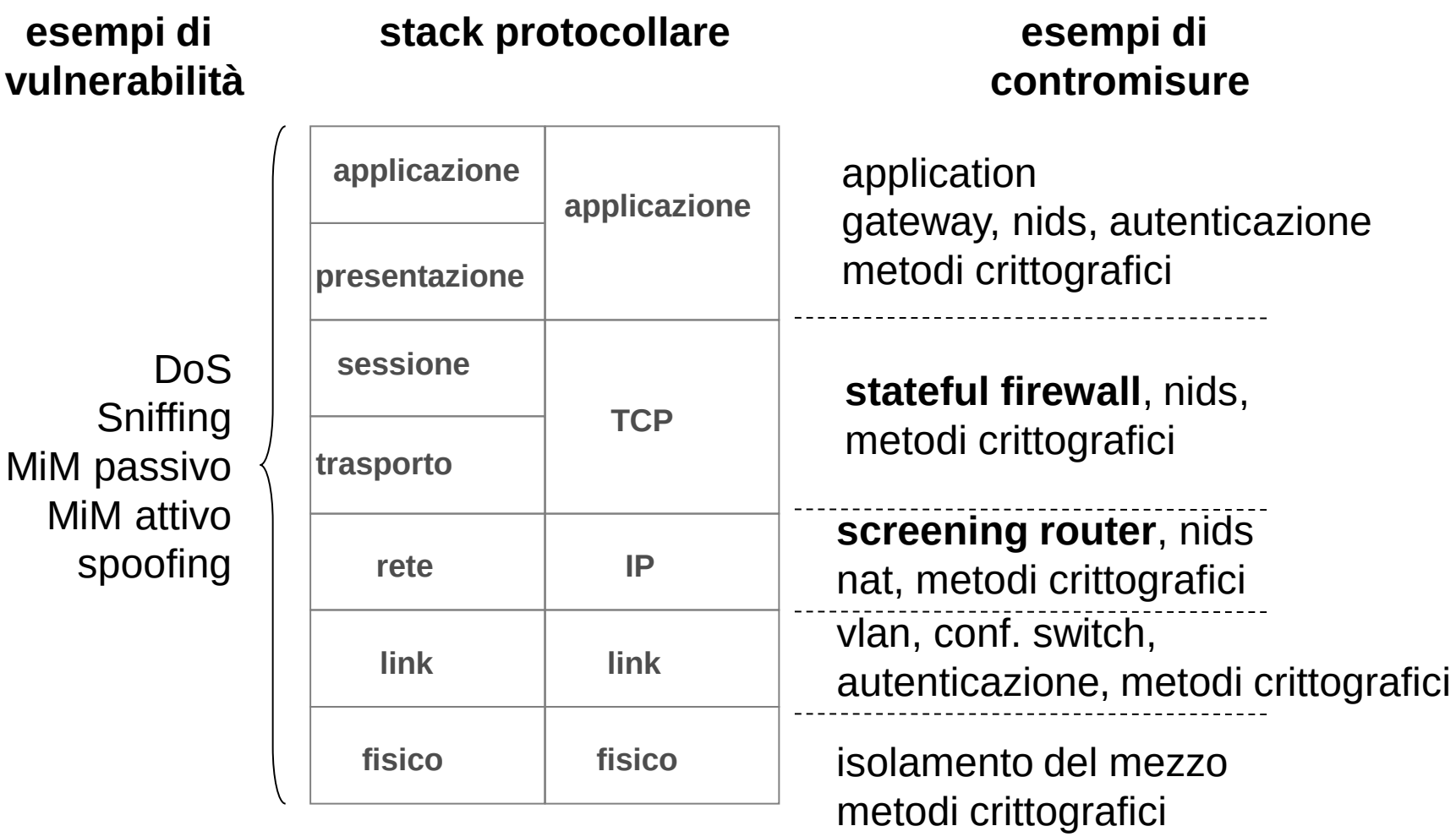
sicurezza delle reti

confinamento nelle reti

- confinamento per classi di utenti e per gestione
  - spesso i due criteri portano alle stesse decisioni o a decisioni complementari
- esempi di confinamento per gestione
  - organizzazioni diverse, dipartimenti diversi, ecc. hanno responsabili diversi
- esempi di classi di utenti
  - amministrazione
    - integrità e disponibilità: critiche per il business
    - confidenzialità: ciritica per legge
    - insieme di applicazioni ben definito
    - sistemi sotto controllo diretto
  - docenti
    - integrità e disponibilità: critiche per il business
    - confidenzialità: ciritica per certi aspetti particolari
    - richiesta alta flessibilità nelle applicazioni
    - il controllo dei sistemi è delegato ai gruppi di ricerca
  - studenti
    - best effort
    - sistemi non controllabili
  - utenti da internet di ricerca
    - servizi selezionati: web, email, siti di ricerca interni, login alle macchine di ricerca e a utenti da internet (altri)
    - servizi selezionati: web, email, siti di ricerca interni

la sicurezza nelle reti



sicurezza delle reti
i firewall e l'esempio di netfiilter

vulnerabilità e minacce a livello rete, trasporto e applicativo

- rete/trasporto come veicolo per attacchi ai sistemi
  - l‘attacco può essere sferrato da molto lontano
  - elemento umano: email, web, ecc. l’utente permette l’entrata di virus, trojan, ecc.
- vulnerabilità dei protocolli
  - già visti all’inizio del corso: protocolli in chiaro e non autenticati, DoS, DDoS

firewalls

- sono apparecchiature che confinano (filtrano) selettivamente il traffico di rete
- network fw: layer3+4
  - stateful vs. stateless
- application fw: layer7
  - a.k.a deep packet inspection o applicative content inspection
- hardware vs. software
- personal fw vs. appliance fw vs. enterprise fw
- possono avere molte altre funzionalità
  - nat, virtual private network, autenticazione utenti, ecc. spesso detti UTM

Unified Threat Management (UTM)

- evoluzione del concetto di firewall
- unisce in un unico dispositivo molte delle funzionalità di sicurezza relative alle reti
  - firewall
  - network intrusion detection/prevention
  - sicurezza delle email
    - antivirus, anti-spam
  - VPN termination
  - applicative content inspection
  - load balancing
- semplicità di gestione e riduzione dei costi
- vedremo molte delle funzionalità come se fossero apparati separati

il firewall come reference monitor

un firewall può essere visto come un reference monitor di rete

- soggetto: l'host sorgente che ha inviato il pacchetto/messaggio
- oggetto: l'host destinazione che riceverà il pacchetto/messaggio
- accesso: richiesta all'host destinazione di processare il pacchetto/messaggio inviato dall'host sorgente
  - caratterizzato dal valore di campi del pacchetto stesso
  - la semantica vera (cioè l'effetto sull'host destinazione) dell'accesso è data dal protocollo di rete e dal contesto in cui viene usato: il fw non ha bisogno di conoscerla
- diritti: categorie di traffico ammesso per la coppia di host

- permette di attuare una policy espressa come access matrix
- normalmente dovrebbe realizzare il principio di mediazione completa

stateless firewall

- regole di “packet filtering”
  - basate sulla quadrupla <saddr,sport,daddr,dport>
  - in pratica sono router configurati con delle access control list
    - detti anche “screening routers”
- non mantengono alcuno stato
- tipicamente quando si parla di firewall si fa riferimento a **firewall “stateful”**

stateful firewall

- i fw stateful si ricordano le “connessioni”
  - stato della connessione: NEW, ESTABLISHED, altro
  - connessioni NEW: la connessione è creata quando arriva il primo pacchetto (è NEW per il primo pacchetto)
  - la stessa connessione è ESTABLISHED per i pacchetti successivi al primo
- ciascun pacchetto è assegnato ad una connessione
  - permettono regole in base allo stato della connessione
  - verificano la correttezza del protocollo
    - es. syn ammesso solo per connessioni nuove
- possono modificare il traffico
  - es. per implementare schemi anti-SYNflood

connessioni

connessione: vari significati a seconda del contesto

- connessione per i protocolli, vedi standard:
  - TCP connesso, UDP non connesso
- connessione per il sistema operativo
  - il sistema operativo mantiene uno stato per i protocolli connessi
  - per il sistema operativo esiste una versione connessa di UDP

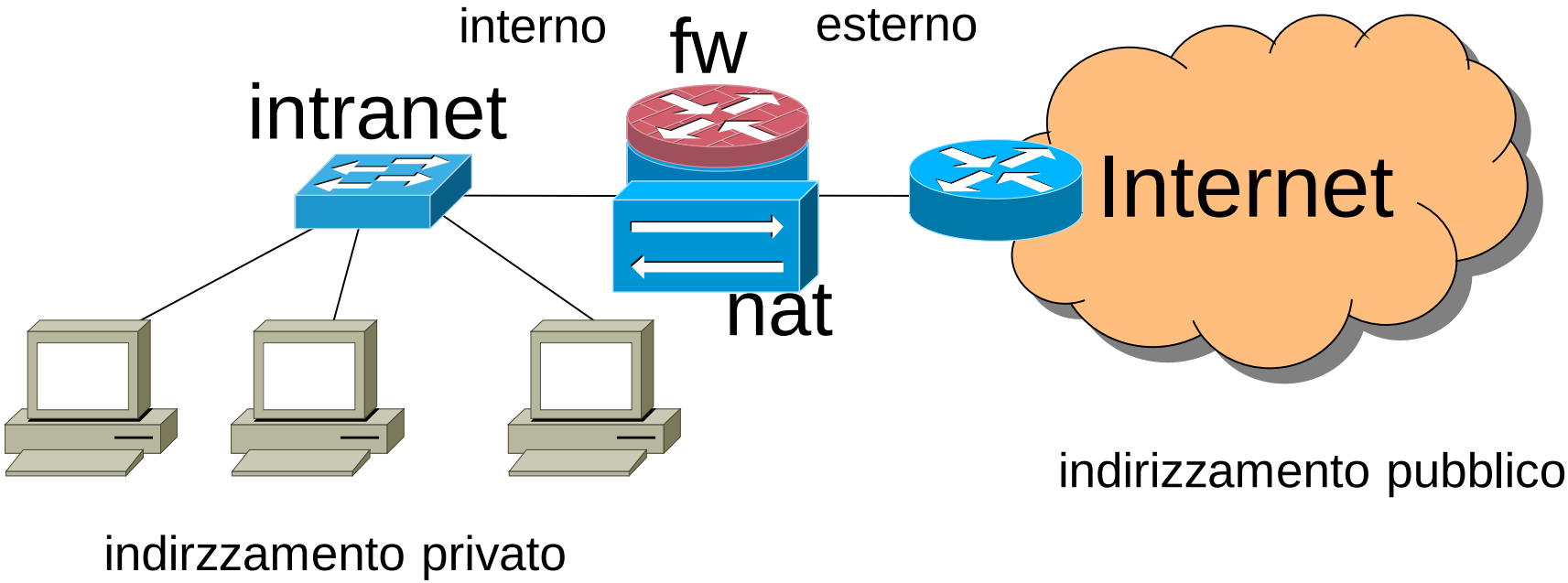
- **connessione per il firewall**
  - tipicamente identificata con la quadrupla <saddr,sport,daddr,dport> (o dalla coppia <saddr,daddr>)
  - **il primo pacchetto** con tale quadrupla **crea la connessione**
  - **gli altri pacchetti** con tale quadrupla sono **relativi alla connessione già creata**
  - es. per un firewall ping e le richieste DNS (su udp) creano “connessioni”

politiche facilmente realizzabili

- politiche per i protocolli tipo domanda-risposta
  - **accettato il primo pacchetto** tutti gli altri della connessione sono accettati
  - se **non viene accettato il primo pacchetto**, tutti i pacchetti della conessione sono scarati

firewall, nat e intranet

- caso d'uso:
  - intranet protetta da Internet
  - in questa configurazione tipicamente il fw fa anche nat ma ciò non è strettamente necessario

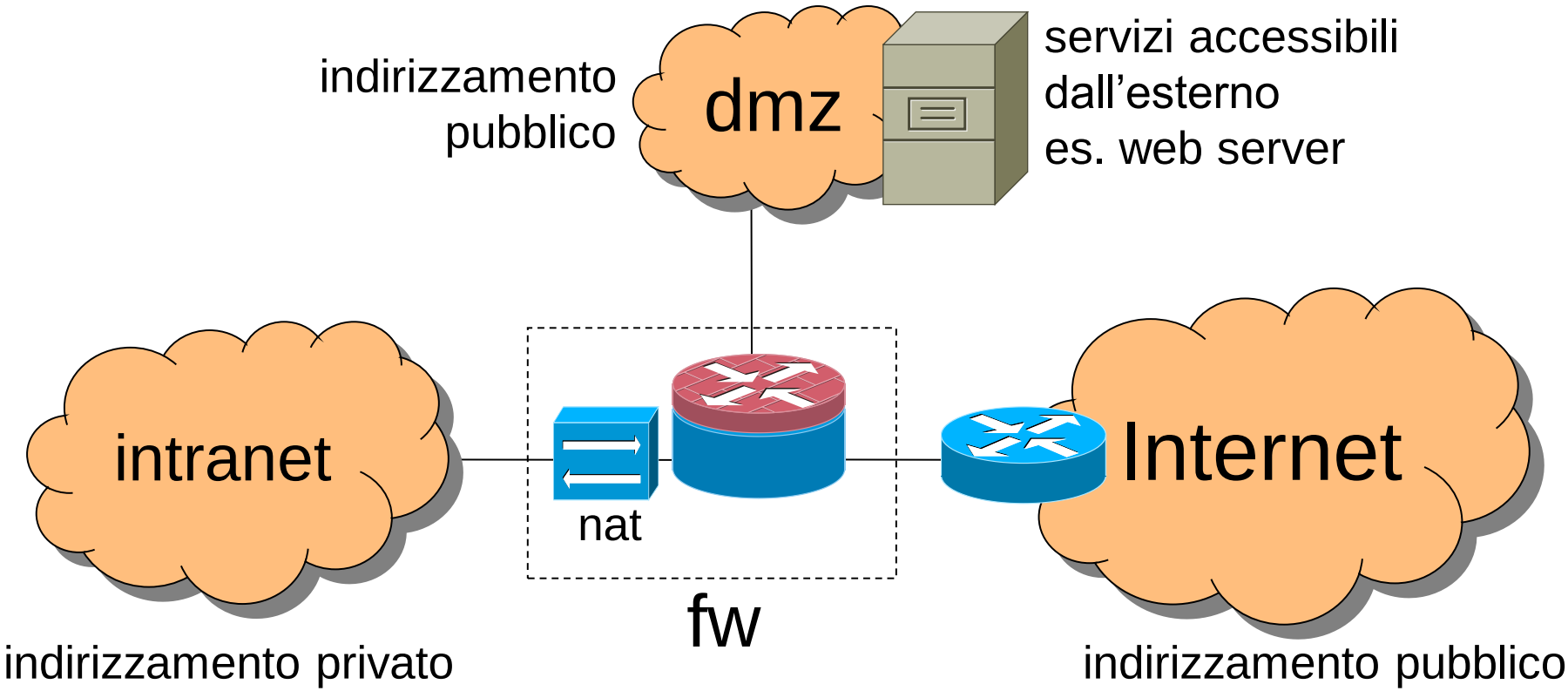


configurazione “canonica”

- due zone (insiemi di interfacce)
  - interno (fidato)
  - esterno (non fidato)
- due regole
  - dall’interno verso l’esterno (outbound)
    tutti i pacchetti sono accettati
    - **questo permette di creare nuove connessioni**
  - dall’esterno verso l’interno (inbound)
    sono accettati solo i pacchetti relativi a connessioni ESTABLISHED
- questa è normalmente la configurazione di default di un firewall
  - può essere considerato un default sicuro

intranet + DMZ

- sistemazione di server protetti in una classe di sicurezza diversa da quella della intranet
- DMZ: demiliterized zone (zona smilitarizzata) o perimeter network (rete perimetrale)
  - rete distinta sia da Internet che dalla intranet accessibile tramite il firewall



intranet + DMZ: esempio di policy

da \ a	host intranet	host dmz	host Internet
host intranet	all non si passa per il fw	richiesta dalla intranet si accede ai servizi della dmz	richiesta dalla intranet si accede a Internet
host dmz	risposta dalla dmz si risponde alle richieste della intranet	all non si passa per il fw	risposta dalla dmz si risponde alle richieste di Internet
host Internet	risposta da Internet si risponde alla intranet	richiesta da Internet si accede ai servizi della dmz	all non si passa per il fw

richieste e risposte per protocolli basati su tcp

- “richiesta”
  - il syn per una nuova connessione (NEW)
  - tutto i pacchetti relativi ad una connessione ESTABLISHED
- “risposta”
  - tutto ciò che è relativo ad una connessione ESTABLISHED (dopo il syn)

- facile da implementare in un firewall stateful

struttura della matrice

- solo richieste e risposte
- richieste accoppiate a risposte
- le coppie hanno posizione simmetrica rispetto alla diagonale

dalla matrice alla configurazione

- una regola per ciascuna «richiesta»
  - tale regola accetta una connessione NEW
    - cioè il primo pacchetto di una connessione
  - memorizza la connessione per i seguenti pacchetti
    - questo avviene implicitamente, non necessita di configurazione esplicita
- una regola complessiva che accetta tutti i pacchetti di connessioni ESTABLISHED
  - realizza tutte le «risposte» e le «richieste» per tutto ciò che concerne i pacchetti successivi al primo

possibili varianti per la dmz

- bloccare l'accesso a Intranet dalla internet
  - infatti Internet è una fonte di input non fidato
    - consigliato l'uso di application level gateway o applicative content inspection
    - consigliato limitare i protocolli per rendere il content inspection più efficace (es. solo http)
- dmz potrebbe dover accedere a…
  - dns, per arricchire i log
    - fonte non fidata
  - servizi di altri fornitori
    - verso cui possiamo avere un certo grado di fiducia che si dovrebbe tradurre in una più o meno stringente configurazione del firewall

problemi

- la configurazione può essere molto complessa
  - facile fare errori
  - risoluzione DNS:
    - non fidata
    - quando farla?
    - inefficiente
  - configurazione deve essere fatta con indirizzi IP
    - o con nomi configurati staticamente

firewall e DDoS

- DDoS che saturano la banda
  - il firewall non aiuta
    - perché il target del DDoS è l'infrastruttura di rete
  - anycast e CDN sono approcci migliori ma costosi
    - esistono specifici servizi in cloud che fanno economia di scala

- DDoS sulle risorse del server nel caso **senza spoofing**
  - rilevamento in base all'attività delle sessioni
    - aging e numero di sessioni aperte
  - contrasto mediante **white/black-list**
    - gli IP non censiti vengono monitorati, o almeno un campione di essi viene monitorato
    - gli IP che tengono molte sessioni aperte a lungo immotivatamente vengono messi in blacklist
    - gli IP che si comportano bene vengono messi in whitelist
      - tanti IP: memorizzati in forma compressa con *bloom filters*

syn-flood e syn-proxy

- **syn-flood**: DDoS che inviano tanti syn con **spoofing della sorgente**
- **syn-proxy**: un firewall che protegge il server da syn malevoli e da syn-flood
- come funziona:
  - quando arriva un syn: il fw risponde syn-ack e non inoltra il syn al server
  - quando arriva l'ack, il fw riconosce la sessione come buona e apre una sessione tcp con il server
  - da ora in poi il fw fa da proxy tra le due sessioni
  - il fw è progettato per scalare sulle connessioni "mezze aperte" (per cui solo il syn è stato ricevuto)
    - ricorda solo IP e numeri di sequenza

syn cookies

- **syn cookies:** contromisura per attacchi syn-flood
- implementata su server o in fw con funzione di syn-proxy
- una delle poche soluzioni in grado di discriminare il traffico lecito da quello non lecito in maniera automatica
- **obiettivo: rispondere al syn senza mantenere stato**
- questo permette di scalare arbitrariamente rispetto al numero delle connessioni mezze aperte

- implementato in Linux

  echo 1 > /proc/sys/net/ipv4/tcp_syncookies

syn cookies

- lo stato è codificato nel sequence number di tcp
- per le **connessioni «buone»** il server **ottiene nella risposta lo stato che avrebbe dovuto mantenere**
  - il syn+ack contiene il sequence number +1
- se l'attaccante risponde con un ack corretto, non sta facendo spoofing: approccio white/black-list
- l'attaccante potrebbe inviare direttamente un ack!
- **contrasto a ack malevoli**
  - la codifica è firmata in modo da poter riconoscere gli ack genuini es. segreto *s*, random number *n*, sequence number nell'ack: *n* | hash(*n*|*s*)
    - le realizzazioni sostituiscono *n* con un derivato di IP, porte, e timestamp.
  - lo schema sarebbe ovviamente facilmente attaccabile senza questa contromisura

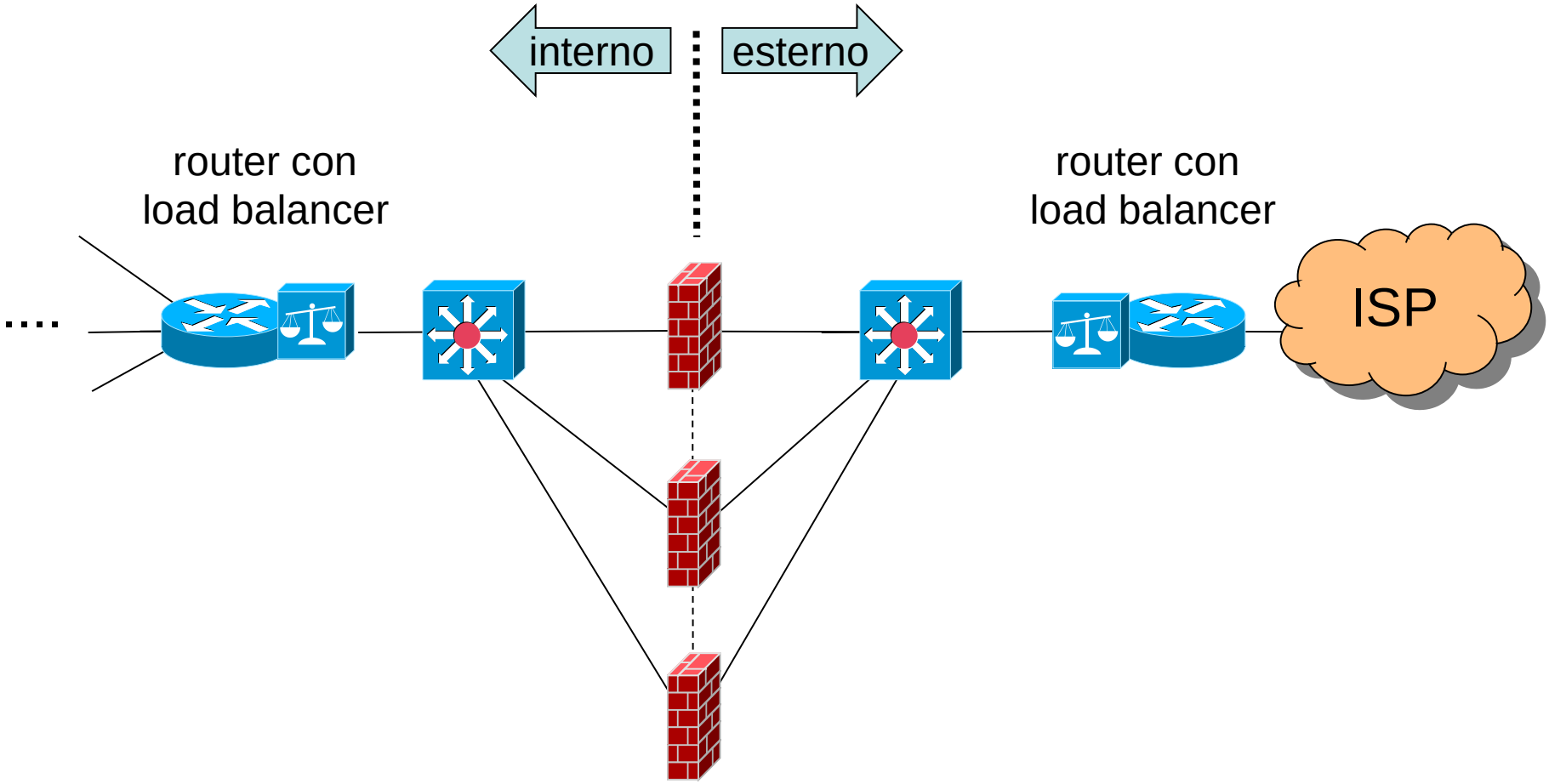
syn cookies: problemi

- il campo sequence number è di soli 32 bit
  - lo spazio è poco: deve contenere una firma + contenuto da firmare
  - la firma è quindi piuttosto debole
  - comunque un attacco (es. brute force) riesce a far passare solo una frazione minima di pacchetti
- limitato supporto a tcp options
  - devono esere codificati nel sequence number o non accettati
- limitato supporto per maximum “segment size”
  - devono essere codificati sequence number, solo un insieme limitato è accettato

firewalls: prestazioni e affidabilità

- i fw spesso sono
  - un **collo di bottiglia** per le prestazioni
  - un **single point of failure**
- spesso utilizzati in configurazione “cluster” con tutti gli elementi attivi
  - **bilanciamento del carico**
  - **ridondanza**
- sono necessari load balancer e switch aggiuntivi

una architettura d'esempio



firewall e condivisione dello stato...

- se i load balancer inviano i pacchetti ad un firewall a caso...
- ... tutti i firewall devono vedere lo stesso stato delle connessioni!

- necessità di...
  - un collegamento tra i firewall
  - protocollo ad-hoc per la notifica delle nuove connessioni agli altri firewall
- si tratta di soluzioni proprietarie

architetture senza condivisione di stato

- strategia alternativa: i pacchetti di una stessa connessione sono sempre trattati da uno stesso firewall
- il load balancer non può fare scelte casuali
- ciascun fw deve vedere il traffico di una connessione in entrambe le direzioni
- in qualche modo i load balancer devono fare scelte coordinate

architetture senza condivisione di stato: coordinamento tra i load balancer

prima possibilità

- ciascun load balancer si ricorda il firewall assegnato alla connessione
  - ... e usa tale fw per i prossimi pacchetti

- richiede memoria lineare con il numero di connessioni
  - poco male la ram costa poco
- richiede di accedere alla ram at wire speed!
  - ...e la cache costa tanto!

architetture senza condivisione di stato
coordinamento tra i load balancer

seconda possibilità

- i load balancer adottano lo stesso algoritmo di scelta, tale che...
  - dipende solo dal pacchetto (es. da src, dest)
  - deterministico
- ...ma usato in maniera «speculare»
  - LB esterno: fw=hash(src_ip, dest_ip)
  - LB interno: fw=hash(dest_ip, src_ip)
  - infatti i pacchetti di ritorno hanno i src e dest invertiti
- no ram, no cache, solo cpu!
  - facile da realizzare in hardware (ASIC)

architetture senza condivisione di stato
coordinamento tra i load balancer

caveat

- il load balancer assegna l'intera connessione ad un fw, ma…
- ...alcuni protocolli coinvolgono più sessioni tcp
  - es. ftp

configurazioni high-availability (HA)

- quanto down-time siamo disposti a sopportare?
  - dipende dal business (service level agreement, penali, ecc.)
- le configurazioni tolleranti ai guasti per mezzo di elementi ridondati si dicono in “fail over” o “high-availability” (HA)
- i single point of failure vanno evitati
  - altrimenti esiste un fault per cui si ha down immediato del sistema
  - vogliamo invece avere del tempo per permettere ad un tecnico di ripristinare lo stato inziale senza che ci sia un down
    - un limitato degrado del servizio potrebbe essere ammesso nel frattempo (dipende dallo SLA)

HA e analisi del rischio

l'high-availability è necessaria quando...

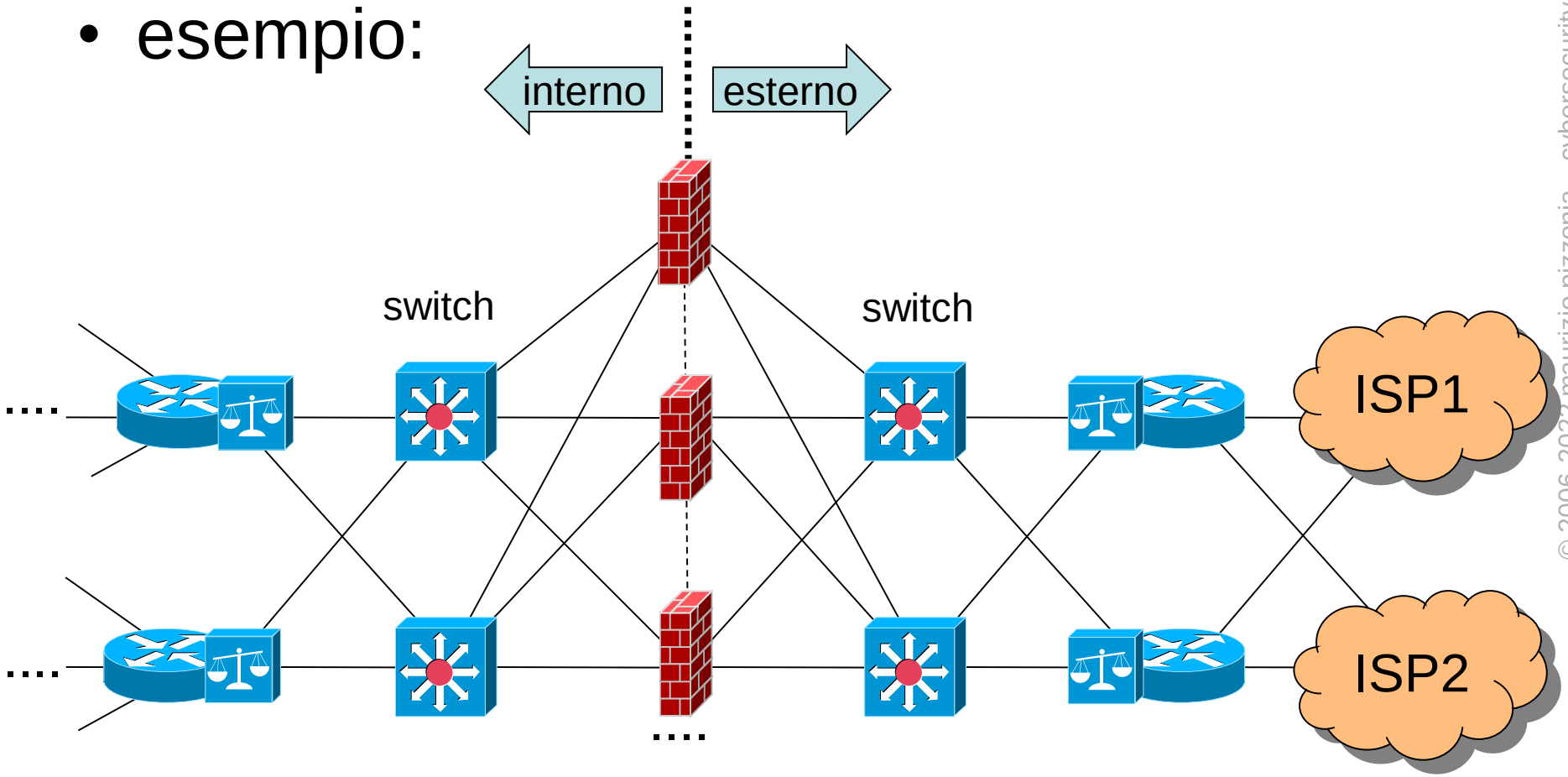
- gli SLA non prevedono il tempo per la sostituzione di uno degli elementi prima che scatti la penale
  - cioè esiste un rischio di pagare penali
- la penale (impatto) è sufficientemente alta da essere un rischio da mitigare
  - è possibile calcolare la spessa attesa per penali se riusciamo a stimare la probabilità di fault
  - i clienti che richiedono la penale potrebbero essere molti (es. hosting)
- il costo della configurazione HA dovrebbe essere inferiore al costo atteso derivante dal rischio di pagare penali

full HA

- in un cluster di fw, load balancer e switch sono dei «single point of failure»
- per una full high-avaliability bisogna ridondare tutto!

full high-availability

- ciascun elemento dell'architettura deve essere ridondato
- esempio:



HA: tempi di reazione

- in caso di guasto di un singolo elemento la rete di deve riconfiguare automaticamente
- OSPF ha tempi di convergenza molto rapidi
  - deve girare sui load balancer e sui firewall
  - convergenza in ~200ms (fonte cisco)
- soluzioni di livello 2 (802.1w rapid spanning tree) sono più lente
  - ~1s (fonte cisco)

HA: costi

- le configurazioni HA sono più complesse e più costose
- capex (capital expenses)
  - apparati ridondati
  - deployment più complesso
  - sistema di alert sui fault
- opex (operating expense)
  - personale (reperibilità, skill specifici)
  - più apparati da manutenere
  - più spazio da affittare nel datacenter

(intermezzo: gw ridondato)

- principio: no single point of failure in lan
- soluzioni tipiche per il default gateway di una lan
  - quando si rompe corri ad accendere il fail-over (!)
  - fai partecipare ciascuna macchina della lan al protocollo di routing (!!!!)
  - Virtual Routing Redoundancy Protocol, rfc 5798

- VRRP
  - un indirizzo ip *A* per un «router virtuale»
  - il router virtuale è composto da, almeno, due router fisici
  - solo uno dei due risponde all'arp request per *A*: il *master*
    - la risposta è messa nella arp cache dalla macchine della LAN
  - il master manda un heartbeat periodico agli altri router
  - quando il master muore, uno degli altri prende il suo posto
    - c'è un meccanismo di elezione come per il root bridge nello spanning tree degli switch
  - il nuovo master manda una **gratuitous arp reply per aggiornare le arp cache di tutte le macchine sulla LAN**
- HSRP: analogo, proprietario cisco

linux netfilter

- sistema di moduli Linux che realizza un fw
  - packet filter
  - stateful
  - nat
  - sistema di access list (chains) organizzate per funzionalità (tables)
  - tables
    - filter
      - chains: INPUT, OUTPUT, FORWARD
    - nat
      - chains: PREROUTING, POSTROUTING, OUTPUT
    - mangle (packet marking), security (mandatory access control), ecc.
  - si possono definire delle “user-defined chain”
  - ciascuna table viene gestita da un modulo del kernel

linux netfilter: chains (acl)

- ciascuna chain ha una sequenza di regole e un target
  - il target specifica cosa fare del pacchetto se questo non ha attivato alcuna regola
- target possibili
  - ACCEPT (pacchetto ok, lascia passare)
  - DROP (pacchetto droppato)
  - REJECT (come drop ma invia un icmp di errore al mittente)
  - <chain name> (salta alla chain specificata, come una chiamata a procedura, utile con “user-defined chain”)
  - RETURN (ritorna alla chain chiamante)
  - ...

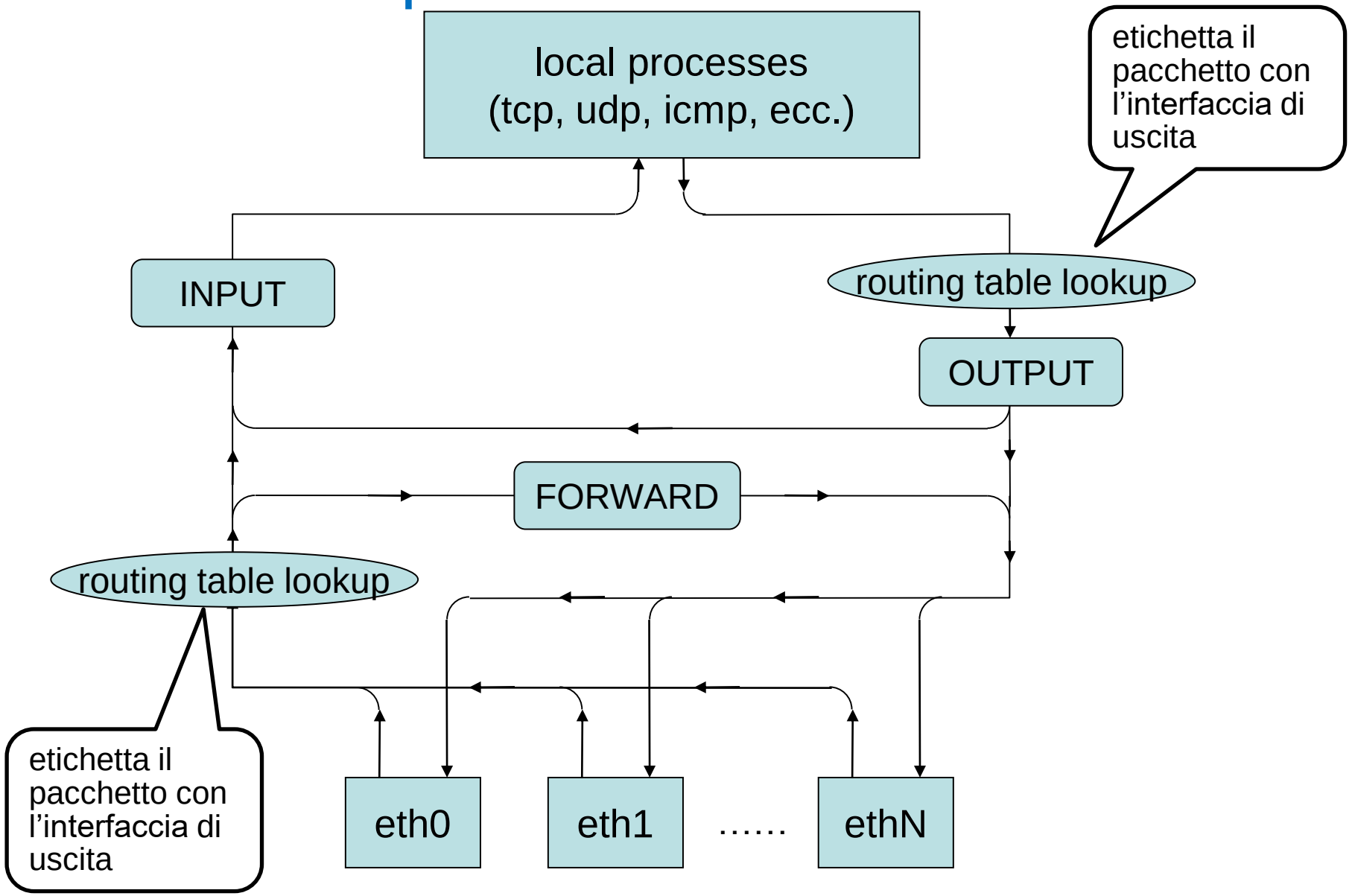
netfilter: comandi

- iptables
  - modifica e mostra la configurazione
    - -L mostra la configurazione
    - --flush cancella la configurazione
    - -A <chain> aggiunge in coda a <chain> un regola
    - ecc.
- iptables-save
  - dump della configurazione attuale
- iptables-restore
  - carica la configurazione da un dump fatto con iptables-save (più efficiente che molte chiamate a iptables)

linux netfilter: firewall rules

- una regola ha dei parametri e un target
  - i **parametri** specificano per quali pacchetti la regola viene attivata
  - il **target** specifica cosa fare
- parametri
  - protocollo (-p ip/tcp/udp/icmp)
  - source/destination address (-s/-d [!]x.x.x.x/x)
  - source/destination port (--sport/--dport port)
  - input/output interface (-i/-o ethN)
  - tcp flags syn=1 e ack=0 (--syn)
  - e altri

chains e flusso pacchetti per la «filter table»



linux netfilter: connection tracking

- netfilter tiene traccia per ciascuna “connessione”:
  - il protocollo
  - per tcp e udp: coppia (non ordinata) di coppie {<addr, port>, <addr, port>}
  - per icmp echo: coppia di indirizzi {addr, addr}
- a ciascun pacchetto è associata una connessione (eventualmente ne viene creata una nuova)
  - questo viene fatto appena un pacchetto entra in netfilter da una interfaccia o dai processi locali
  - le connessioni muoiono per timeout o per protocollo (fin/ack, rst)
- stato del pacchetto
  - NEW: primo pacchetto della connessione
  - ESTABLISHED: pacchetto di una connessione già esistente
  - RELATED: nuova connessione associata ad una precedente (ftp, icmp errors)
  - INVALID: impossibile associare una connessione (es. icmp error non associati ad alcuna connessione)

linux netfilter: extended matches

- l'opzione -m <nome> permette di attivare molti altri tipi di regole
  - **state**: filtro in base allo stato (NEW, ESTABLISHED, RELATED, INVALID)
    - **conntrak** è una versione estesa
  - **mac**: filtro in base agli indirizzi mac di livello 2
  - **limit**: filtro in base alla frequenza di arrivo
    - anti DoS, anti scanning ecc.
  - **iprange**: es. 192.168.1.13-192.168.2.19
  - e molti altri

esempio: nessun filtro

- **iptables --flush**
- **iptables-save**

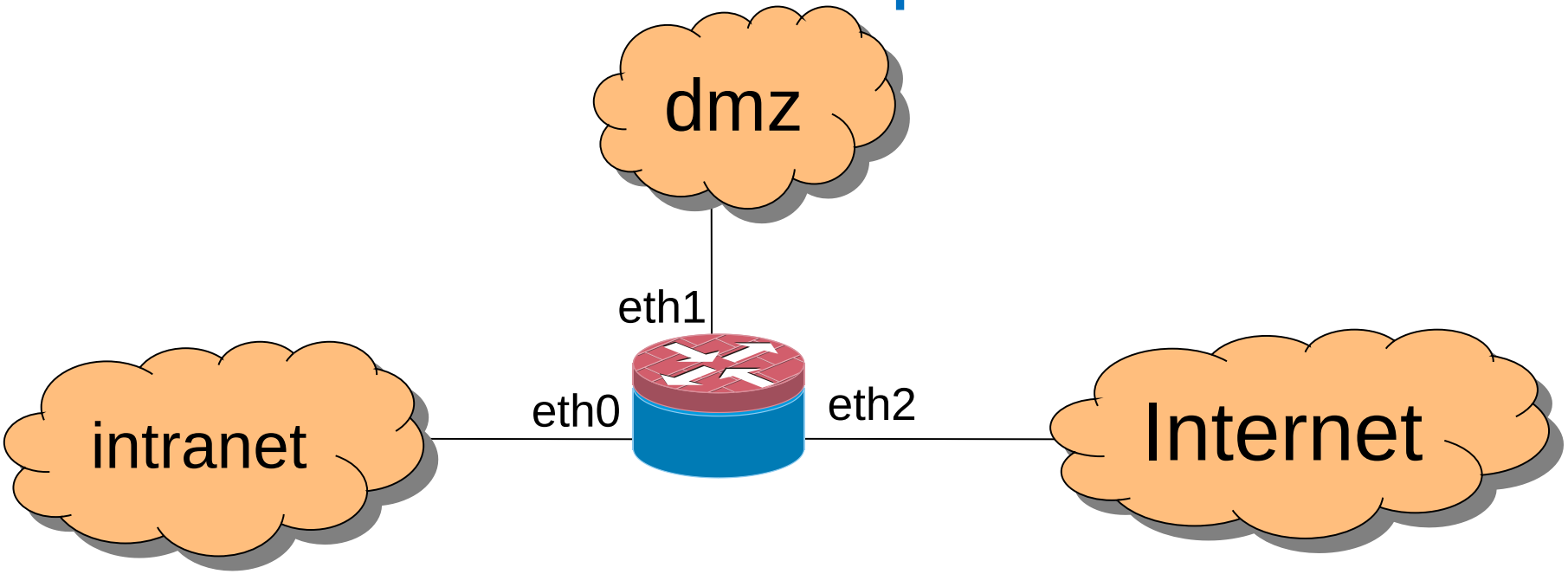
```
# Generated by iptables-save v1.3.3 on Fri Dec  8 17:58:19 2006
*filter
:INPUT ACCEPT [37:4620]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [19:1352]
COMMIT
# Completed on Fri Dec  8 17:58:19 2006
```

esempio: un “personal firewall” con configurazione “canonica”

- `iptables -P INPUT DROP`
- `iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT`

```
# Generated by iptables-save v1.3.3 on Fri Dec  8 18:26:51 2006
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [14735:2397896]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Fri Dec  8 18:26:51 2006
```

implementazione della politica dmz di esempio




```
*filter
:INPUT ACCEPT [64:3448]
:FORWARD DROP [13:858]
:OUTPUT ACCEPT [409:37987]
-A FORWARD -i eth0 -o eth1 -m state --state NEW -j ACCEPT
-A FORWARD -i eth0 -o eth2 -m state --state NEW -j ACCEPT
-A FORWARD -i eth2 -o eth1 -m state --state NEW -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
```

nftables

- iptables è in via di dismissione
- **nftables** sarà il nuovo comando per configurare netfilter
  - sintassi più simile a quella di un router
  - sintassi più semplice
  - un solo comando per tutte le feature
  - supporta matching di *set*
    - permette di scalare sul numero di prefissi e porte su cui fare matching