramite la tabella **`nat`**

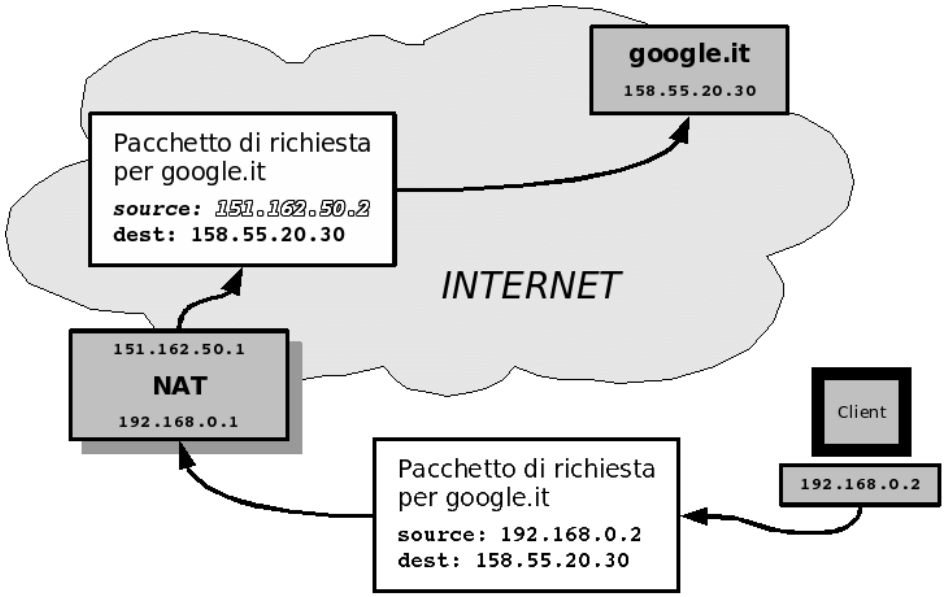
La tabella **`nat`** ha 3 catene:

- **`PREROUTING`**, per fare il *destination NAT*, cioè alterare indirizzo/porta di destinazione dei pacchetti in arrivo
- **`OUTPUT`**, per fare D-NAT dei pacchetti in uscita dai processi locali prima del routing
- **`POSTROUTING`**, per fare il *source NAT*, cioè alterare indirizzo/porta sorgente dei pacchetti in partenza

iptables e NA[P]T

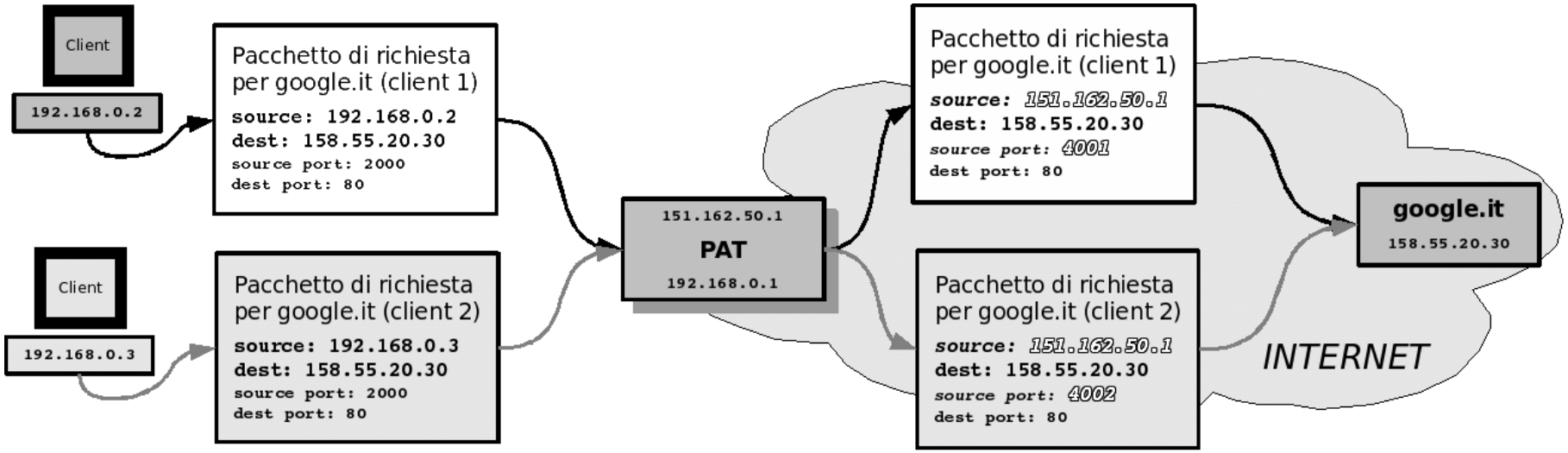


S-NAT



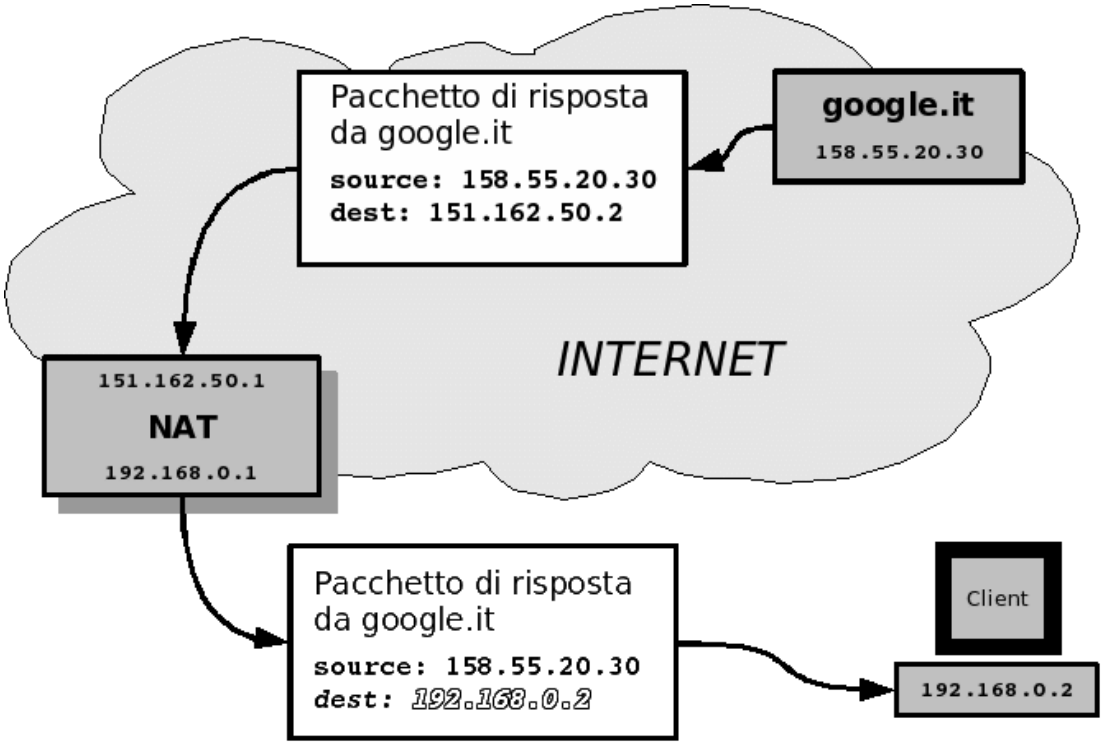

```
# iptables -t nat -A POSTROUTING
-s 192.168.0.2 -j SNAT --to-source  151.162.50.2
```

S-NAT



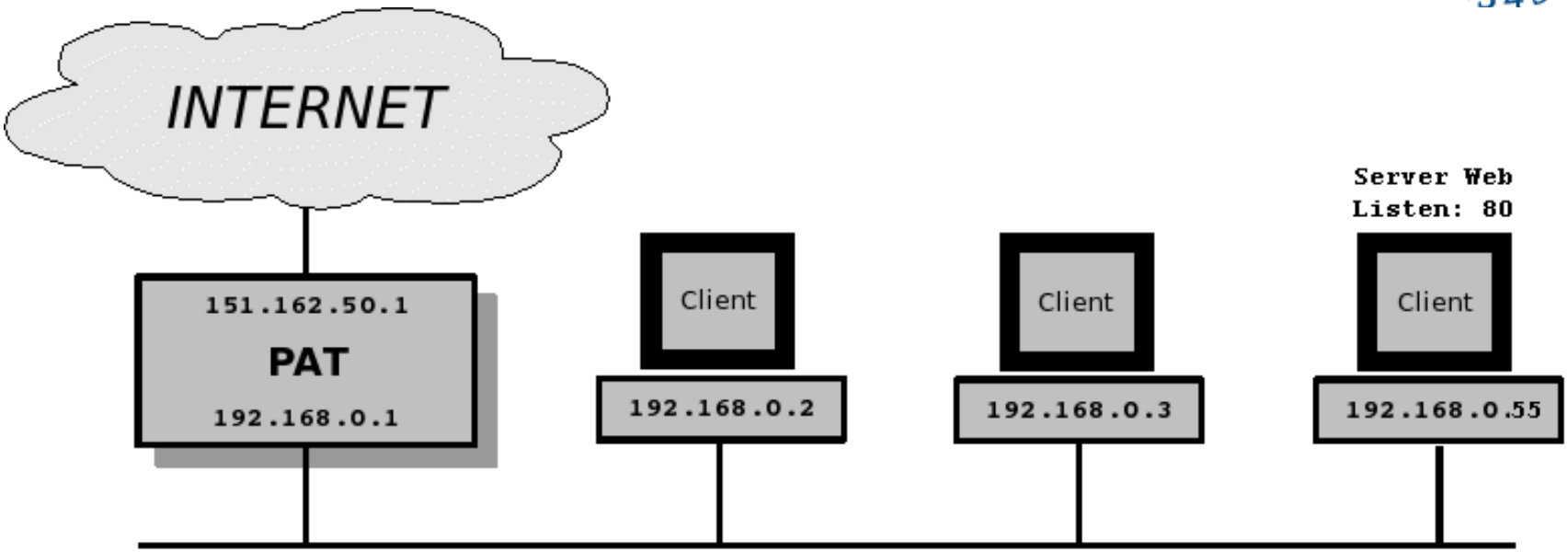

```
# iptables -t nat -A POSTROUTING -s 192.168.0.0/24
-j SNAT --to-source 151.162.50.1:4001-4100
```

D-NAT




```
# iptables -t nat -A PREROUTING
-d 151.162.50.2 -j DNAT --to 192.168.0.2
```

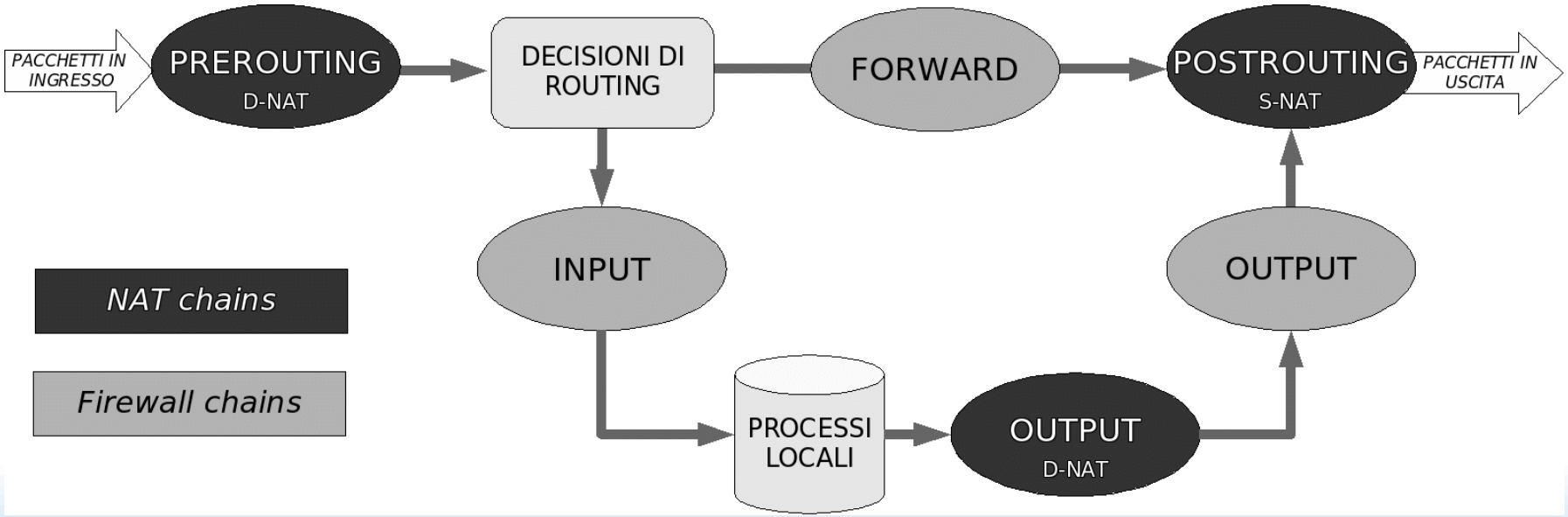
D-NAT




```
# iptables -t nat -A PREROUTING -p tcp
--dport 80 -j DNAT --to 192.168.0.55:80
```

filter e nat

- Le catene di **`filter`** e **`nat`** sono disposte in modo che quelle di **`filter`** vedano indirizzi e porta "reali"



Stateful filtering

Stato della connessione TCP

- Possiamo specificare nella regola un criterio basato sullo stato della connessione TCP di cui un pacchetto fa parte
- Esempio: vogliamo che un host (192.168.0.1) sia accessibile via ssh (porta 22) solo dal computer dell'amministratore (192.168.0.5), ma non possa iniziare sessioni ssh da solo

```
# iptables -P DROP

# iptables -A INPUT -p tcp -i -s 192.168.10.5 -d
192.168.10.1 --dport 22 -m state --state
NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -p tcp -o eth0 -s 192.168.10.1 -d
192.168.10.5 --sport 22 -m state --state ESTABLISHED -j
ACCEPT
```

Stato della connessione TCP

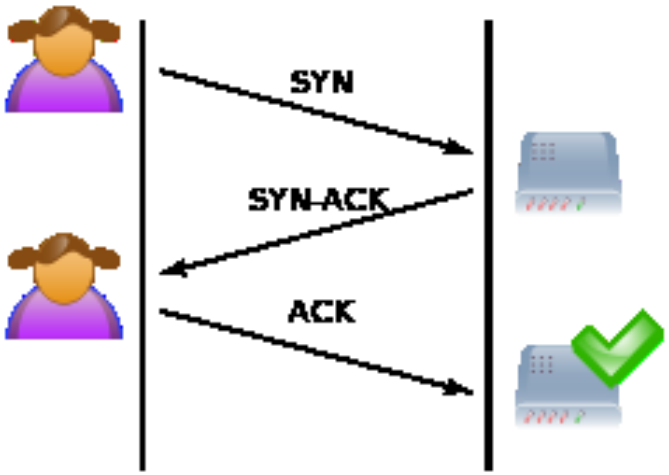
- Esempio: vogliamo che il firewall blocchi le connessioni dall'esterno ma permetta le connessioni che partono dalla rete locale

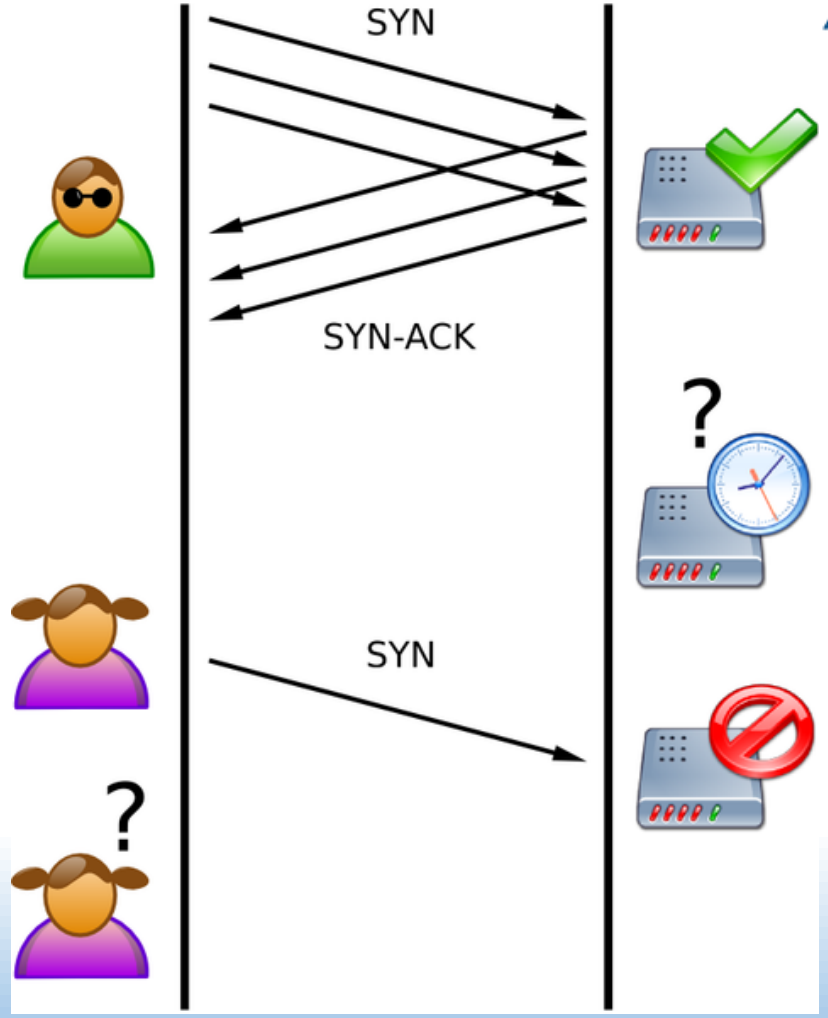
```
# iptables -P DROP
# iptables -A FORWARD -s 192.168.10.0/24 -i eth0 -m
state --state NEW,RELATED,ESTABLISHED -j ACCEPT
# iptables -A FORWARD -d 192.168.10.0/24 -i eth1 -m
state --state RELATED,ESTABLISHED -j ACCEPT
```

- Oppure

```
# iptables -A FORWARD -s 192.168.10.0/24 -i eth0 -m
state --state NEW,RELATED,ESTABLISHED -j ACCEPT
# iptables -A FORWARD -d 192.168.10.0/24 -i eth1 -m
state --state NEW -j DROP
```

Protezione dal SYN flooding





Protezione dal SYN flooding

- Si può aggiungere una regola per cui si accetta non più di una connessione con SYN settato al secondo

```
iptables -I INPUT -p tcp --syn -m limit
--limit 1/s -j ACCEPT
```