IV Jornadas Sistema Operativo Linux

Cortafuegos en Linux con iptables

Andrés J. Díaz <ajdiaz@gpul.org>

¿Para qué un cortafuegos doméstico?

Lo que puede hacer

- Evitar en la medida de lo posible ataques DoS
- Controlar el acceso a la máquina
- Detectar posibles usos fraudulentos
- Monitorizar el tráfico de nuestro equipo
- Controlar el acceso a servicios privilegiados

Lo que no hace

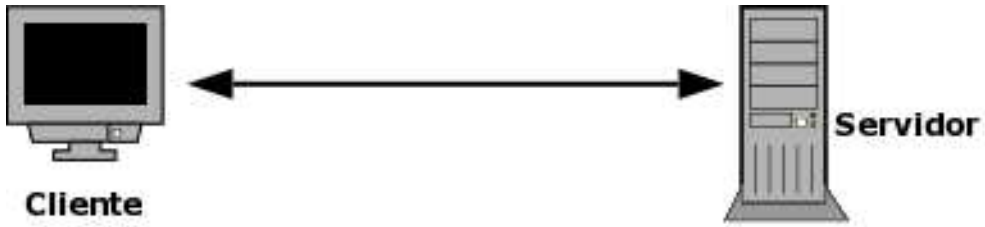
- No evita virus/gusanos/troyanos/*anos
- No detecta intrusos (para eso IDS)
- No monitoriza el tráfico de la red (salvo promiscuos)

Los peligros en la Red (I)

En el mundo de Fresita todo era felicidad:

- Los clientes enviaban peticiones a los servidores
- Los servidores contestaban a los clientes

- Los SO no tenían fallos explotables
- La comunicaciones eran seguras

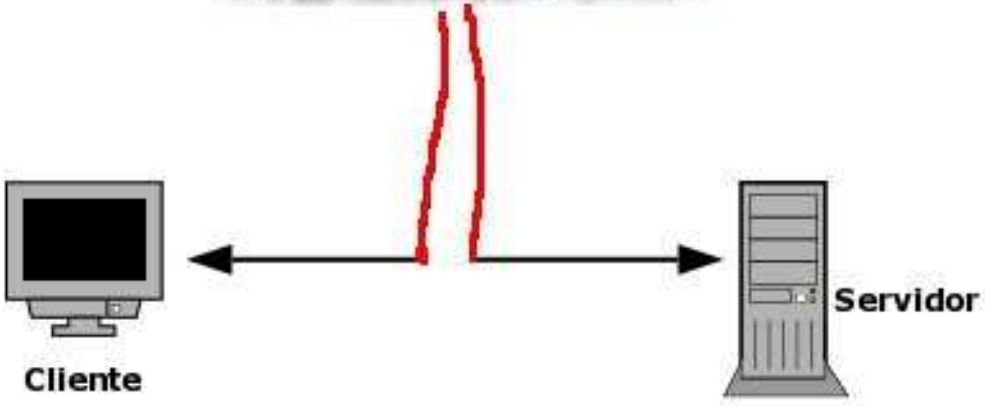


Pero...

Los peligros en la red (II)

Apareció el temible ¡Capitán Piratilla!:

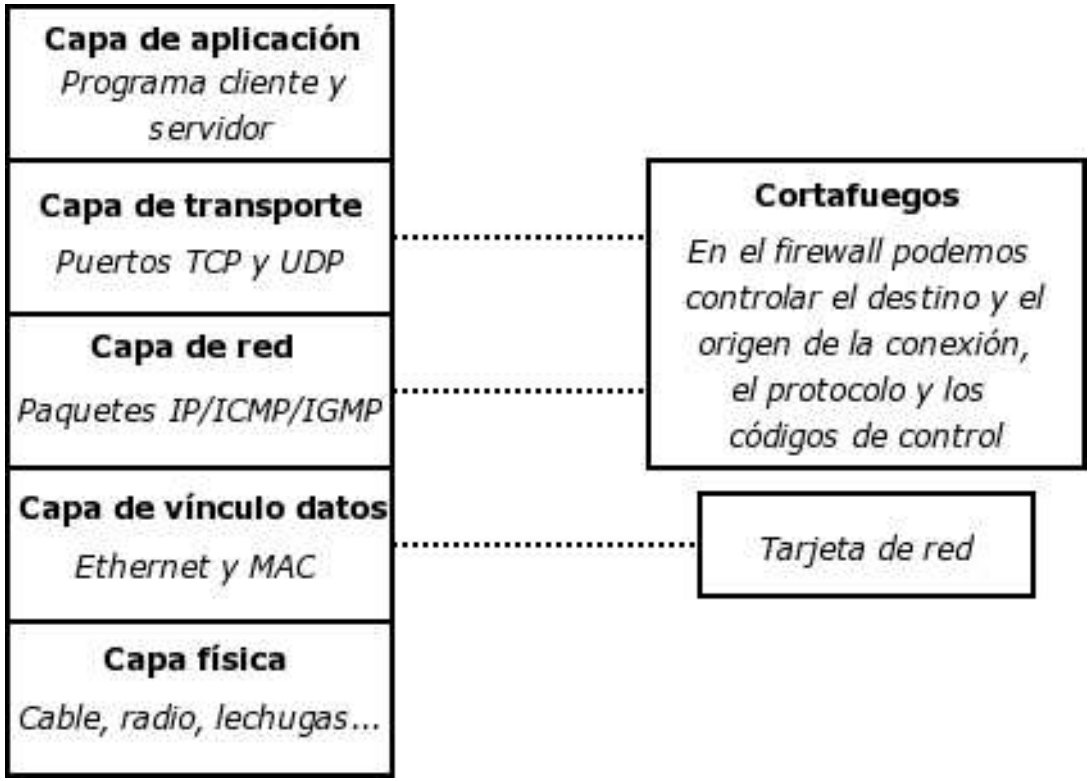
- El piratilla se introduce en la comunicación
- Falsea direcciones
- Aprovecha vulnerabilidades
- Quiere hundir nuestra máquina :-(



¿Dónde actúa el firewall?

Firewall:

- Entre la capa de transporte y la capa de red



Historia

Kernel 1.1

- ipfw de BSD adaptado - Alan Cox (1994)
- Trabajaba en el espacio de kernel (ring0)

Kernel 2.0

- ipfwadm - Jos Vos (1996)
- Ya en el espacio de usuario

Kernel 2.2

- ipchains - Paul "Rusty" Russel et al (1998)

Kernel 2.4

- iptables - Paul "Rusty" Russel (1999)

Kernel 2.6

- iptables mejorado - Paul "Rusty" Russel (2003)

Los dos iptables

OJO: Dos iptables

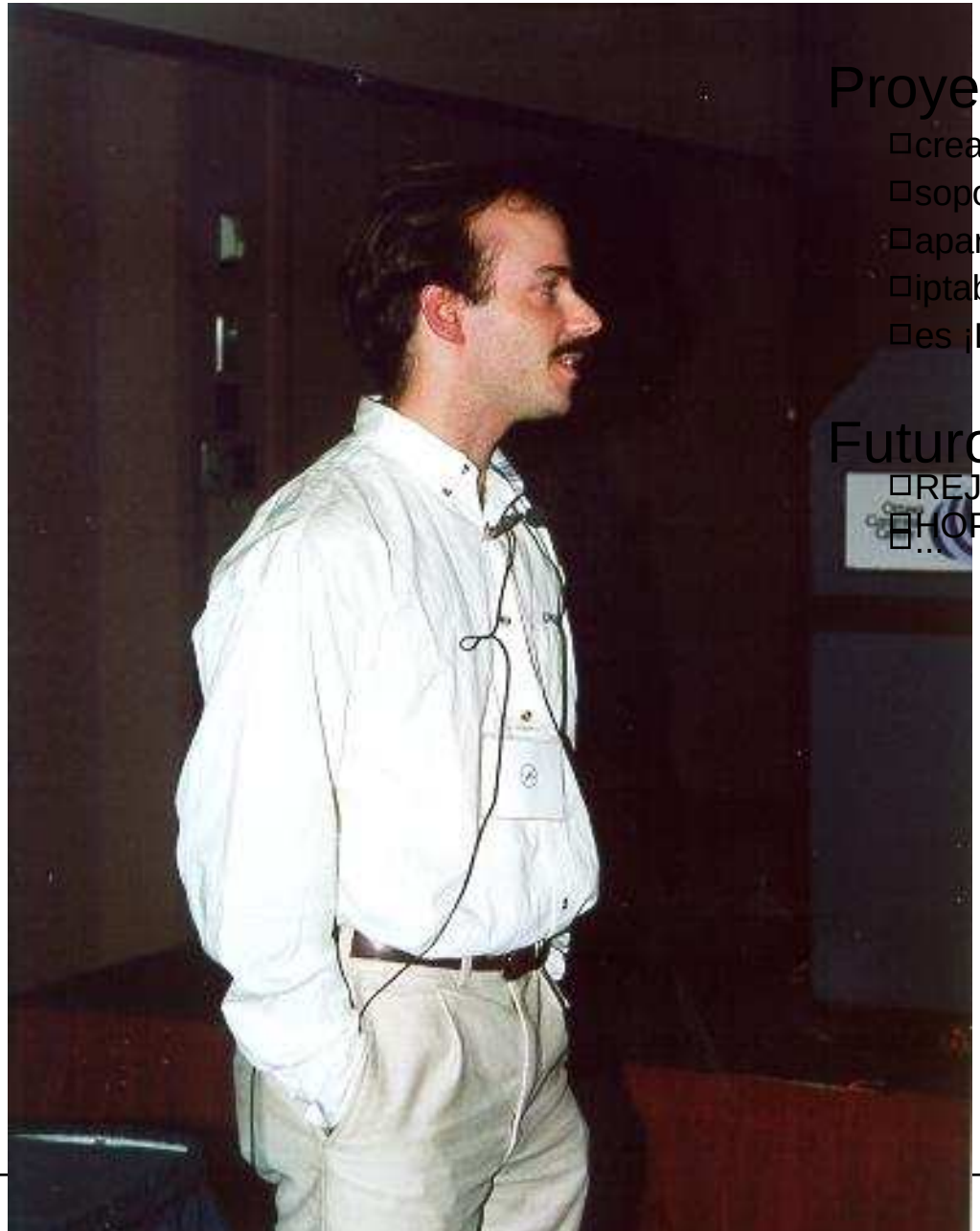
En genérico:

- iptables = cortafuegos y filtrado para Linux
- iptables = herramienta (ejecutable)

iptables es:

- el sistema de filtrado para kernel 2.4 y 2.6
- dentro del proyecto netfilter
- se maneja con la herramienta iptables

Netfilter e iptables (I)



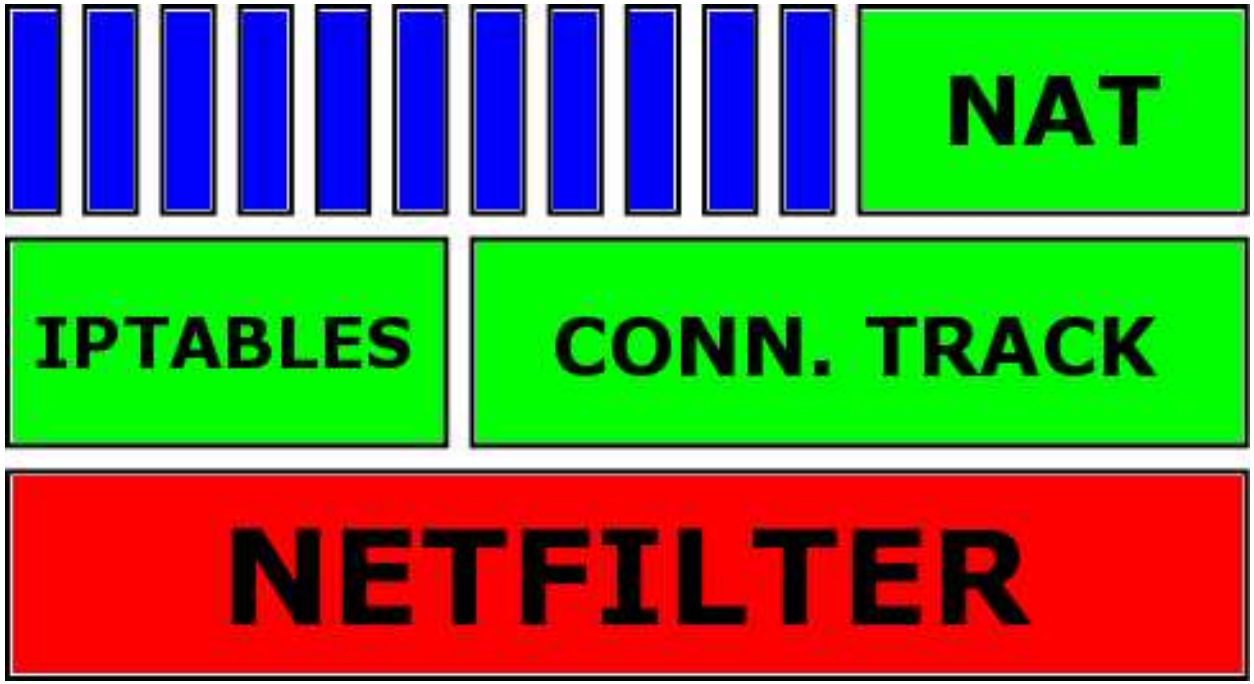
Proyecto Netfilter:

- creado por Paul "Rusty" Russel
- soporte firewall para Linux
- aparece en el kernel 2.4
- iptables es parte de netfilter
- es ¡LIBRE! (GPL)

Futuro:

- REJECT en IPv6
- HOPLIMIT en IPv6
- ...

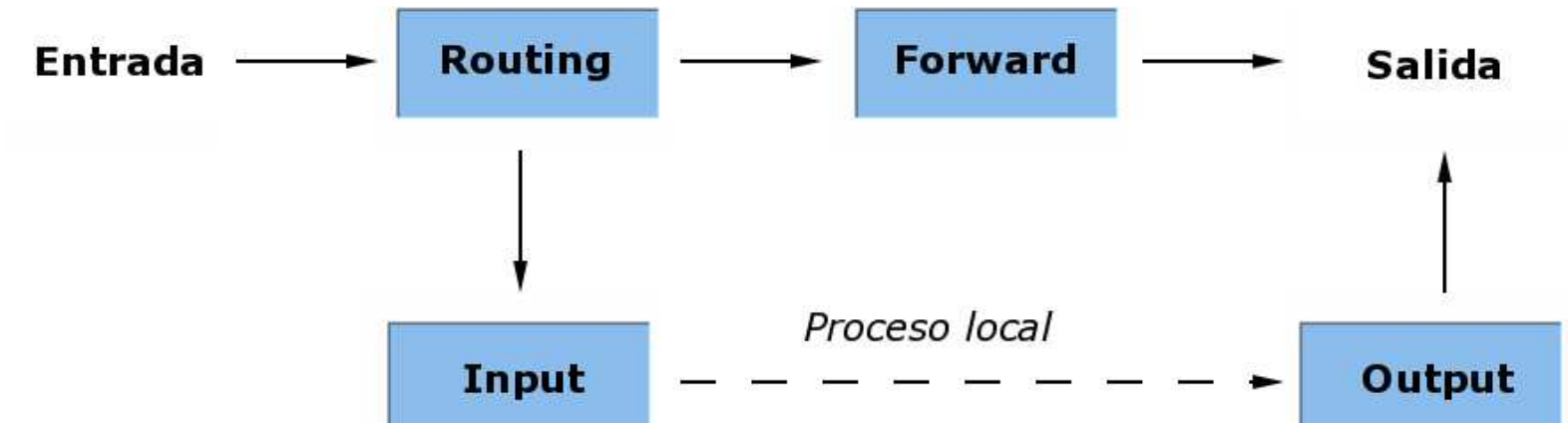
Netfilter e iptables (y II)



Netfilter Framework

- Nos centraremos en la parte iptables
- Se están añadiendo nuevos elementos
  - QoS, IPv6 ...

¿Cómo funciona?



Decisión de entrada

- Si el destino es esta máquina: INPUT
- Si el destino es otra máquina: FORWARD (NAT)

Proceso local

- El servidor, cliente, demonio, aplicación...

Reglas de cortafuegos (I)

Una regla es:

- una cadena de entrada
- una cadena de salida
- una cadena de reenvio (sólo NAT)
- una cadena definida por el usuario

Evaluación

- Si un paquete concuerda con la regla se manda a la "cadena de salida"
- Si un paquete no concuerda, se examina la siguiente regla

Reglas de cortafuegos (II)

Reglas internas (o listas de reglas):

- INPUT: Todo lo que quiera entrar en nuestra máquina
- OUTPUT: Todo lo que quiera salir de nuestra máquina
- FORWARD: Todo lo que se quiera redirigir a otra máquina
- PREROUTING: Lo que se hará antes de encaminar el paquete
- POSTROUTING: Lo que se hará inmediatamente después de encaminar el paquete

Pero:

- Podemos crear nuestras listas de reglas
- Podemos combinarlas con las reglas internas
- Podemos añadir y borrar cadenas de las reglas

Reglas de cortafuegos (III)

Objetivos (cadenas de salida)

- ACCEPT: El paquete es "legal" y sale de INPUT
- DROP: El paquete se descarta (nos lo comememos)
- LOG: El paquete será registrado (NO TIENE PORQUE DESCARTARSE)
- REJECT: Se envía un ICMP de rechazo

Obviamente:

- Podemos usar nuestras propias reglas como objetivos

Reglas de cortafuegos (y IV)

En general:

- iptables -A regla opciones -j objetivo

Ejemplo tonto:

- iptables --append INPUT --jump DROP
- iptables -A INPUT -j DROP

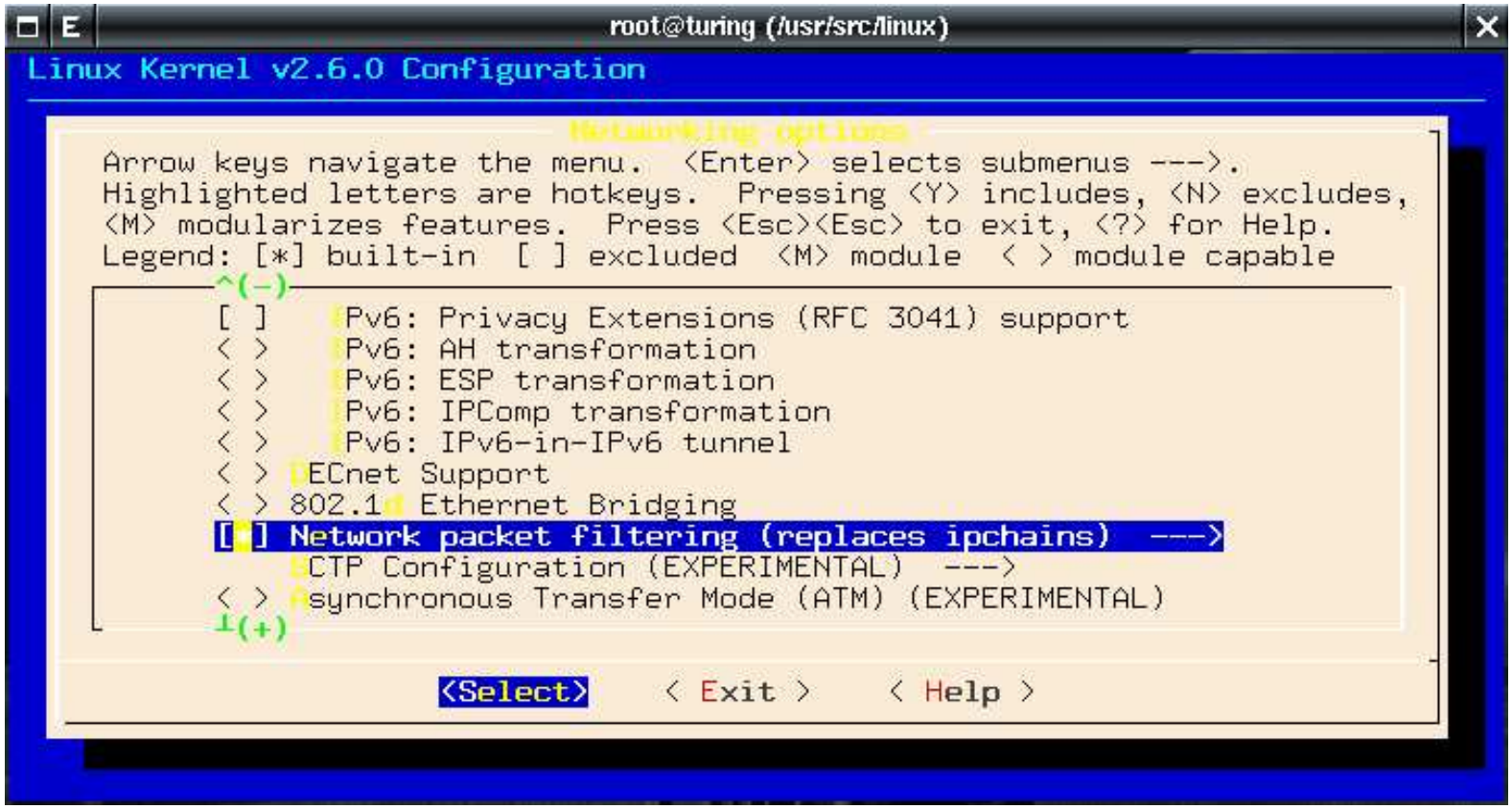
¿Qué hace?

- Añade una nueva regla a la lista INPUT
- Lo que quiera entrar en la máquina se manda "DROP" (al carajo)

Configurando el kernel (I)

make menuconfig

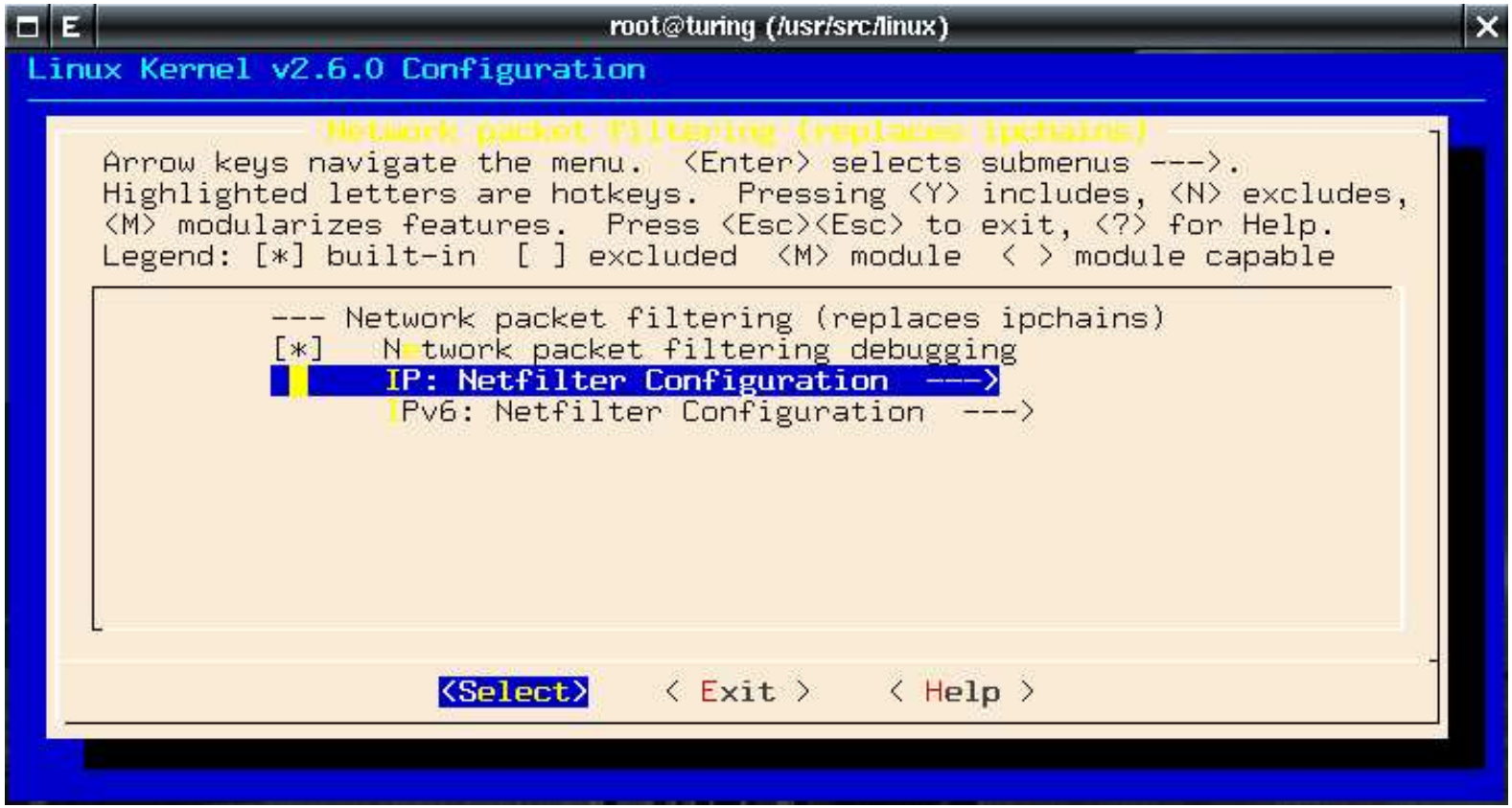
- Device Drivers / Networking Support / Networking options



Configurando el kernel (II)

IPv4
- IP Netfilter configuration

IPv6
- IP6 Netfilter configuration



Configurando el kernel (y III)

¿Que debemos activar?

- IP tables support (SIEMPRE)
- limit match support
- Multiple port match support
- TOS match support (para el Type of Service)
- TTL match support (tiempo de vida)
- Connection state match support (controlar conexiones muertas)
- Packet filtering
- REJECT target support
- Packet mangling
- TOS target support (para el Type of Service)
- LOG target support (para registros)

Funcionamiento modular

iptables es modular

- necesitamos cargar un módulo para ciertas acciones
- algunos módulos se cargan solos
  - protocolos icmp, tcp, udp ...
- otros es necesario importarlos manualmente
- podemos crear nuestros propios módulos

Cargar modulos manualmente:

- iptables -m <modulo> <resto de la regla>

Errores comunes:

- Argumento desconocido

Tablas en iptables

iptables tiene 3 tablas

- filter: iptables actúa como cortafuegos
- nat: iptables actúa como router
- mangle: altera paquetes especiales

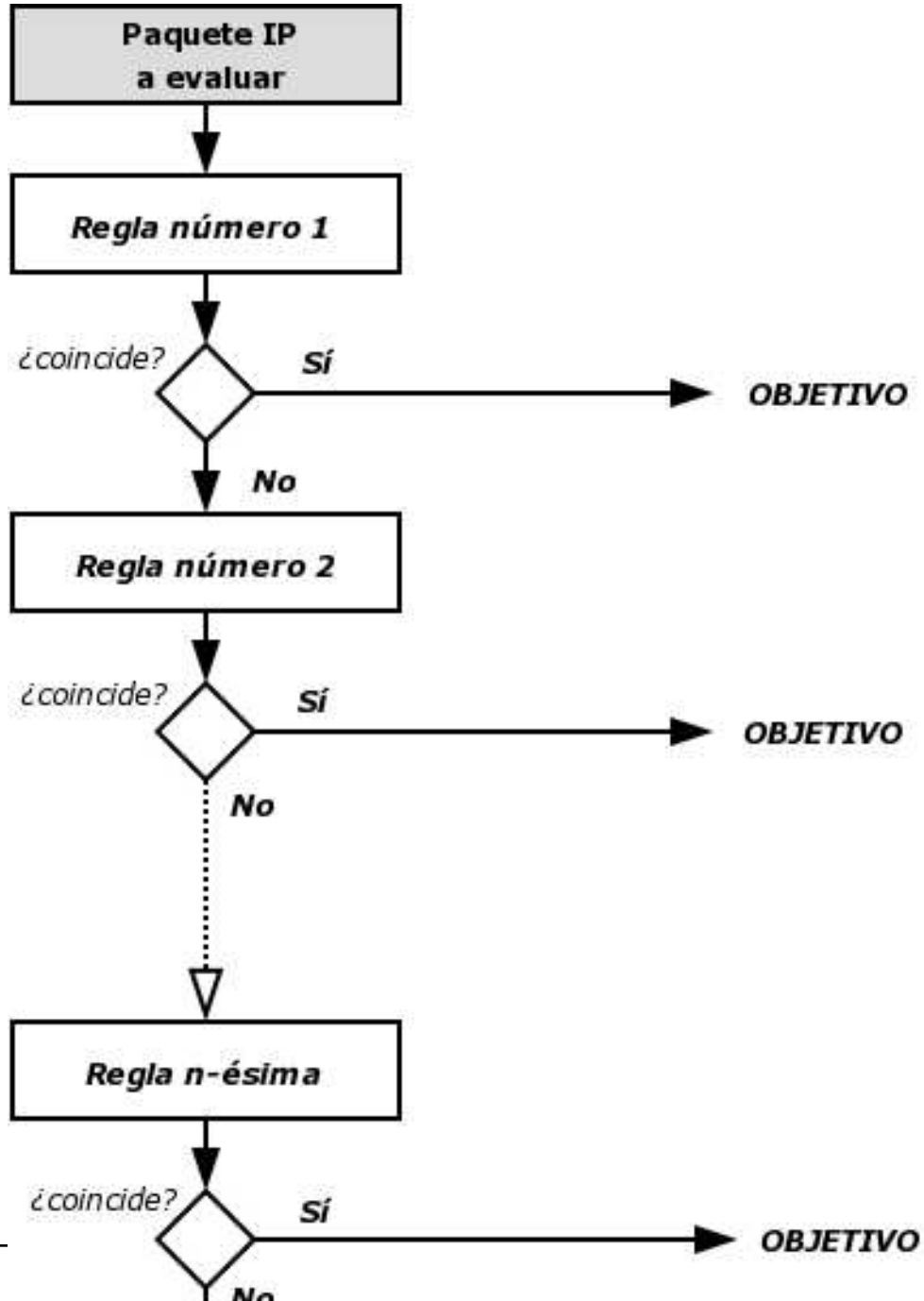
Nosotros usaremos sólo

- filter: para filtrado habitual
- mangle: para especificar TOS

Las tablas se especifican con -t:

- iptables -t filter <regla>
- por defecto se usa filter

¿Cómo se filtra con iptables?



Filtrando por interfaz

-i, --in-interface [!] nombre (para INPUT)
-o, --out-interface [!] nombre (para OUTPUT)

nombre ha de ser el nombre de una interfaz del sistema:

- ppp0
- eth0
- ...

Ejemplos:

- iptables --add INPUT --in-interface lo -j ACCEPT
- iptables --add INPUT --in-interface ! lo -j DROP

OJO: lo debe aceptarse

Filtrando por protocolo

-p, --protocol [!] protocolo

protocolo debe ser:

- udp: para filtrar UDP
- tcp: para filtrar TCP
- icmp: para filtrar ICMP
- all: para todos los protocolos

- cualquier otro definido en /etc/protocols

Ejemplos:

- iptables --add INPUT --protocol tcp --jump DROP
- iptables --add INPUT --protocol udp --jump ACCEPT
- iptables --add INPUT --protocol icmp --jump EVALUAR

Filtrando por dirección

Dirección de destino:

- -d, --destination [!] dirección[/máscara]

Dirección de origen:

- -s, --source [!] dirección[/máscara]

Ejemplo:

- iptables -A INPUT --destination 192.168.0.2 -j DROP
- iptables -A INPUT --destination 192.168.0.1/255.255.255.0 -j DROP
- iptables -A INPUT --destination ! 192.168.0.2/16 -j ACCEPT

- iptables -A INPUT --source 66.66.66.66 -j DROP
- iptables -A INPUT --source 69.69.69.69/255.255.255.0 -j ACCEPT
- iptables -A INPUT --source ! 192.168.0.0/16 -j DROP

Direcciones ilegales

Direcciones de red locales:

- Clase A: 10.0.0.0 - 10.255.255.255
- Clase B: 172.16.0.0 - 172.31.255.255
- Clase C: 192.168.0.0 - 192.168.255.255

Direcciones multidifusión:

- Clase D: 224.0.0.0 - 239.255.255.255

Direcciones militares:

- Clase E: 240.0.0.0 - 247.255.255.255

Direcciones de «bucle invertido»:

- 127.0.0.1 - 127.255.255.255 (OJO CON LA INTERFAZ LO)

Direcciones mal formadas:

- 0.0.0.0 (OJO CON DHCP)

Filtrando por puerto

OJO: Sólo UDP y TCP

Puerto de destino:

- -dport, --destination-port [!] puerto[:puerto]

Puerto de origen:

- -sport, --source-port [!] puerto[:puerto]

Ejemplos:

- iptables --add INPUT --protocol tcp --destination-port 25 -j DROP
- iptables --add INPUT --protocol tcp --destination-port !25 -j ACCEPT
- iptables --add INPUT --protocol tcp --destination-port 21:25 -j DROP

- iptables --add INPUT --protocol udp --source-port 80 -j DROP
- iptables --add INPUT --protocol udp --source-port !80 -j ACCEPT
- iptables --add INPUT --protocol udp --source-port 60:80 -j DROP

Filtrando por tipo de ICMP (I)

OJO: Sólo ICMP

--icmp-type [!] nombre-de-tipo

donde nombre-de-tipo es:

- echo-reply (pong)
- destination-unreachable
- echo-request
- time-exceeded
- ...

Ejemplos:

- iptables -A INPUT --protocol icmp --icmp-type echo-reply -j DROP
- iptables -A INPUT -p icmp --icmp-type !destination-unreacheable -j ACCEPT

Filtrando por tipo de ICMP (y II)

¡Si bloqueas todos los paquetes ICMP, te irás directo al infierno!

Filtro por límites

-m limit --limit <ratio>

donde <ratio> puede ser, siendo x un número:

- x/second
- x/minute
- x/hour (por defecto 3/hour)
- x/day

Ejemplo:

- iptables -a INPUT -p icmp -m limit --limit 5/minute -j ACCEPT

Filtro por banderas TCP

--tcp-flags [!] máscara-tcp-flags tcp-flags-activos

donde máscara-tcp-flags y tcp-flags-activos son:

- SYN ACK FIN RST URG PSH ALL NONE
- separados por comas

```
se evalúan todas las banderas de má
scara-tcp-flags
si tcp-flags-activos están activas,
el resto de
banderas deben estar desactivadas.
```

Ejemplos

- iptables -A -p tcp --tcp-flags SYN,ACK SYN -j DROP
  - Bandera SYN activa y ACK desactivada

Filtro por estado de conexion

-m state --state estado

donde estado es:

- INVALID: dirección desconocida
- ESTABLISHED: conexión que ha enviado paquetes en ambas direcciones
- NEW: conexión sólo ha enviado paquetes en una dirección
- RELATED: está iniciando una nueva conexión asociada a otra existente

Ejemplos:

- iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
  - Acepta todo paquete que forme parte de una conexión realizada
- iptables -A INPUT -m state -m tcp -p tcp --dport 32768:61000 --state RELATED -j ACCEPT
  - Para conexiones pasivas :-)

Objetivo REJECT

--jump REJECT --reject-with tipo

en donde tipo ha de ser:

- icmp-net-unreachable: envía un icmp de red inalcanzable
- icmp-host-unreachable: envía un icmp de host inalcanzable
- icmp-port-unreachable: envía un icmp de puerto inalcanzable
- icmp-proto-unreachable: envía un icmp de protocolo inalcanzable
- icmp-net-prohibited: envía un icmp de red prohibida
- icmp-host-prohibited: envía un icmp de host prohibido
- tcp-reset: envía un RST de tcp

OJO: tcp-reset sólo en TCP

Ejemplos:

- iptables -A INPUT -p tcp --dport 113 -j REJECT --reject-with tcp-reset

Objetivo LOG

--jump LOG --log-level nivel --log-prefix prefijo

nivel de log es alguno de /etc/syslog.conf

- crit, debug, err ...
- LOG siempre registra en syslog

y prefijo es:

- una cadena de texto que se antepone al LOG para identificarlo

Ejemplos:

- iptables -A INPUT -p tcp -dport 22 -j LOG --log-level debug --log-prefix "SSH: "

OJO: LOG no deniega ni autoriza

OJO: LOG siempre con --limit

Tipo de servicio (I)

¿Qué es?

- Un número que identifica el tipo del servicio al que queremos acceder

¿Para qué sirve?

- Para establecer una jerarquía a la hora de enviar paquetes por la red
- Páquetes más rápidos, más seguros, etc...

Valores de TOS:

- Minimize-Delay (0x10) = retardo mínimo
- Maximize-Throughput (0x08) = rendimiento máximo
- Maximize-Reliability (0x04) = fiabilidad máxima
- Minimize-Cost (0x02) = coste mínimo
- Normal-Service (0x00) = normal sin prioridad

Tipo de servicio (y II)

-t mangle --jump TOS --set-tos tos

donde:

- tos es un valor de TOS (habitualmente 0x10 o 0x08)

Ejemplos:

- iptables -t mangle -A OUTPUT -p tcp --dport 20 -j TOS --set-tos 0x08
- iptables -t mangle -A OUTPUT -p tcp --dport 21 -j TOS --set-tos 0x10

Combinando todo

Las cadenas se pueden combinar entre si, pero:

- tienen que ser coherentes con la tabla y los modulos
- no se puede hacer una regla para dos protocolos diferentes
- no se pueden poner dos objetivos por regla!
- ...

Ejemplo:

- iptables -A INPUT -p tcp --dport 22 -s ! 192.168.0.2 --tcp-flags SYN,ACK SYN -j DROP

Políticas por defecto (I)

Qué hacemos si el paquete no se adecua a ninguna regla:

- ¿lo dejamos pasar?
- ¿hacemos DROP?
- ¿rechazamos?

Dos ideas básicas:

- Permitimos todo, menos lo prohibido
- Prohibimos todo, menos lo que permitimos

-P, --policy cadena objetivo

Ejemplos:

- iptables -P INPUT DROP
- iptables -P OUTPUT ACCEPT

Ejemplo

```
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP

iptables -A INPUT -p tcp --dport ssh -j ACCEPT
iptables -A INPUT -p tcp --dport www -j ACCEPT
iptables -A INPUT -p tcp --dport ident -j ACCEPT

iptables -A INPUT -p tcp -s 69.69.69.69 -j ACCEPT
iptables -A INPUT -p tcp -s 55.55.55.55 -j ACCEPT
iptables -A INPUT -p tcp -s 44.44.44.44 -j ACCEPT
```

¡Y ya está configurada!

¡Pues no!

NO

Nuestra máquina está bloqueada
(no hemos aceptado la interfaz lo)

Guardando nuestras reglas

Problema:

- al reiniciar las reglas se pierden

iptables-save, iptables-restore

- son herramientas que permiten guardar y cargar las reglas

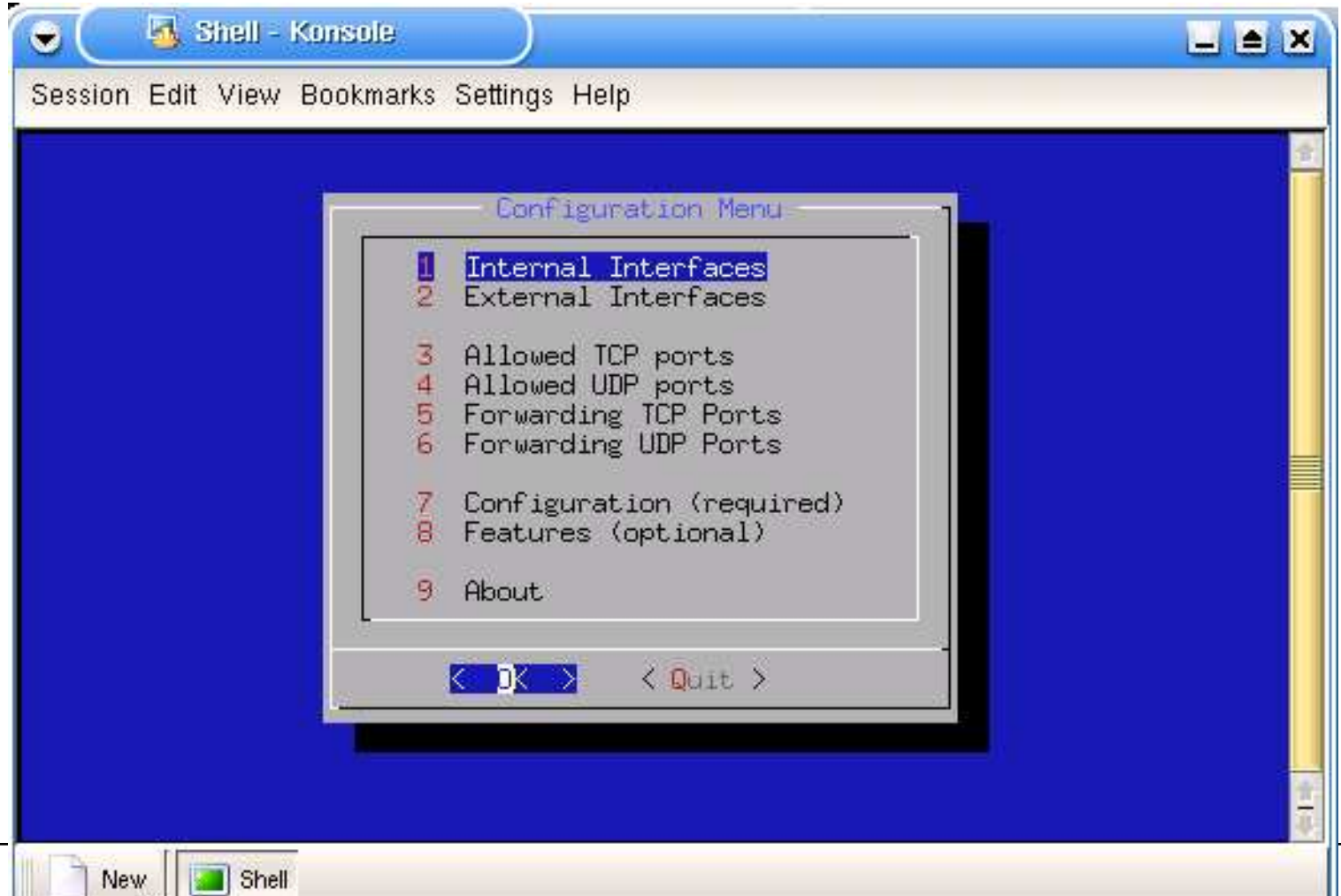
Caso de uso:

- iptables-save >misreglas.iptables
- iptables-restore <mireglas.iptables

Utilidades para perezosos

Existen diversos scripts:

- http://iptables-script.dk/ (PHP)
- http://lug.mfh-iserlohn.de/uif (Macrolenguaje)
- http://firewall-jay.sourceforge.net/ (curses)



Problemas comunes

Por despistes:

- Olvidarse de aceptar la interfaz lo
- Olvidarse de usar limit con LOG

Por servidores con mala leche:

- Aceptar puertos locales (sysctl -a | grep local_port)
- Aceptar ftp-data
- Rechazar (que no DROP) 113, 8080, 3128, 25, 12345...

Bibliografía...

Libros:

- Sonnenreich, Wes: «Building Linux and OpenBSD Firewalls», Ed. Wiley
- Ziegler, Robert L.: «Firewalls Linux», Ed. Prentice Hall
- Purdy, Gregor N.: «Linux iptables Pocket Reference», Ed. O'Reilly

...y «webliografía»

Ínternet:

- http://www.netfilter.org
- http://lists.samba.org/mailman/listinfo/netfilter (Lista de correo)
- http://www.cert.org

Localmente:

- iptables(8) ip(7) /etc/protocols /etc/services

RFCs:

- RFC 792 - Internet Control Message Protocol
- RFC 793 - Transmission Control Protocol
- RFC 1984 - Secuencias TCP seguras

¡GRACIAS!

(aplausos y preguntas)