Introduzione al Networking

Guida introduttiva alla configurazione di rete e al firewalling in ambiente GNU/Linux

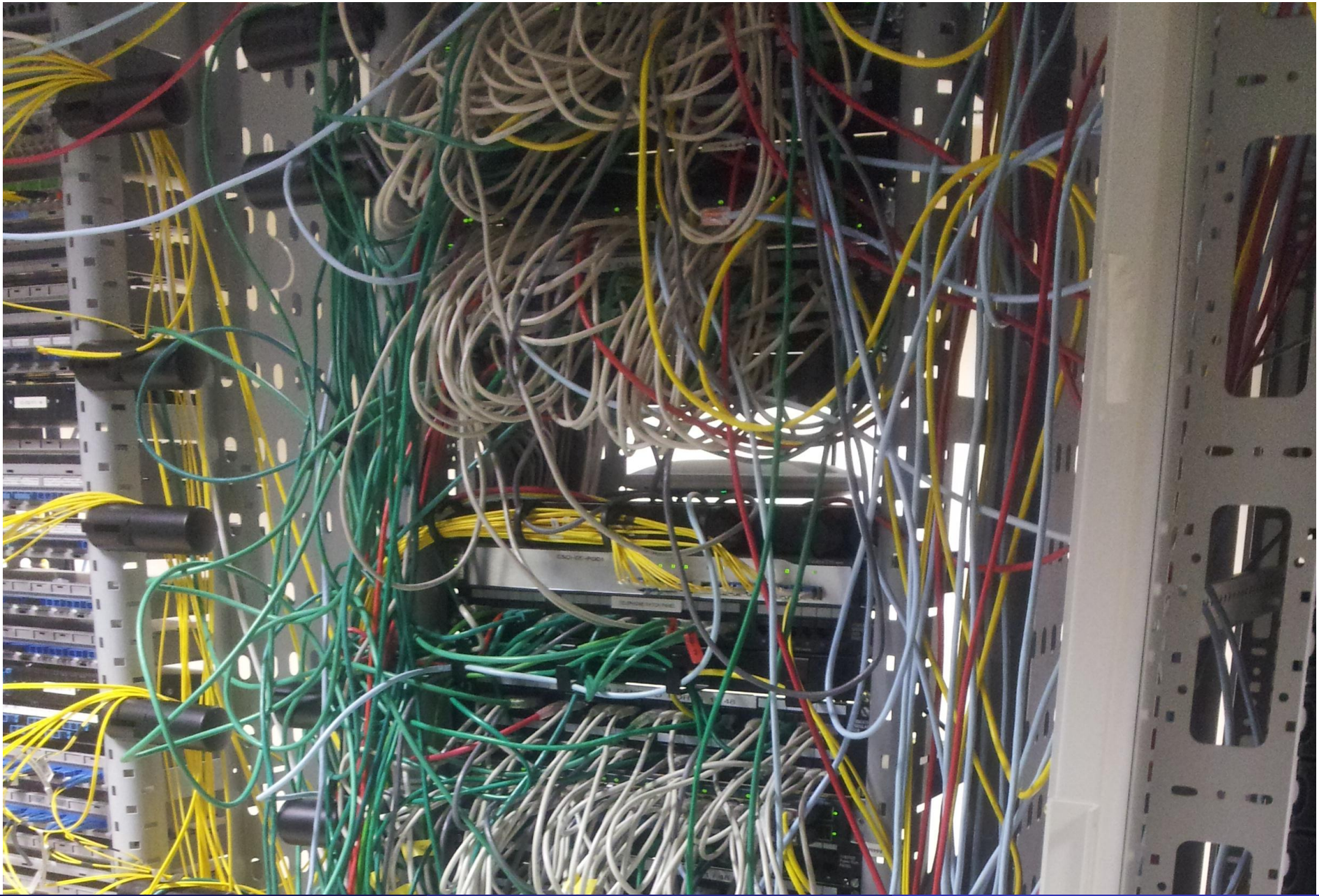
Andrea Grazioso
grazioandre@gmail.com

Corsi GNU/Linux avanzati 2015 - Amministrazione di Sistema

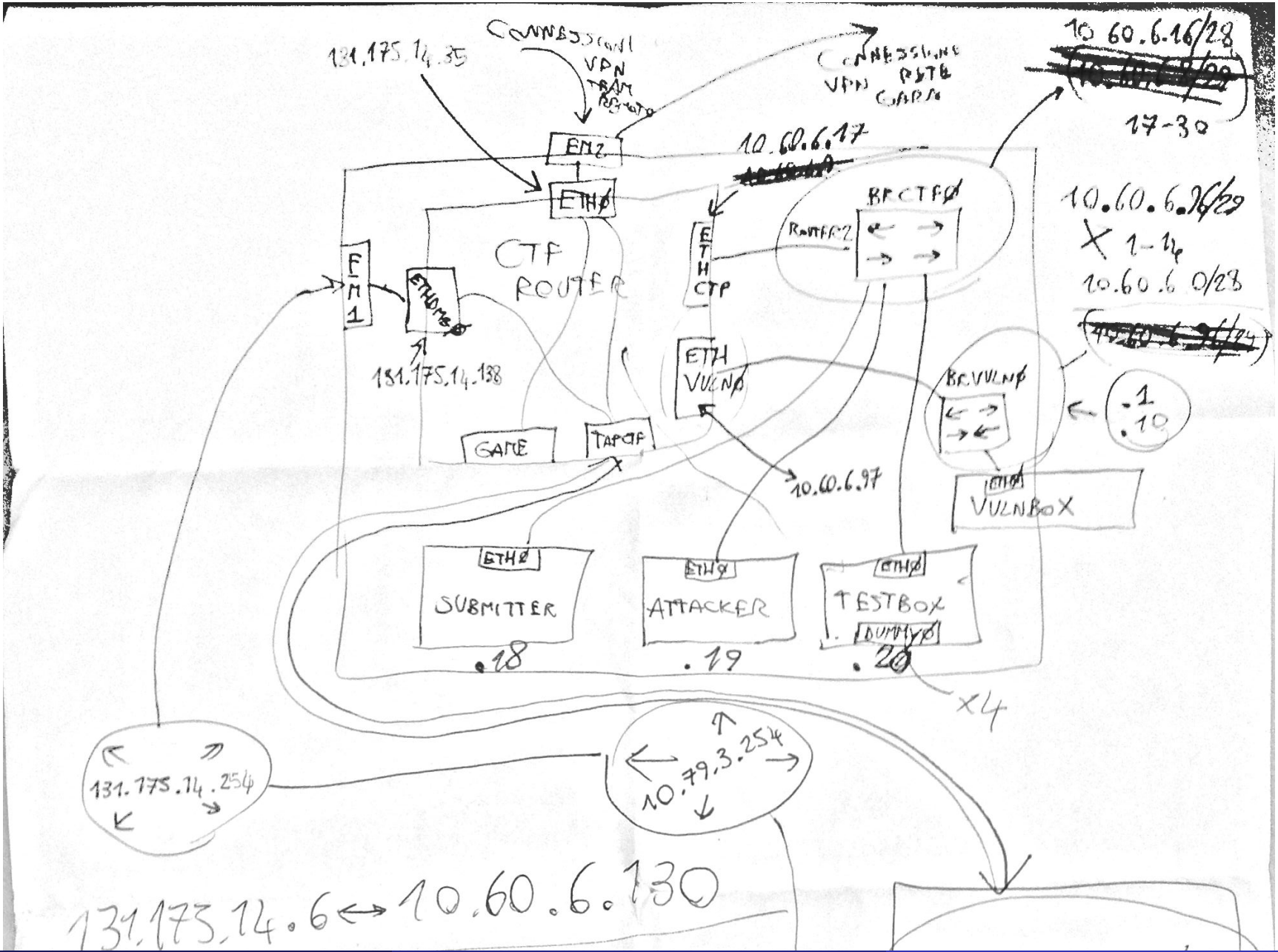
POLITECNICO OPEN
unix LABS
Come hack with us.

Sezione 1

Networking General



Networking General



Talk about what

- Conoscere le basi della rete
- Gestire un host da connettere alla rete
- ...possibilmente senza farci fare le scarpe dall'esterno (aka **Firewalling**)

Talk about what

- Conoscere le basi della rete
- Gestire un host da connettere alla rete
- ...possibilmente senza farci fare le scarpe dall'esterno (aka **Firewalling**)

Talk about what

- Conoscere le basi della rete
- Gestire un host da connettere alla rete
- ...possibilmente senza farci fare le scarpe dall'esterno (aka **Firewalling**)

Disclaimer: Ambienti di Test

Netkit The poor man's system to experiment computer networking

- Utilizza macchine virtuali basate su *User Mode Linux*
- *Nuova versione* (moar features)

Disclaimer: Ambienti di Test

Netkit The poor man's system to experiment computer networking

- Utilizza macchine virtuali basate su *User Mode Linux*
- *Nuova versione* (moar features)

Disclaimer: Ambienti di Test

Netkit The poor man's system to experiment computer networking

- Utilizza macchine virtuali basate su *User Mode Linux*
- *Nuova versione* (moar features)

Sezione 2

ISO/OSI STACK

- Le comunicazioni di rete sono organizzate in **protocolli**
  - Organizzati mediante **stack**
- Dai livelli bassi (livello **fisico**) si sale verso l'alto (livello **utente**)

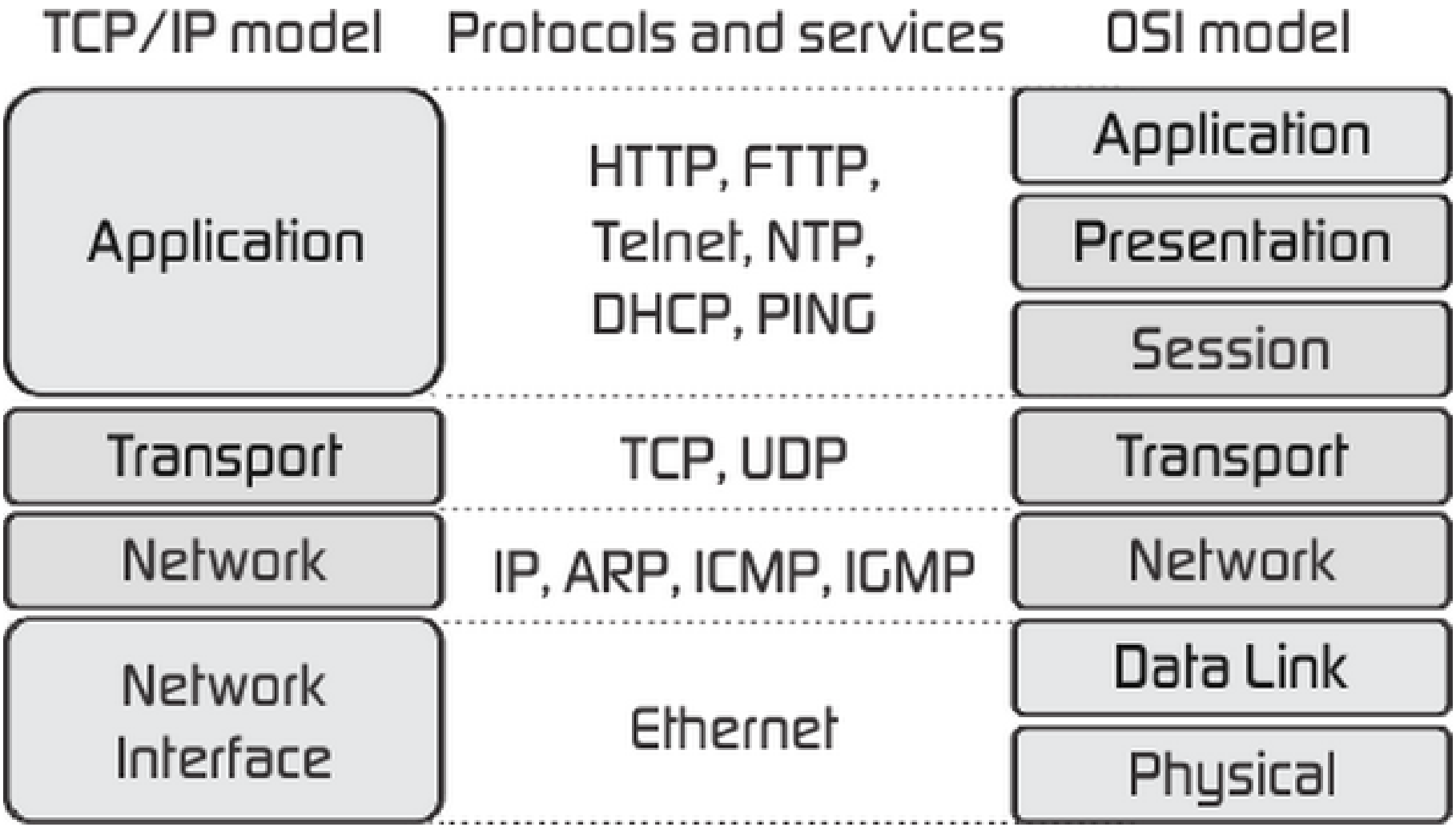
ISO/OSI STACK

- Le comunicazioni di rete sono organizzate in **protocolli**
  - Organizzati mediante **stack**
- Dai livelli bassi (livello **fisico**) si sale verso l'alto (livello **utente**)

ISO/OSI STACK

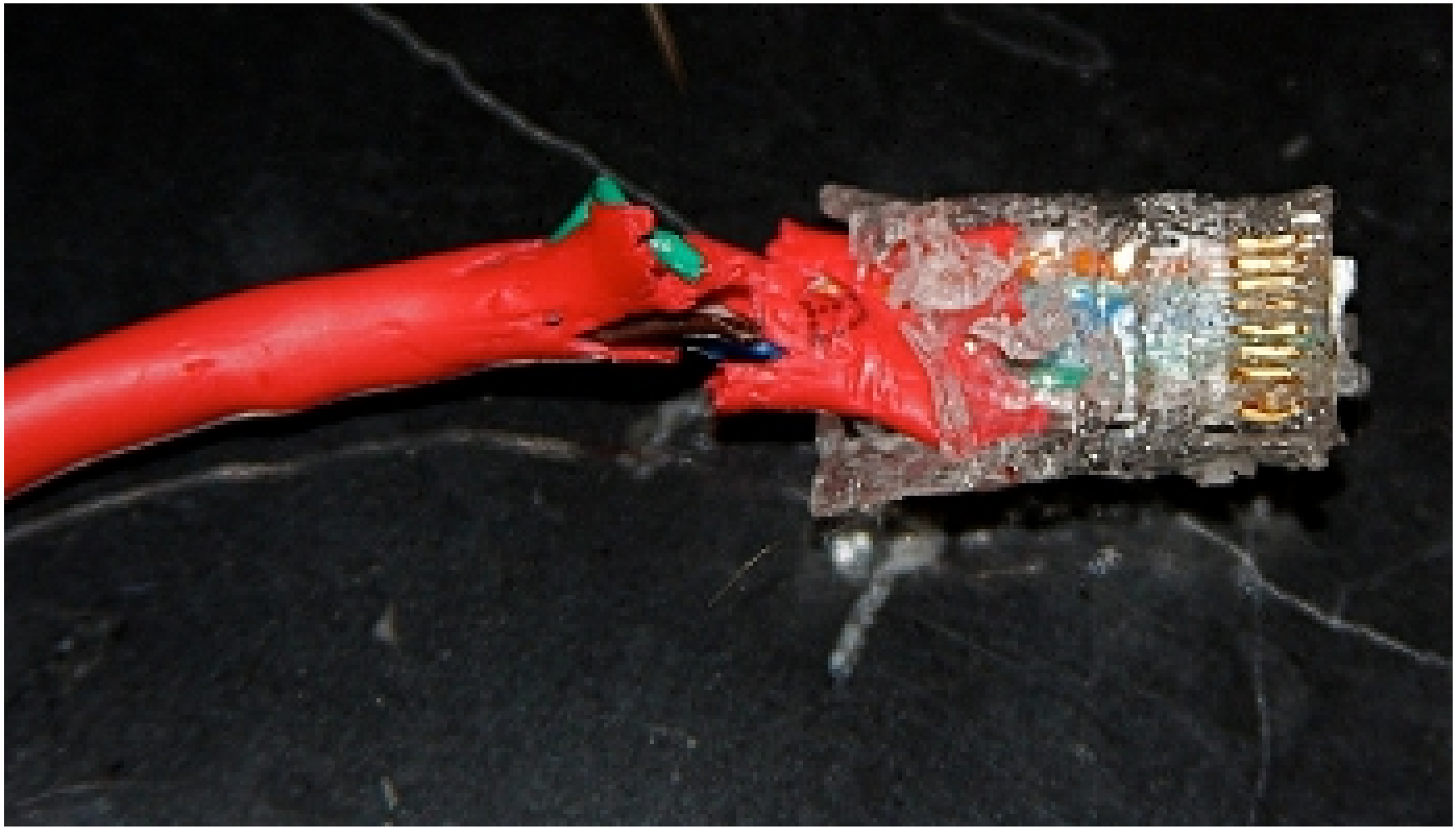
- Le comunicazioni di rete sono organizzate in **protocolli**
  - Organizzati mediante **stack**
- Dai livelli bassi (livello **fisico**) si sale verso l'alto (livello **utente**)

Stack


TCP/IP model
Protocols and services
OSI model
Application
HTTP, FTTP,
Telnet, NTP,
DHCP, PING
Application
Presentation
Session
Transport
TCP, UDP
Transport
Network
IP, ARP, ICMP, IGMP
Network
Network
Interface
Ethernet
Data Link
Physical

Layer 1: Physical

Layer 1: Physical



Layer 2: Data Link - MAC

- Il secondo layer della scala gerarchica è **Ethernet**
  - Permette alle schede di rete di comunicare dei dati
  - Gestisce le eventuali collisioni
- Una scheda di rete è identificata da un indirizzo *univoco* Ethernet (**MAC**[1] **Address**) identificato da 6-byte

[1] Medium Access Control

Layer 2: Data Link - MAC

- Il secondo layer della scala gerarchica è **Ethernet**
  - Permette alle schede di rete di comunicare dei dati
  - Gestisce le eventuali collisioni
- Una scheda di rete è identificata da un indirizzo *univoco* Ethernet (**MAC**[1] **Address**) identificato da 6-byte

[1] Medium Access Control

Layer 2: Data Link - MAC

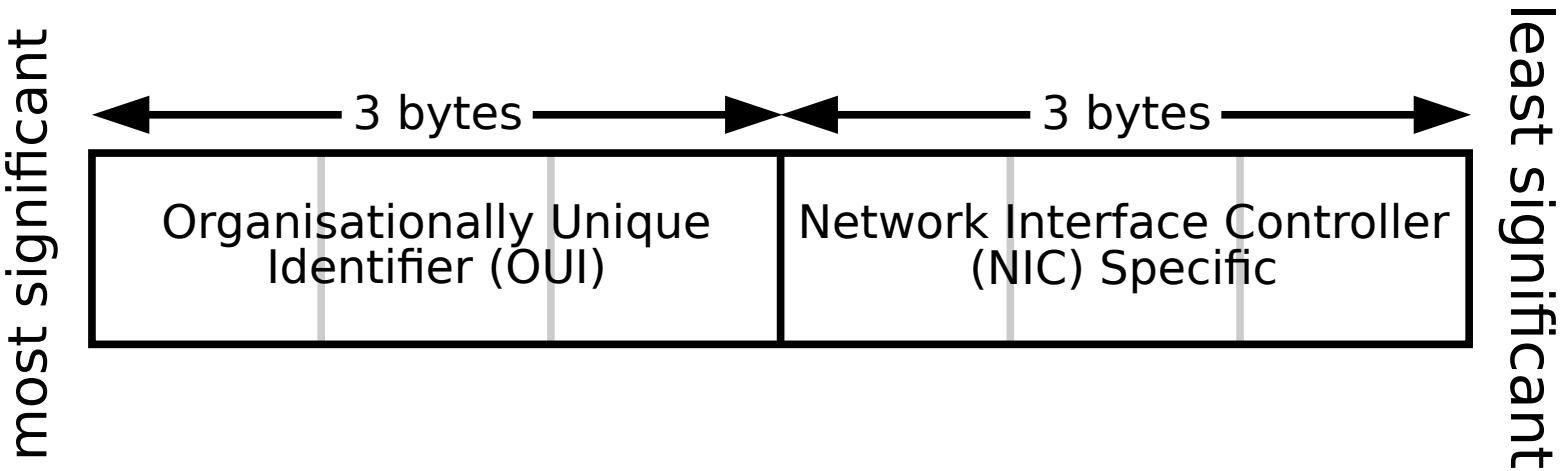
- Il secondo layer della scala gerarchica è **Ethernet**
  - Permette alle schede di rete di comunicare dei dati
  - Gestisce le eventuali collisioni
- Una scheda di rete è identificata da un indirizzo *univoco* Ethernet (**MAC**[1] **Address**) identificato da 6-byte

[1] Medium Access Control

Layer 2: Data Link - MAC

- Il secondo layer della scala gerarchica è **Ethernet**
  - Permette alle schede di rete di comunicare dei dati
  - Gestisce le eventuali collisioni

- Una scheda di rete è identificata da un indirizzo *univoco* Ethernet (**MAC**[1] **Address**) identificato da 6-byte



[1] Medium Access Control

Layer 3: Network - IP

- Il layer di IP permette ai nostri pacchetti di essere spediti correttamente verso la destinazione (purtroppo senza *sani e salvi*)
- I pacchetti IPv4 sono instradati lungo la rete e arrivano all'host...
- ...oppure muoiono provandoci

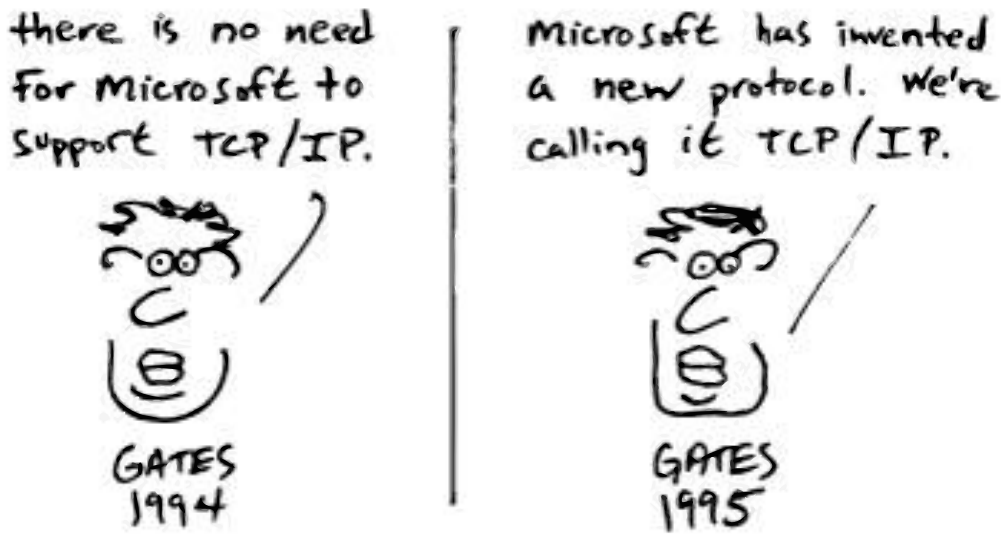
Layer 3: Network - IP

- Il layer di IP permette ai nostri pacchetti di essere spediti correttamente verso la destinazione (purtroppo senza *sani e salvi*)
- I pacchetti IPv4 sono instradati lungo la rete e arrivano all'host...
- ...oppure muoiono provandoci

Layer 3: Network - IP

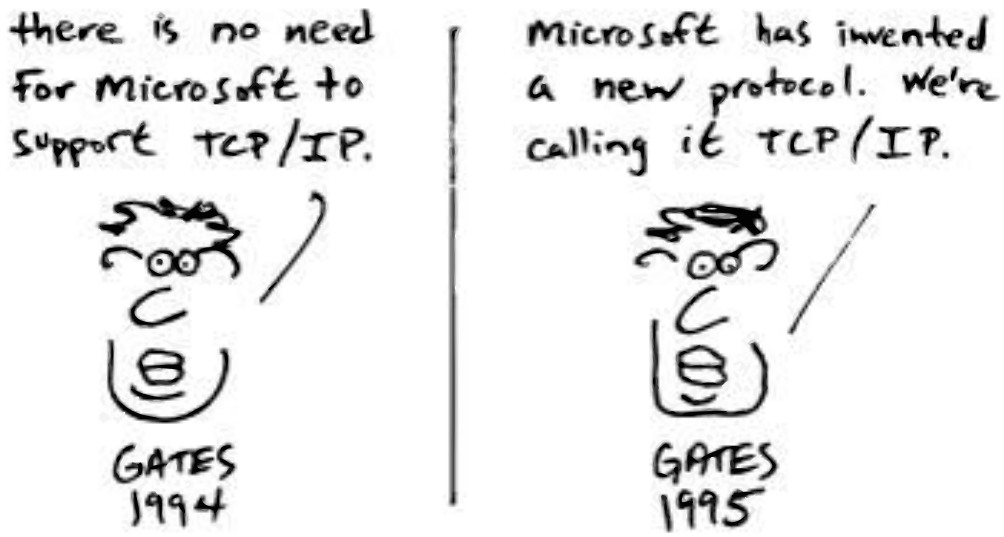
- Il layer di IP permette ai nostri pacchetti di essere spediti correttamente verso la destinazione (purtroppo senza *sani e salvi*)
- I pacchetti IPv4 sono instradati lungo la rete e arrivano all'host...
- ...oppure muoiono provandoci

Layer 4: Trasport - TCP/IP



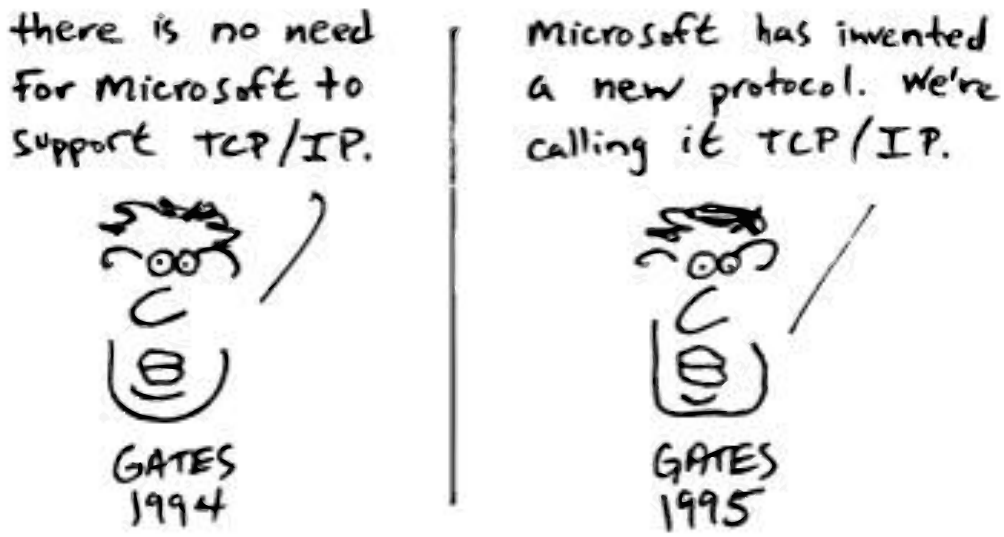
- TCP sopperisce alla «semplicità» del protocollo IP introducendo ulteriori servizi e funzionalità:
  - Permette di avere conferma della ricezione di un pacchetto e in caso di perdita si occupa del reinvio
  - Inoltre TCP permette connessioni multiple tra due stessi host per mezzo di **porte** (0-65535)

Layer 4: Trasport - TCP/IP



- TCP sopperisce alla «semplicità» del protocollo IP introducendo ulteriori servizi e funzionalità:
  - Permette di avere conferma della ricezione di un pacchetto e in caso di perdita si occupa del reinvio
  - Inoltre TCP permette connessioni multiple tra due stessi host per mezzo di **porte** (0-65535)

Layer 4: Trasport - TCP/IP



- TCP sopperisce alla «semplicità» del protocollo IP introducendo ulteriori servizi e funzionalità:
  - Permette di avere conferma della ricezione di un pacchetto e in caso di perdita si occupa del reinvio
  - Inoltre TCP permette connessioni multiple tra due stessi host per mezzo di **porte** (0-65535)

Network Administration/Level Debugging

- Tutto lo stack di rete su linux è basato sulle **interfacce**
  - *Reali*, cioè quelle realmente esistenti nella nostra macchina (*eth0*, *wlan0*)
  - *Virtuali*, generate via software per emulare quelle vere (*tun*, *tap*, virtual devices)
  - *Bridge*, unione (virtuale) di più interfacce

Network Administration/Level Debugging

- Tutto lo stack di rete su linux è basato sulle **interfacce**
  - *Reali*, cioè quelle realmente esistenti nella nostra macchina (*eth0*, *wlan0*)
  - *Virtuali*, generate via software per emulare quelle vere (*tun*, *tap*, virtual devices)
  - *Bridge*, unione (virtuale) di più interfacce

Network Administration/Level Debugging

- Tutto lo stack di rete su linux è basato sulle **interfacce**
  - *Reali*, cioè quelle realmente esistenti nella nostra macchina (*eth0*, *wlan0*)
  - *Virtuali*, generate via software per emulare quelle vere (*tun*, *tap*, virtual devices)
  - *Bridge*, unione (virtuale) di più interfacce

Network Administration/Level Debugging

- Tutto lo stack di rete su linux è basato sulle **interfacce**
  - *Reali*, cioè quelle realmente esistenti nella nostra macchina (*eth0*, *wlan0*)
  - *Virtuali*, generate via software per emulare quelle vere (*tun*, *tap*, virtual devices)
  - *Bridge*, unione (virtuale) di più interfacce

Nel dettaglio: iproute2

- Per il debugging/configurazione delle interfacce, e più in generale dei livelli 2, 3 si usa la suite **iproute2**
- Il nuovo tool **ip** sostituisce i vecchi tool (net-tools[2]) per operare sulle schede di rete
  - Performance migliori per l'utilizzo di **netlink**
  - Comandi più semplici

    ```
    ip [options] object command
    ```

  - Feature di rete più recenti meglio implementate in un nuovo e unico tool

[2] ifconfig, route, netstat, e i tool presenti nel pacchetto *net-tools* sono a tutti gli effetti deprecati

Nel dettaglio: iproute2

- Per il debugging/configurazione delle interfacce, e più in generale dei livelli 2, 3 si usa la suite **iproute2**
- Il nuovo tool **ip** sostituisce i vecchi tool (net-tools[2]) per operare sulle schede di rete
  - Performance migliori per l'utilizzo di **netlink**
  - Comandi più semplici

    ```
    ip [options] object command
    ```

  - Feature di rete più recenti meglio implementate in un nuovo e unico tool

[2] ifconfig, route, netstat, e i tool presenti nel pacchetto *net-tools* sono a tutti gli effetti deprecati

Nel dettaglio: iproute2

- Per il debugging/configurazione delle interfacce, e più in generale dei livelli 2, 3 si usa la suite **iproute2**
- Il nuovo tool **ip** sostituisce i vecchi tool (net-tools[2]) per operare sulle schede di rete
  - Performance migliori per l'utilizzo di **netlink**
  - Comandi più semplici

    ```
    ip [options] object command
    ```

  - Feature di rete più recenti meglio implementate in un nuovo e unico tool

[2] ifconfig, route, netstat, e i tool presenti nel pacchetto *net-tools* sono a tutti gli effetti deprecati

Nel dettaglio: iproute2

- Per il debugging/configurazione delle interfacce, e più in generale dei livelli 2, 3 si usa la suite **iproute2**
- Il nuovo tool **ip** sostituisce i vecchi tool (net-tools[2]) per operare sulle schede di rete
  - Performance migliori per l'utilizzo di **netlink**
  - Comandi più semplici

    ```
    ip [options] object command
    ```

  - Feature di rete più recenti meglio implementate in un nuovo e unico tool

[2] ifconfig, route, netstat, e i tool presenti nel pacchetto *net-tools* sono a tutti gli effetti deprecati

Nel dettaglio: iproute2

- Per il debugging/configurazione delle interfacce, e più in generale dei livelli 2, 3 si usa la suite **iproute2**
- Il nuovo tool **ip** sostituisce i vecchi tool (net-tools[2]) per operare sulle schede di rete
  - Performance migliori per l'utilizzo di **netlink**
  - Comandi più semplici

    ```
    ip [options] object command
    ```

  - Feature di rete più recenti meglio implementate in un nuovo e unico tool

[2] ifconfig, route, netstat, e i tool presenti nel pacchetto *net-tools* sono a tutti gli effetti deprecati

Layer 2: link

- *ip link show* lista le interfacce con i loro MAC address
- *ip link set dev «interface» «up/down»* permette di accendere o spegnere un' interfaccia
- *ip link set dev «interface» address «MAC»* cambia il MAC address che il sistema operativo riporta con uno da noi scelto

Layer 3: IPv4

- *ip addr show* lista tutte le interfacce con i relativi indirizzi IP
- *ip addr add «IPAddress»/ «VLSM» dev «interface»* aggiunge un indirizzo IPv4 all'interfaccia specificata
- *ip addr del «IPAddress»/ «VLSM» dev «interface*

Level4

Il controllo del traffico usualmente viene svolto tramite **Netfilter/Iptables**

iproute2

Layer 2: link

- *ip link show* lista le interfacce con i loro MAC address
- *ip link set dev «interface» «up/down»* permette di accendere o spegnere un' interfaccia
- *ip link set dev «interface» address «MAC»* cambia il MAC address che il sistema operativo riporta con uno da noi scelto

Layer 3: IPv4

- *ip addr show* lista tutte le interfacce con i relativi indirizzi IP
- *ip addr add «IPAddress»/ «VLSM» dev «interface»* aggiunge un indirizzo IPv4 all'interfaccia specificata
- *ip addr del «IPAddress»/ «VLSM» dev «interface*

Level4

Il controllo del traffico usualmente viene svolto tramite **Netfilter/Iptables**

iproute2

Layer 2: link

- *ip link show* lista le interfacce con i loro MAC address
- *ip link set dev «interface» «up/down»* permette di accendere o spegnere un' interfaccia
- *ip link set dev «interface» address «MAC»* cambia il MAC address che il sistema operativo riporta con uno da noi scelto

Layer 3: IPv4

- *ip addr show* lista tutte le interfacce con i relativi indirizzi IP
- *ip addr add «IPAddress»/ «VLSM» dev «interface»* aggiunge un indirizzo IPv4 all'interfaccia specificata
- *ip addr del «IPAddress»/ «VLSM» dev «interface*

Level4

Il controllo del traffico usualmente viene svolto tramite **Netfilter/Iptables**

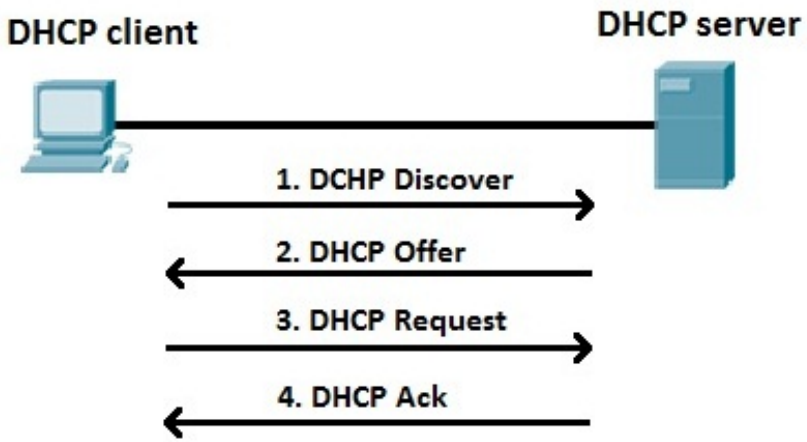
alternativa: DHCP

- invece di assegnare a mano gli ip alle relative interfacce è possibile configurarlo **automaticamente**
- questa configurazione viene effettuata mediante DHCP
  - utile se ci sono tanti host che si collegano e discollegano di continuo
  - permette di gestire meglio gli ip della rete

alternativa: DHCP

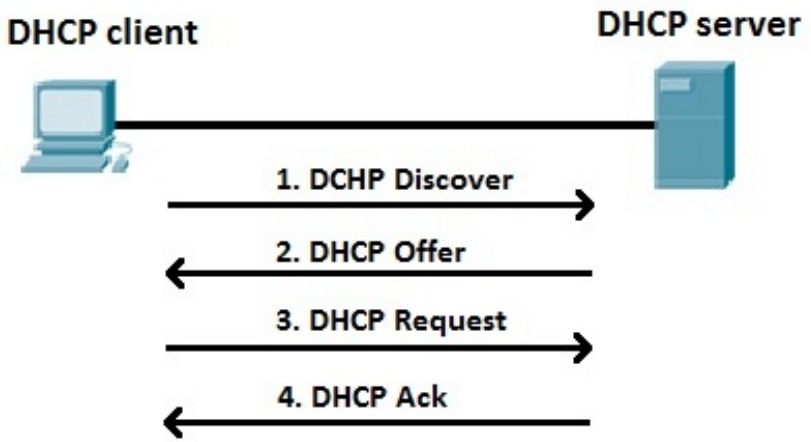
- invece di assegnare a mano gli ip alle relative interfacce è possibile configurarlo **automaticamente**
- questa configurazione viene effettuata mediante DHCP
  - utile se ci sono tanti host che si collegano e discollegano di continuo
  - permette di gestire meglio gli ip della rete

alternativa: DHCP



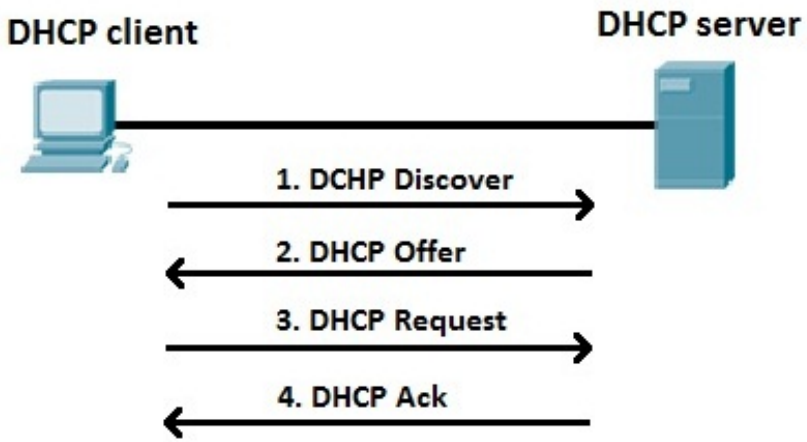
- la rete necessita di un DHCP server, che, su richiesta, rilascia un ip per un certo lasso di tempo (**lease**)
- i client dhcp si trovano solitamente preinstallati (dhclient, dhcpcd)
  - per utilizzare dhcp su di un'interfaccia si usa «dhcp_client» «nome_interfaccia»
- se correttamente configurato il dhcp provvederà a fornire tutte le informazioni necessarie per il corretto funzionamento della rete (Ip address, default gateway, netmask, broadcast)

alternativa: DHCP



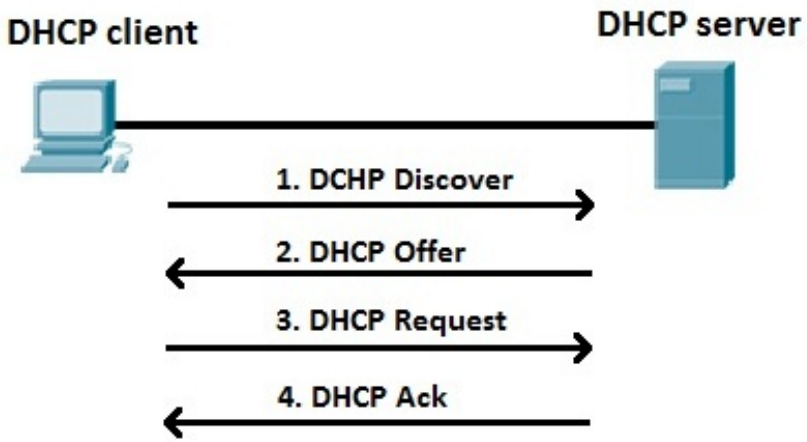
- la rete necessita di un DHCP server, che, su richiesta, rilascia un ip per un certo lasso di tempo (**lease**)
- i client dhcp si trovano solitamente preinstallati (dhclient, dhcpcd)
  - per utilizzare dhcp su di un'interfaccia si usa «dhcp_client» «nome_interfaccia»
- se correttamente configurato il dhcp provvederà a fornire tutte le informazioni necessarie per il corretto funzionamento della rete (Ip address, default gateway, netmask, broadcast)

alternativa: DHCP



- la rete necessita di un DHCP server, che, su richiesta, rilascia un ip per un certo lasso di tempo (**lease**)
- i client dhcp si trovano solitamente preinstallati (dhclient, dhcpcd)
  - per utilizzare dhcp su di un'interfaccia si usa «dhcp_client» «nome_interfaccia»
- se correttamente configurato il dhcp provvederà a fornire tutte le informazioni necessarie per il corretto funzionamento della rete (Ip address, default gateway, netmask, broadcast)

alternativa: DHCP



- la rete necessita di un DHCP server, che, su richiesta, rilascia un ip per un certo lasso di tempo (**lease**)
- i client dhcp si trovano solitamente preinstallati (dhclient, dhcpcd)
  - per utilizzare dhcp su di un'interfaccia si usa «dhcp_client» «nome_interfaccia»
- se correttamente configurato il dhcp provvederà a fornire tutte le informazioni necessarie per il corretto funzionamento della rete (Ip address, default gateway, netmask, broadcast)

Vicinato: rete, gateway e subnet

```
IP Address (numbers):   192.168.0.2
IP Address (bits):      11000000 10101000 00000000 00000010
                        -------- -------- -------- --------
                           192      168       0        2
```

- L'Indirizzo **IP** permette al nostro host di essere raggiunto dagli altri host connessi in rete
- Il modo standard di indicare gli indirizzi IP è la forma decimale puntata...

Vicinato: rete, gateway e subnet

```
IP Address (numbers):   192.168.0.2
IP Address (bits):      11000000 10101000 00000000 00000010
                        -------- -------- -------- --------
                           192      168       0        2
```

- L'Indirizzo **IP** permette al nostro host di essere raggiunto dagli altri host connessi in rete
- Il modo standard di indicare gli indirizzi IP è la forma decimale puntata...

gateway e subnet

```
IP address:    192      168      0        2
            11000000 10101000 00000000 00000010
Netmask:    11111111 11111111 11111111 00000000
               255      255      255        0
            +--------------------------+--------+
                       Network                Host
```

- ...seguita dalla rispettiva **subnet mask** in formato *Variable Lenght Subnet Mask*
- Ci permette di identificare quale host è nella nostra stessa rete locale
- Per gli host esterni dobbiamo mandare i pacchetti al **gateway** che provvederà ad instradarli al di fuori della rete per conto nostro[3]

[3] è scontato che il gateway deve risiedere nella nostra rete locale, altrimenti ci è impossibile contattare la rete esterna

gateway e subnet

```
IP address:    192      168      0        2
            11000000 10101000 00000000 00000010
Netmask:    11111111 11111111 11111111 00000000
               255      255      255        0
            +--------------------------+--------+
                       Network                Host
```

- ...seguita dalla rispettiva **subnet mask** in formato *Variable Lenght Subnet Mask*
- Ci permette di identificare quale host è nella nostra stessa rete locale
- Per gli host esterni dobbiamo mandare i pacchetti al **gateway** che provvederà ad instradarli al di fuori della rete per conto nostro[3]

[3] è scontato che il gateway deve risiedere nella nostra rete locale, altrimenti ci è impossibile contattare la rete esterna

gateway e subnet

```
IP address:    192      168      0        2
            11000000 10101000 00000000 00000010
Netmask:    11111111 11111111 11111111 00000000
               255      255      255        0
            +--------------------------+--------+
                       Network                Host
```

- ...seguita dalla rispettiva **subnet mask** in formato *Variable Lenght Subnet Mask*
- Ci permette di identificare quale host è nella nostra stessa rete locale
- Per gli host esterni dobbiamo mandare i pacchetti al **gateway** che provvederà ad instradarli al di fuori della rete per conto nostro[3]

[3] è scontato che il gateway deve risiedere nella nostra rete locale, altrimenti ci è impossibile contattare la rete esterna

DNS

- Dato che ogni host è identificato da un indirizzo IP per raggiungerlo tramite nome abbiamo bisogno di un servizio che traduca i nomi in indirizzi IP
- DNS (Domain Name Service) effettua questo servizio, per cui è necessario che i name server siano definiti in /etc/resolv.conf

```
$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.0.1
search local
```

- In generale il gateway di default agisce da nameserver, ma alternativamente si possono usare dei nameserver pubblici

Google	Opendns
8.8.8.8	208.67.222.222
8.8.4.4	208.67.220.220

DNS

- Dato che ogni host è identificato da un indirizzo IP per raggiungerlo tramite nome abbiamo bisogno di un servizio che traduca i nomi in indirizzi IP
- DNS (Domain Name Service) effettua questo servizio, per cui è necessario che i name server siano definiti in /etc/resolv.conf

```
$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.0.1
search local
```

- In generale il gateway di default agisce da nameserver, ma alternativamente si possono usare dei nameserver pubblici

Google	Opendns
8.8.8.8	208.67.222.222
8.8.4.4	208.67.220.220

DNS

- Dato che ogni host è identificato da un indirizzo IP per raggiungerlo tramite nome abbiamo bisogno di un servizio che traduca i nomi in indirizzi IP
- DNS (Domain Name Service) effettua questo servizio, per cui è necessario che i name server siano definiti in /etc/resolv.conf

```
$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 192.168.0.1
search local
```

- In generale il gateway di default agisce da nameserver, ma alternativamente si possono usare dei nameserver pubblici

Google	Opendns
8.8.8.8	208.67.222.222
8.8.4.4	208.67.220.220

Recap


The 7 Layered
OSI BURGER
Layer 7
Application Layer
Layer 5
Session Layer
Layer 3
Network Layer
Layer 1
Physical Layer
Layer 6
Presentation Layer
Layer 4
Transport Layer
Layer 2
Data Link Layer

Recap

Network Item	Example
The system IP address	192.168.0.2
Netmask	255.255.255.0
Broadcast	192.168.0.255
Gateway	192.168.0.1
Nameserver(s)	195.130.130.5, 195.130.130.133

Sezione 3

Debugging

- Per diagnostica di rete si utilizza solitamente il comando ping

```
root@Hello:~# ping -c3 poul.org
PING poul.org (91.121.220.9) 56(84) bytes of data.
64 bytes from poul.org (91.121.220.9): icmp_seq=1 ttl=59 time=6.83 ms
64 bytes from poul.org (91.121.220.9): icmp_seq=2 ttl=59 time=6.72 ms
64 bytes from poul.org (91.121.220.9): icmp_seq=3 ttl=59 time=6.68 ms
--- poul.org ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 6.685/6.748/6.839/0.093 ms
```

- Possiamo controllare la corretta risoluzione dei nomi utilizzando il comando *host*

```
$ host poul.org
poul.org has address 91.121.182.81
poul.org has IPv6 address 2001:41d0:1:f751::1
poul.org mail is handled by 1 poul.org.
```

Debugging

- Per diagnostica di rete si utilizza solitamente il comando ping

```
root@Hello:~# ping -c3 poul.org
PING poul.org (91.121.220.9) 56(84) bytes of data.
64 bytes from poul.org (91.121.220.9): icmp_seq=1 ttl=59 time=6.83 ms
64 bytes from poul.org (91.121.220.9): icmp_seq=2 ttl=59 time=6.72 ms
64 bytes from poul.org (91.121.220.9): icmp_seq=3 ttl=59 time=6.68 ms
--- poul.org ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 6.685/6.748/6.839/0.093 ms
```

- Possiamo controllare la corretta risoluzione dei nomi utilizzando il comando *host*

```
$ host poul.org
poul.org has address 91.121.182.81
poul.org has IPv6 address 2001:41d0:1:f751::1
poul.org mail is handled by 1 poul.org.
```

Analisi

- **SS** (*Socket Status*) è un tool che consente di visualizzare tutte le comunicazioni aperte all'interno del nostro host
- **Netcat** è un client/server tcp/udp da linea di comando

```
$ nc [hostname] [port]
```

```
$ nc -l -p [port]
```

- *-l -p [porta]* per restare in ascolto per connessioni sulla porta data

Analisi

- **SS** (*Socket Status*) è un tool che consente di visualizzare tutte le comunicazioni aperte all'interno del nostro host
- **Netcat** è un client/server tcp/udp da linea di comando

```
$ nc [hostname] [port]
```

```
$ nc -l -p [port]
```

- *-l -p [porta]* per restare in ascolto per connessioni sulla porta data

Sezione 4

Routing

- Ogni host GNU/Linux è anche un router
  Per permettere il "forwarding" di pacchetti ip dobbiamo abilitarlo con

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

- Possiamo visualizzare le rotte configurate dal sistema

```
$ ip route show
default via 192.168.0.254 dev eth0
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.2  metric 2
```

- E aggiungerne/rimuoverne

```
$ ip route add 1.2.3.0/24 via 192.168.0.254 dev eth0
$ ip route del 1.2.3.4 via 192.168.0.254 dev eth0
```

- Altri host possono utilizzare noi come router (o gateway di default) e noi provvederemo ad inoltrare il traffico in base alle nostre regole di routing

Routing

- Ogni host GNU/Linux è anche un router
  Per permettere il "forwarding" di pacchetti ip dobbiamo abilitarlo con

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

- Possiamo visualizzare le rotte configurate dal sistema

```
$ ip route show
default via 192.168.0.254 dev eth0
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.2  metric 2
```

- E aggiungerne/rimuoverne

```
$ ip route add 1.2.3.0/24 via 192.168.0.254 dev eth0
$ ip route del 1.2.3.4 via 192.168.0.254 dev eth0
```

- Altri host possono utilizzare noi come router (o gateway di default) e noi provvederemo ad inoltrare il traffico in base alle nostre regole di routing

Routing

- Ogni host GNU/Linux è anche un router
  Per permettere il "forwarding" di pacchetti ip dobbiamo abilitarlo con

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

- Possiamo visualizzare le rotte configurate dal sistema

```
$ ip route show
default via 192.168.0.254 dev eth0
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.2  metric 2
```

- E aggiungerne/rimuoverne

```
$ ip route add 1.2.3.0/24 via 192.168.0.254 dev eth0
$ ip route del 1.2.3.4 via 192.168.0.254 dev eth0
```

- Altri host possono utilizzare noi come router (o gateway di default) e noi provvederemo ad inoltrare il traffico in base alle nostre regole di routing

Routing

- Ogni host GNU/Linux è anche un router
  Per permettere il "forwarding" di pacchetti ip dobbiamo abilitarlo con

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

- Possiamo visualizzare le rotte configurate dal sistema

```
$ ip route show
default via 192.168.0.254 dev eth0
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.2  metric 2
```

- E aggiungerne/rimuoverne

```
$ ip route add 1.2.3.0/24 via 192.168.0.254 dev eth0
$ ip route del 1.2.3.4 via 192.168.0.254 dev eth0
```

- Altri host possono utilizzare noi come router (o gateway di default) e noi provvederemo ad inoltrare il traffico in base alle nostre regole di routing

Firewall

- Un firewall è un “programma” che si occupa di decidere se accettare o meno i pacchetti in transito su un host
- Un firewall serve per:
  - Evitare connessioni non autorizzate
  - Correggere eventuali pacchetti durante il filtraggio degli stessi
  - Applicare strategie di NATting agli host che sono dietro un firewall server
- “Blocchiamo” certi tipi di pacchetti in transito, in base a certe regole

Firewall

- Un firewall è un “programma” che si occupa di decidere se accettare o meno i pacchetti in transito su un host
- Un firewall serve per:
  - Evitare connessioni non autorizzate
  - Correggere eventuali pacchetti durante il filtraggio degli stessi
  - Applicare strategie di NATting agli host che sono dietro un firewall server
- “Blocchiamo” certi tipi di pacchetti in transito, in base a certe regole

Firewall

- Un firewall è un “programma” che si occupa di decidere se accettare o meno i pacchetti in transito su un host
- Un firewall serve per:
  - Evitare connessioni non autorizzate
  - Correggere eventuali pacchetti durante il filtraggio degli stessi
  - Applicare strategie di NATting agli host che sono dietro un firewall server
- “Blocchiamo” certi tipi di pacchetti in transito, in base a certe regole

Firewall

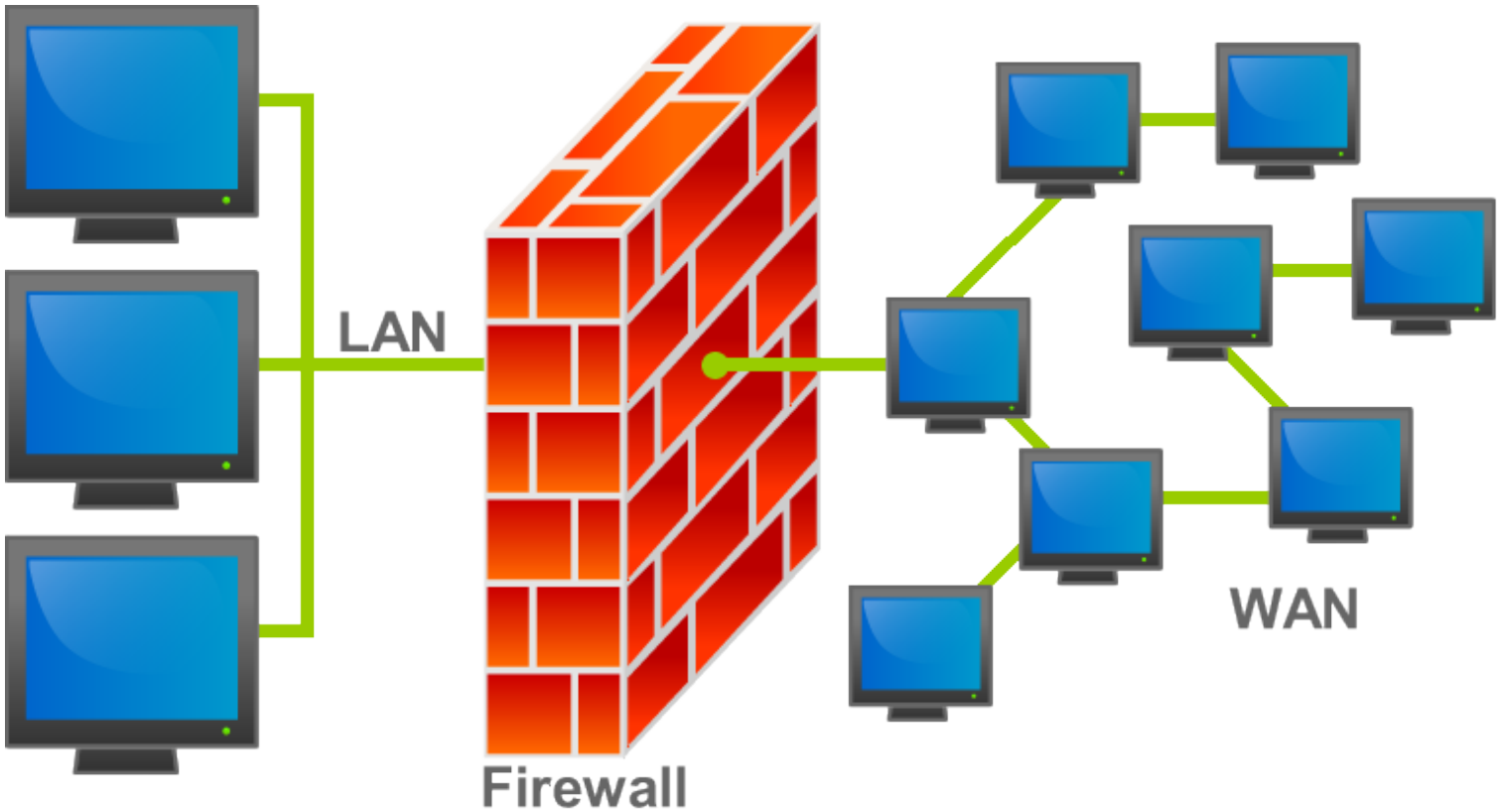
- Un firewall è un “programma” che si occupa di decidere se accettare o meno i pacchetti in transito su un host
- Un firewall serve per:
  - Evitare connessioni non autorizzate
  - Correggere eventuali pacchetti durante il filtraggio degli stessi
  - Applicare strategie di NATting agli host che sono dietro un firewall server
- “Blocchiamo” certi tipi di pacchetti in transito, in base a certe regole

Firewall

- Un firewall è un “programma” che si occupa di decidere se accettare o meno i pacchetti in transito su un host
- Un firewall serve per:
  - Evitare connessioni non autorizzate
  - Correggere eventuali pacchetti durante il filtraggio degli stessi
  - Applicare strategie di NATting agli host che sono dietro un firewall server
- “Blocchiamo” certi tipi di pacchetti in transito, in base a certe regole

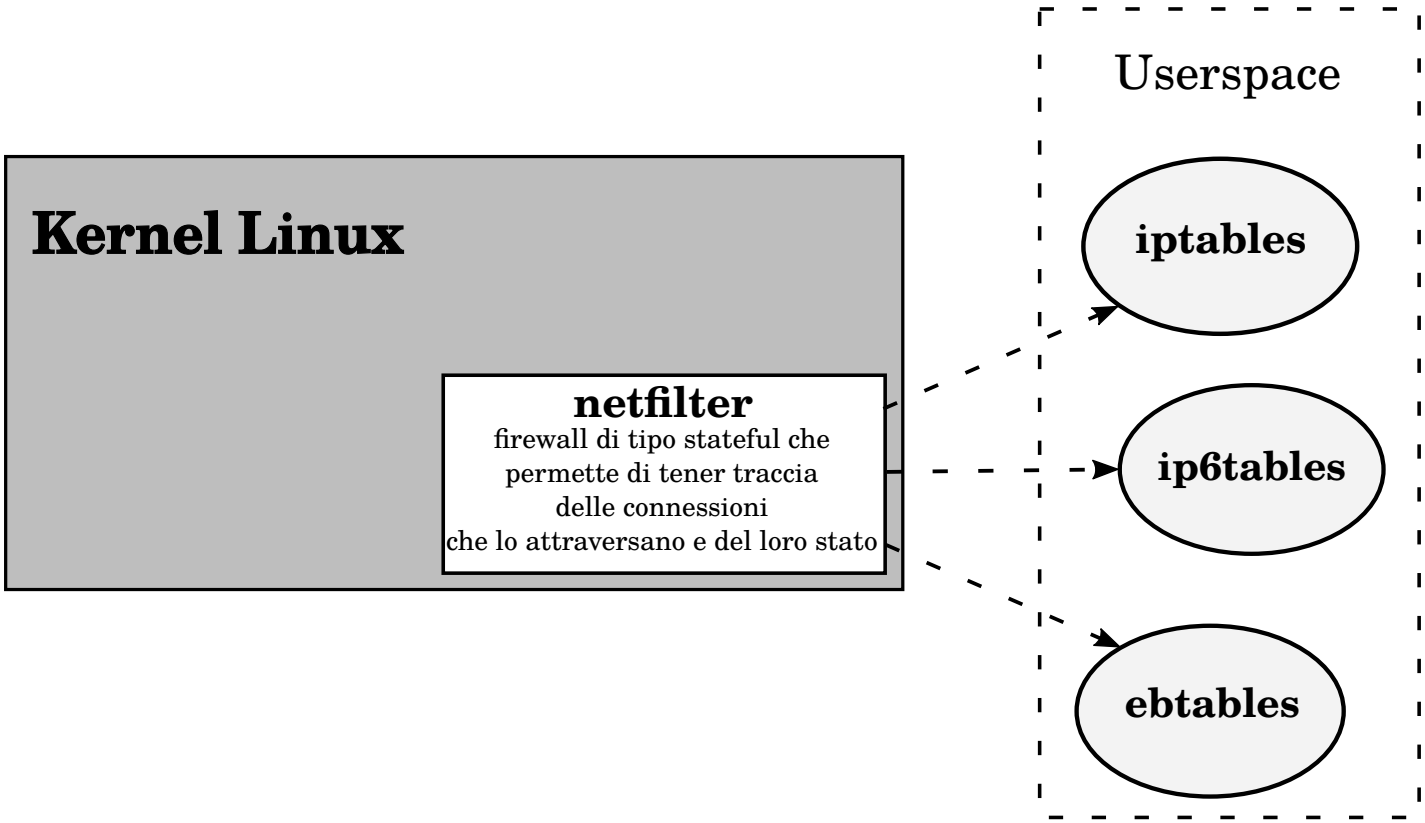
Firewalling

- Il firewall deve essere l'unico punto di contatto della rete con l'esterno[4]



[4]Per evitare il problema di generare *Single Point Of Failure* è eventualmente possibile utilizzare più firewall sincronizzati costantemente tra di loro: `http://conntrack-tools.netfilter.org/`

Firewalling



- Tramite i tool in userspace gli utenti possono configurare il firewall per dare accesso selettivo al sistema, bloccando ad esempio alcune porte TCP, o restringendo l'accesso a certi indirizzi IP [5]

[5] A breve si migrerà ad un nuovo unico tool, chiamato **nftables**, che tuttavia non modifica la struttura delle tabelle e hooks che vedremo tra un attimo.

Netfilter

- L'infrastruttura di Netfilter è basata su 5 "hooks" presenti nel percorso che i pacchetti devono compiere all'interno del sistema.
- In ognuno di questi "punti" è possibile
  - Permettere il passaggio del pacchetto (ACCEPT)
  - Scartare il pacchetto e interromperlo (DROP)
  - Redirigere il pacchetto (REDIRECT)
  - Modificare alcuni campi del pacchetto (MANGLED)

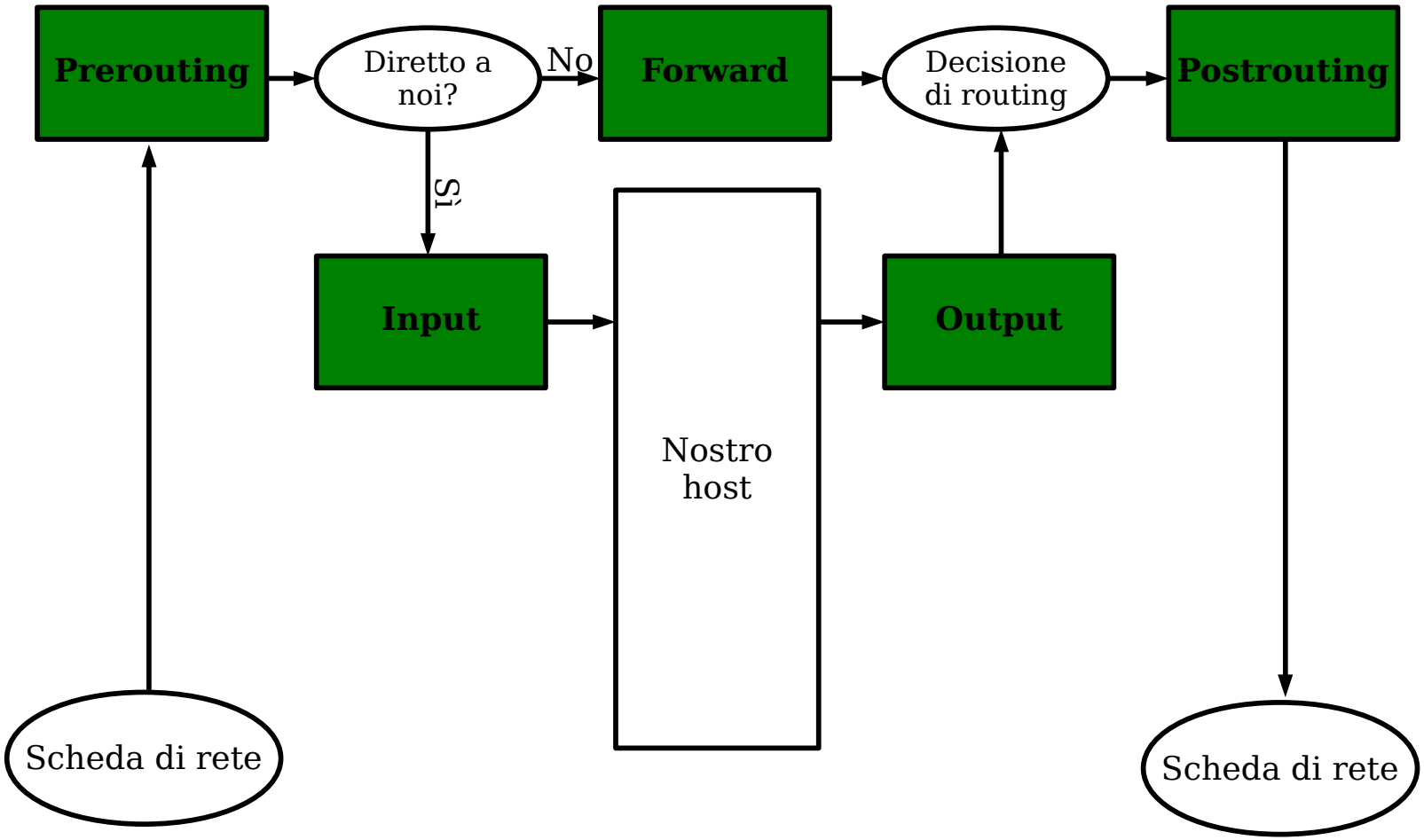
Netfilter

- L'infrastruttura di Netfilter è basata su 5 "hooks" presenti nel percorso che i pacchetti devono compiere all'interno del sistema.
- In ognuno di questi "punti" è possibile
  - Permettere il passaggio del pacchetto (ACCEPT)
  - Scartare il pacchetto e interromperlo (DROP)
  - Redirigere il pacchetto (REDIRECT)
  - Modificare alcuni campi del pacchetto (MANGLED)

Netfilter

- L'infrastruttura di Netfilter è basata su 5 "hooks" presenti nel percorso che i pacchetti devono compiere all'interno del sistema.
- In ognuno di questi "punti" è possibile
  - Permettere il passaggio del pacchetto (ACCEPT)
  - Scartare il pacchetto e interromperlo (DROP)
  - Redirigere il pacchetto (REDIRECT)
  - Modificare alcuni campi del pacchetto (MANGLED)

Netfilter


Prerouting
Diretto a noi?
No
Forward
Decisione di routing
Postrouting
Si
Input
Output
Nostro host
Scheda di rete
Scheda di rete

Netfilter

- Gli “hooks” non possono essere utilizzati direttamente.
- Le regole devono essere inserite all’interno di **tabelle** con dei *punti di aggancio* agli hook: le **chain**
- Le tabelle utilizzate sono: *filter*, *mangle*, *nat* e *raw*
  - filter è la tabella con le regole principali
  - mangle è la tabella dove risiedono le regole che modificano i pacchetti
  - nat è la tabella dove avvengono le operazioni che permettono il NAT

Netfilter

- Gli “hooks” non possono essere utilizzati direttamente.
- Le regole devono essere inserite all’interno di **tabelle** con dei *punti di aggancio* agli hook: le **chain**
- Le tabelle utilizzate sono: *filter*, *mangle*, *nat* e *raw*
  - filter è la tabella con le regole principali
  - mangle è la tabella dove risiedono le regole che modificano i pacchetti
  - nat è la tabella dove avvengono le operazioni che permettono il NAT

Netfilter

- Gli “hooks” non possono essere utilizzati direttamente.
- Le regole devono essere inserite all’interno di **tabelle** con dei *punti di aggancio* agli hook: le **chain**
- Le tabelle utilizzate sono: *filter*, *mangle*, *nat* e *raw*
  - filter è la tabella con le regole principali
  - mangle è la tabella dove risiedono le regole che modificano i pacchetti
  - nat è la tabella dove avvengono le operazioni che permettono il NAT

Netfilter

- Gli “hooks” non possono essere utilizzati direttamente.
- Le regole devono essere inserite all’interno di **tabelle** con dei *punti di aggancio* agli hook: le **chain**
- Le tabelle utilizzate sono: *filter*, *mangle*, *nat* e *raw*
  - filter è la tabella con le regole principali
  - mangle è la tabella dove risiedono le regole che modificano i pacchetti
  - nat è la tabella dove avvengono le operazioni che permettono il NAT

Netfilter

- Gli “hooks” non possono essere utilizzati direttamente.
- Le regole devono essere inserite all’interno di **tabelle** con dei *punti di aggancio* agli hook: le **chain**
- Le tabelle utilizzate sono: *filter*, *mangle*, *nat* e *raw*
  - filter è la tabella con le regole principali
  - mangle è la tabella dove risiedono le regole che modificano i pacchetti
  - nat è la tabella dove avvengono le operazioni che permettono il NAT

Recap

- il pacchetto segue il suo percorso
- quando il pacchetto arriva in un hook si vanno a consultare le varie tabelle che hanno regole in quell'hook
- il pacchetto entra nella *chain* di una tabella e lì vengono consultate tutte le regole presenti una dopo l'altra
- se al termine delle verifiche, la decisione è "ACCEPT", allora il pacchetto prosegue il suo percorso verso la destinazione
- passando nell'hook successivo

Recap

- il pacchetto segue il suo percorso
- quando il pacchetto arriva in un hook si vanno a consultare le varie tabelle che hanno regole in quell'hook
- il pacchetto entra nella *chain* di una tabella e lì vengono consultate tutte le regole presenti una dopo l'altra
- se al termine delle verifiche, la decisione è "ACCEPT", allora il pacchetto prosegue il suo percorso verso la destinazione
- passando nell'hook successivo

Recap

- il pacchetto segue il suo percorso
- quando il pacchetto arriva in un hook si vanno a consultare le varie tabelle che hanno regole in quell'hook
- il pacchetto entra nella *chain* di una tabella e lì vengono consultate tutte le regole presenti una dopo l'altra
- se al termine delle verifiche, la decisione è "ACCEPT", allora il pacchetto prosegue il suo percorso verso la destinazione
- passando nell'hook successivo

Recap

- il pacchetto segue il suo percorso
- quando il pacchetto arriva in un hook si vanno a consultare le varie tabelle che hanno regole in quell'hook
- il pacchetto entra nella *chain* di una tabella e lì vengono consultate tutte le regole presenti una dopo l'altra
- se al termine delle verifiche, la decisione è "ACCEPT", allora il pacchetto prosegue il suo percorso verso la destinazione
- passando nell'hook successivo

Recap

- il pacchetto segue il suo percorso
- quando il pacchetto arriva in un hook si vanno a consultare le varie tabelle che hanno regole in quell'hook
- il pacchetto entra nella *chain* di una tabella e lì vengono consultate tutte le regole presenti una dopo l'altra
- se al termine delle verifiche, la decisione è "ACCEPT", allora il pacchetto prosegue il suo percorso verso la destinazione
- passando nell'hook successivo

Netfilter/Iptables

- Questo sistema di packet filtering usa **iptables** come interfaccia utente.
- All'arrivo di un pacchetto nel sistema, esso viene preso in consegna da netfilter per **accettarlo**, **scartarlo**, eventualmente **manipolarlo** seguendo le regole impostate tramite iptables.
- Una regola ha due parti, un **match** e un **target**

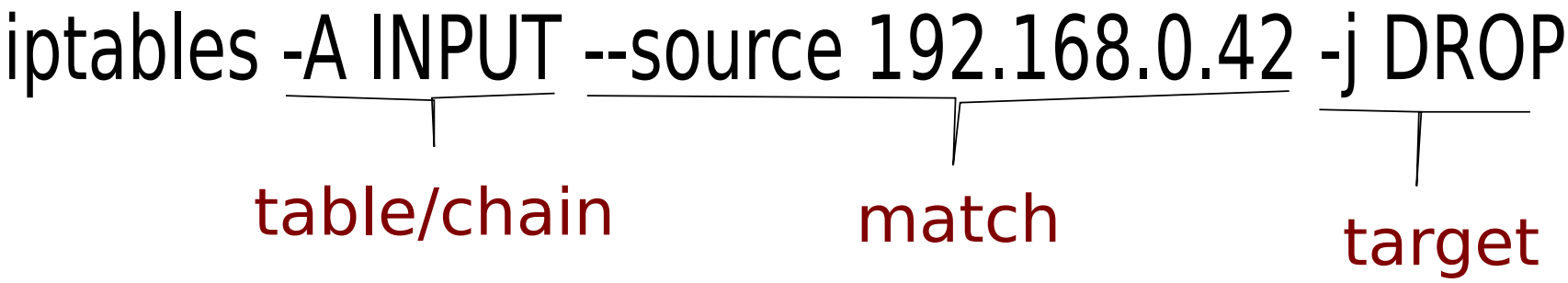
Netfilter/Iptables

- Questo sistema di packet filtering usa **iptables** come interfaccia utente.
- All'arrivo di un pacchetto nel sistema, esso viene preso in consegna da netfilter per **accettarlo**, **scartarlo**, eventualmente **manipolarlo** seguendo le regole impostate tramite iptables.
- Una regola ha due parti, un **match** e un **target**

Netfilter/Iptables

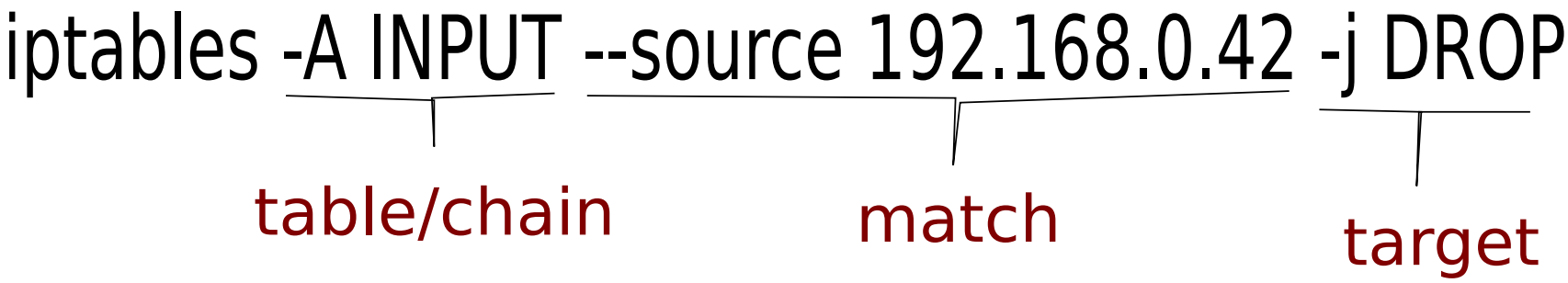
- Questo sistema di packet filtering usa **iptables** come interfaccia utente.
- All'arrivo di un pacchetto nel sistema, esso viene preso in consegna da netfilter per **accettarlo**, **scartarlo**, eventualmente **manipolarlo** seguendo le regole impostate tramite iptables.
- Una regola ha due parti, un **match** e un **target**

Iptables rulez



- **-A** significa che vogliamo aggiungere questa regola **in coda alle altre** nella chain **INPUT**
- **–source** indica che la regola vale solo per i pacchetti IPv4 provenienti dall' indirizzo sorgente 192.168.0.42
- **-j DROP** specifica che se le precedenti condizioni di match sono soddisfatte, scartiamo il pacchetto che non proseguirà nelle altre regole

Iptables rulez



- **-A** significa che vogliamo aggiungere questa regola **in coda alle altre** nella chain **INPUT**
- –**source** indica che la regola vale solo per i pacchetti IPv4 provenienti dall' indirizzo sorgente 192.168.0.42
- **-j DROP** specifica che se le precedenti condizioni di match sono soddisfatte, scartiamo il pacchetto che non proseguirà nelle altre regole

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

Forgive me otacon

- **Iptables**
  - Iptables è il database di regole nonchè attuale firewall usato nei sistemi GNU/Linux
  - Tradizionalmente si usa configurare iptables tramite linea di comando.
  - Esistono vari frontend per alleggerire il compito dei sysadmin
- **UFW**
  - Uncomplicated Firewall è un frontend per iptables indicato sia per desktop che per server.
  - Configurabile da linea di comando
  - Disponibile per varie distribuzioni
  - Possiede un front-end grafico: **Gufw**

UFW: basic

All'avvio si configura autometicamente con un set base i regole che bloccano i pacchetti in **ingresso**

Accendere e spegnere

- *sudo ufw enable* attiva ufw con il set di regole in /etc/ufw/.rules
- *sudo ufw disable*
- *sudo ufw status verbose*[a]

[a] Controllare lo stato di ufw permette di verificare se è abilitato o meno e la lista di regole correntemente applicate

UFW: basic

All'avvio si configura autometicamente con un set base i regole che bloccano i pacchetti in **ingresso**

Accendere e spegnere

- *sudo ufw enable* attiva ufw con il set di regole in /etc/ufw/.rules
- *sudo ufw disable*
- *sudo ufw status verbose*[a]

[a] Controllare lo stato di ufw permette di verificare se è abilitato o meno e la lista di regole correntemente applicate

UFW: rules

rules

- *sudo ufw show raw* mostra le regole attualmente in uso
- *sudo ufw allow «port»/ «protocol*[a]*»* permette di aggungere una nuova regola
- *sudo ufw deny «port»/ «protocol»*
- *sudo ufw delete «port»/ «protocol»*

[a] *opzionale*

Hint le regole possono essere modificate manualmente dal file di configurazione[6]

[6] /etc/ufw/.rules

UFW: services

Con UFW è possibile permettere o bloccare l'accesso utilizzando direttamente i nomi dei servizi[7]

Services

- *sudo ufw allow «service name»* Abilitare i servizi (esempio *sudo allow ssh*)
- *sudo ufw deny «service name»*

Logging

- *sudo ufw logging on* abilita di log
- *sudo ufw logging off* disabilita i log

[7] Per ottenere una lista dei servizi basta usare

```
less /etc/services
```

Qualche cenno in più

Sintassi avanzata

- sudo ufw allow from «ip»/ «netmask»
- *sudo ufw allow from «target» to «destination» port «port number» proto «protocol name»*
- sudo ufw deny from «ip address» to «protocol» port «port number»

Lavorare sulle posizioni

- sudo ufw status numbered
- sudo ufw delete 1
- sudo ufw insert 1 allow from <ip address>

Qualche cenno in più

- Fate molta attenzione se state configurando una macchina remota quando impostate le regole!
- Ad esempio se siete connessi in ssh e avviate ufw non dovete rimuovere la regola per permettere l'input tramite porta 22

Qualche cenno in più

- Fate molta attenzione se state configurando una macchina remota quando impostate le regole!
- Ad esempio se siete connessi in ssh e avviate ufw non dovete rimuovere la regola per permettere l'input tramite porta 22

Qualche cenno in più

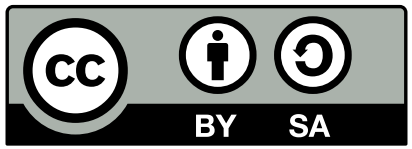
- Fate molta attenzione se state configurando una macchina remota quando impostate le regole!
- Ad esempio se siete connessi in ssh e avviate ufw non dovete rimuovere la regola per permettere l'input tramite porta 22

Fonti

- Le altre mie slide sul Networking (Hard Version)
- Linux Administration Handbook
- Manpages
- `https://help.ubuntu.com/community/Firewall`
- `https://wiki.gentoo.org/wiki/Handbook:AMD64/Installation/Networking`
- `http://www.penguintutor.com/linux/basic-network-reference`
- Slides su firewalling su poul.org
  - `http://www.poul.org/wp-content/uploads/2012/05/presentazione_netfilter.pdf`
  - `https://www.poul.org/wp-content/uploads/2014/04/Networking.pdf`
  - `https://www.poul.org/wp-content/uploads/2013/03/slides.pdf`

Fine

Grazie per l'attenzione!



`http://www.poul.org`