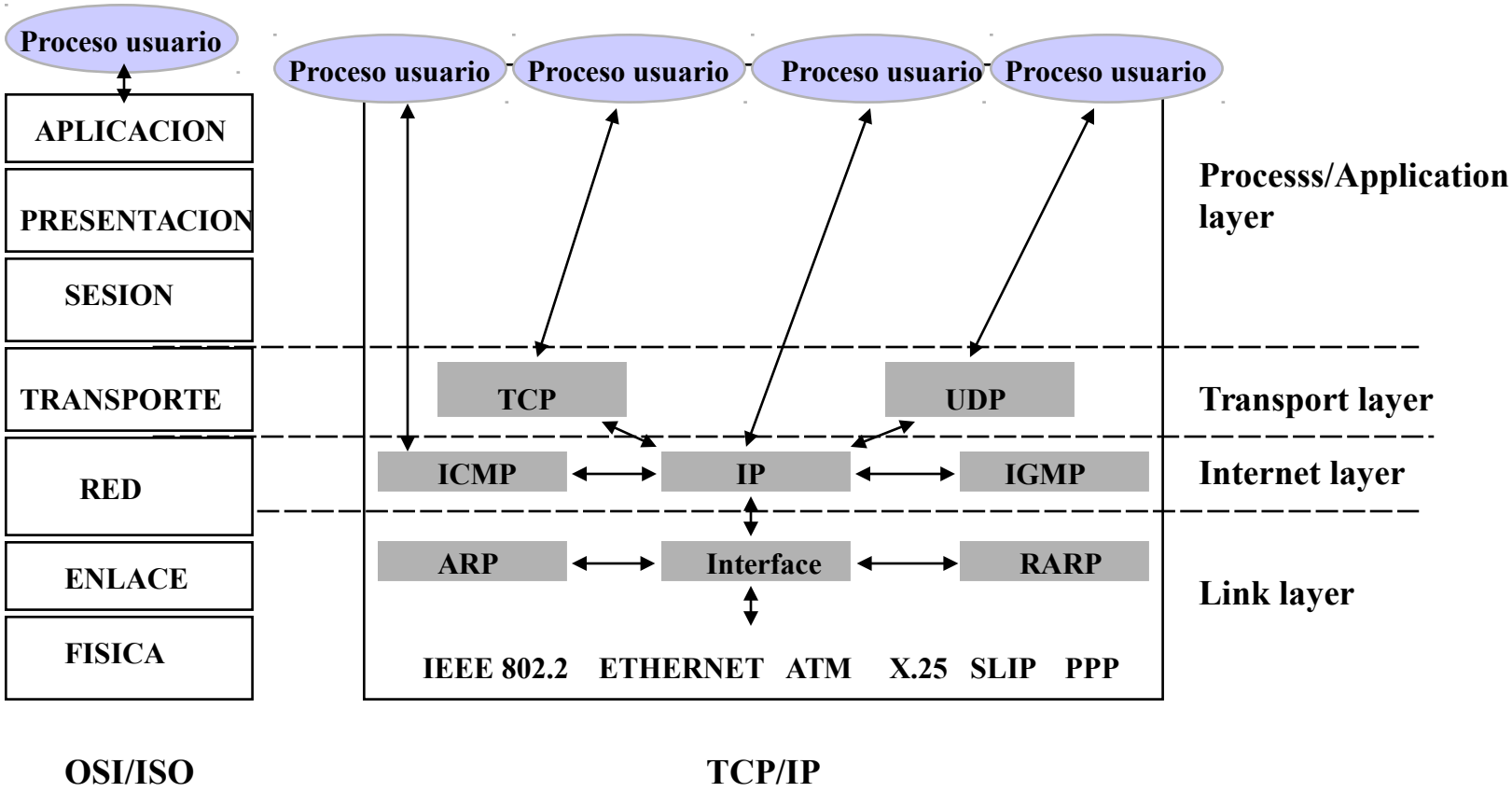
Arquitectura TCP/IP

Surge como proyecto de investigación auspiciado por elDepartamento de Defensa de los EEUA (ARPANET) (1960)
Objetivo: red que fuera capaz de sobrevivir a un ataque nuclear, aparece la técnica de packet switching
Evoluciona dando lugar a la aparición de los protocolos TCP e IP, y a la Internet



Arquitectura TCP/IP(2)

- **Nivel Aplicación**
  - **Incluye niveles Aplicación y Presentación OSI**
  - **Clases de protocolos:**
    - **Usuario (Aplicaciones) (FTP, Telnet, SMTP, específicos)**
    - **Soporte (SNMP, DNS, BOOTP, Ruteo, etc.)**
- **Nivel Transporte**
  - **Incluye niveles Transporte y parte de Sesión OSI**
    - **Punta a Punta**
    - **Protocolos**
      - **TCP (orientado a conexión, contol de flujo, secuenciamiento)**
      - **UDP (connectionless)**
- **Nivel Internet**
  - **Parte del nivel de red OSI/ISO (internetworking)**
  - **Protocolos**
    - **IP: Connectionless, addressing, TOS, fragmentación/reensamblado**
    - **ICMP: Protocolo de control, parte integral de IP, arquitecturalmente sobre IP. Reporte de errores, congestión, redirección**
    - **IGMP Establecimiento de grupos dinámicos multicast**
- **Nivel Network Access (Link layer)**
  - **Todo lo que se encuentra debajo de IP y sobre la capa física**
  - **No estandarizado en este modelo**
  - **Sólo se establece el método de transmisión de paquetes IP sobre este nivel (IP over ...)**

Nivel de Aplicación en TCP/IP

- **Protocolos de Soporte más comunes**
  - **Ruteo (IGPs, EGPs)**
  - **DNS (Domain Name System)**
  - **Configuración dinámica (Bootp, DHCP)**
  - **Network Management (SNMP)**

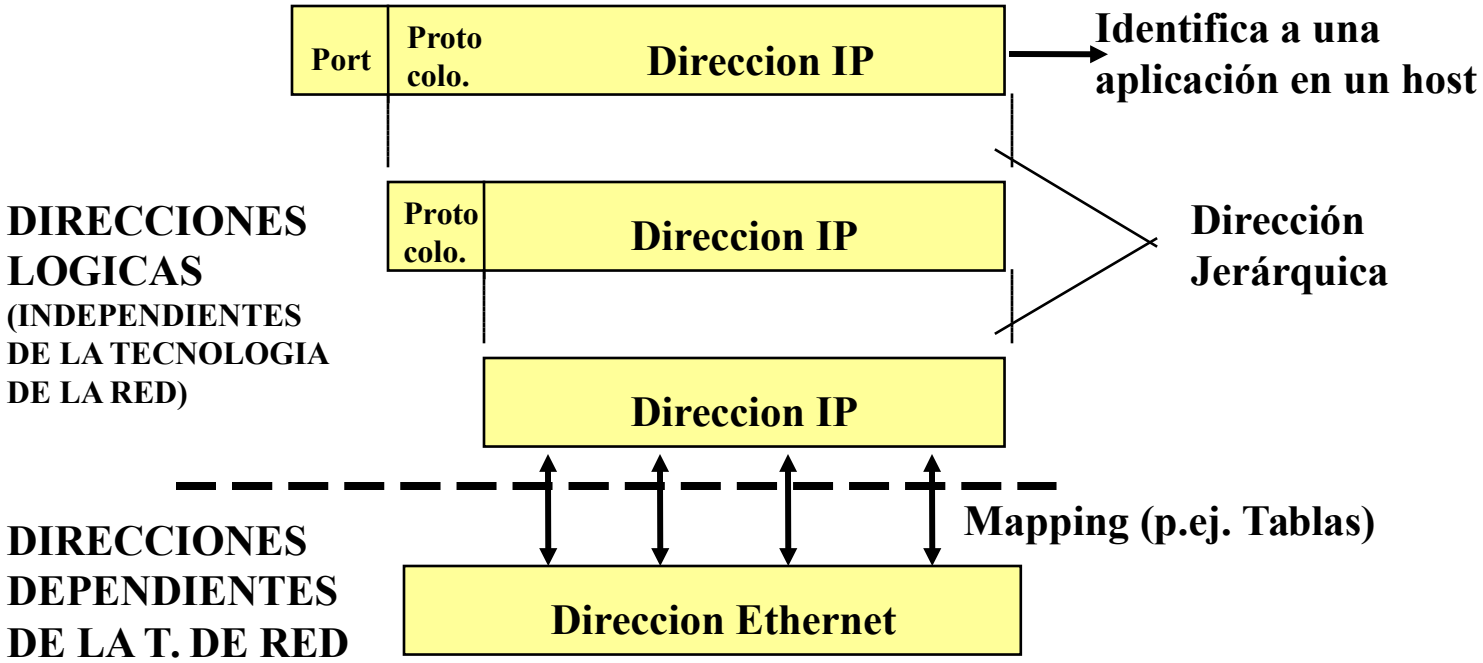
- **Aplicaciones más comunes (interacción con el usuario)**
  - **E Mail (SMTP)**
  - **Transferencia de archivos (FTP)**
  - **Login Remoto (Telnet – reemplazado por SSH)**
  - **Búsqueda de recursos (WWW, Archie, etc)**
  - **Sistemas de archivos remotos (NFS, AFS)**

Direcciones IPv4

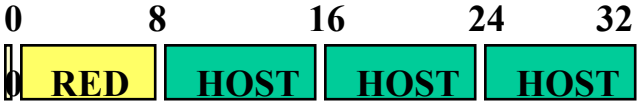
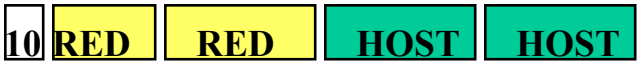
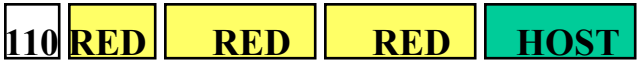
- **Identifican unívocamente un punto de acceso (interfaz) a la red. Un router o un host multi-homed tienen varias.**
- **Tienen un significado global en la Internet.**
- **Son asignadas por una autoridad central: ICCAN (Internet Corporation for Assigned Names and Numbers)**
- **Son números de 32 bits, expresados en notación decimal con puntos, byte a byte (p.ej. 123.3.45.77).**
- **Para facilidad de los usuarios, se define un mapping estático de las direcciones IP con nombres “mas legibles” para las personas (DNS - Domain Name Server).**

Relación de las direcciones IPv4 con las de otros niveles

- Una dirección IP es independiente de las direcciones físicas de subred
- Las direcciones de niveles superiores contienen a la IP



Clases de direcciones IPv4 (obsoleto)

Clase	Formato	Rango	Redes/Hosts
A		0.0.0.0 a 127.255.255.255	126/16.777.214
B		128.0.0.0 a 191.255.255.255	16.382/65.534
C		192.0.0.0 a 223.255.255.255	2.097.150/254
D		224.0.0.0 a 239.255.255.255	
E		240.0.0.0 a 247.255.255.255	

Dirección especial: loopbak (127.0.0.0):

- **Para comunicaciones de procesos en la misma máquina.**
- **Nunca es propagada a la red**

Direcciones IPv4 con significado especial

Notación: <Red, Host>

<0, 0>	**este host en esta red**	**S**	**bootp**
<0, H>	**host H en esta red**	**S**	**host parcialmente inicializado**
<R, 0>	**un host en red R**	**S**	
<R, H>	**host H en red R**	**S/D**	
<R, 1>	**Directed broadcast todos los hosts de la red**	**D**	
<1, 1>	**Limited broadcast**	**D**	**no propagada por los routers**

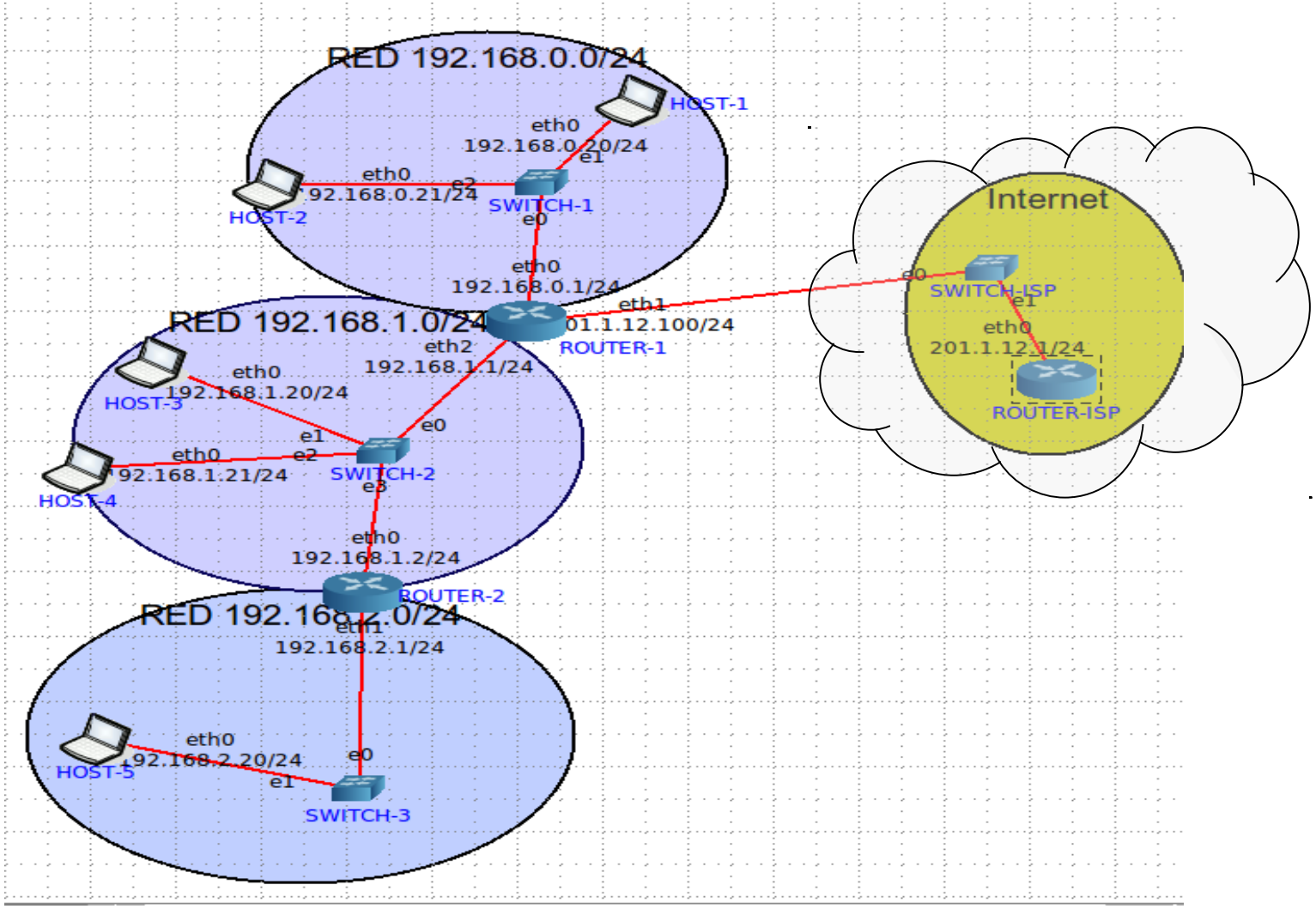
- **Significados especiales:**
  0: "este"
  1: "todos"

No pueden usarse para identificar a un host o red en particular

Direcciones privadas

- **10.0.0.0 a 10.255.255.255 (una clase A)**
- **172.16.0.0 a 172.31.255.255 (16 clases B)**
- **192.168.0.0 a 192.168.255.255 (255 clases C)**

Ejemplo de asignacion de direcciones IP



Direccionamiento IP

- **Direccionamiento jerárquico: <prefijo, host>**
  - **prefijo:**
    - **Conjunto de bits contiguos a la izquierda**
    - **utilizado por los routers para determinar paths para direcciones no locales**
  - **Host:**
    - **Los bits restantes ubicados a la derecha**
    - **utilizado para ubicar el equipo dentro de la red a la que pertenece**

- **Lomgitud del prefijo**
  - **Direcciones classful (obsoleto)**
    - **Longitud determinada por los primeros bits de la direccion (clase de dirección, A, B, etc)**
  - **Direcciones classles**
    - **Longitud del prefijo no incluida en la direccion IP**

- **Alternativas para indicar el prefijo**
  - **Classles: no es necesario indicarlo**
  - **Classful: se indica explicitamente**
    - **Notacion con /: por ejemplo /24**
    - **Mascara: por ejemplo 255.255.255.0**

Problemas del direccionamiento classful

- **Prefijos de longitud fija (clases), provoca un uso ineficiente en el espacio de direcciones.**

- **Crecimiento acelerado de la Internet, evidencia la falta de escalabilidad del esquema de direccionamiento (Agotamiento de clases B, incremento de tamaño de tablas de ruteo al utilizar direcciones de clase C).**

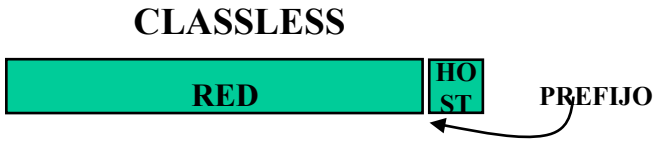
- **Soluciones**
  - **Estos problemas se solucionan a corto plazo en el contexto de Ipv4.**
    - **Prefijos variables (classles)**
    - **Asignacion racional de direcciones (CIDR)**
  - **Definitivamente solucionados en IPv6.**

Clases de direccionamiento

- **Classful Addressing**
  - **Los routers aceptan determinadas longitudes de prefijos (clases de direcciones IP y máscaras locales).**
  - **Los protocolos de ruteo no transmiten información acerca de los prefijos.**
  - **Para rutear un datagram, se busca en la tabla de rutas una dirección de red que coincida con el prefijo de la dirección de destino.**

- **Classless Addressing**
  - **Los routers aceptan longitudes de prefijo variables.**
  - **Los protocolos de ruteo transmiten información de longitud de prefijo, en forma de máscara, junto con cada dirección.**
  - **Para rutear un datagram, se utiliza el criterio de ruta más específica ("longest match" al buscar en las tablas).**

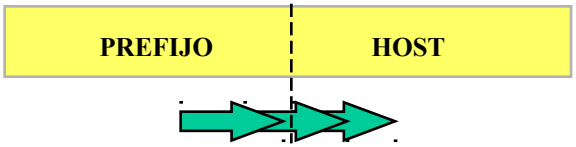
201.2.2.1
C9.02.02.01
1100 1001. …..
110 indica prefijo 24



201.2.2.1/24
Prefijo explicito

Classless Addressing

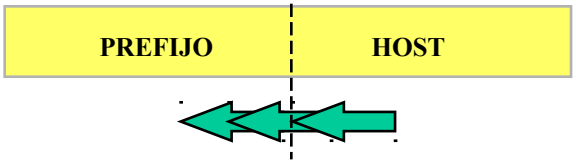
Subnetting (VLSM -Variable Length Subnet Masking-)



Extiende el prefijo hacia la derecha

Permite un mejor uso del espacio de direcciones, al soportar subredes de longitud variable que se adaptan mejor a casos particulares.

Supernetting (sumarización)



Reduce el prefijo hacia la izquierda

Permite reducir tamaño de tablas de ruteo y tráfico de intercambio de información de ruteo al posibilitar que un router anuncie y tenga una única entrada en la tabla para un conjunto de rutas.

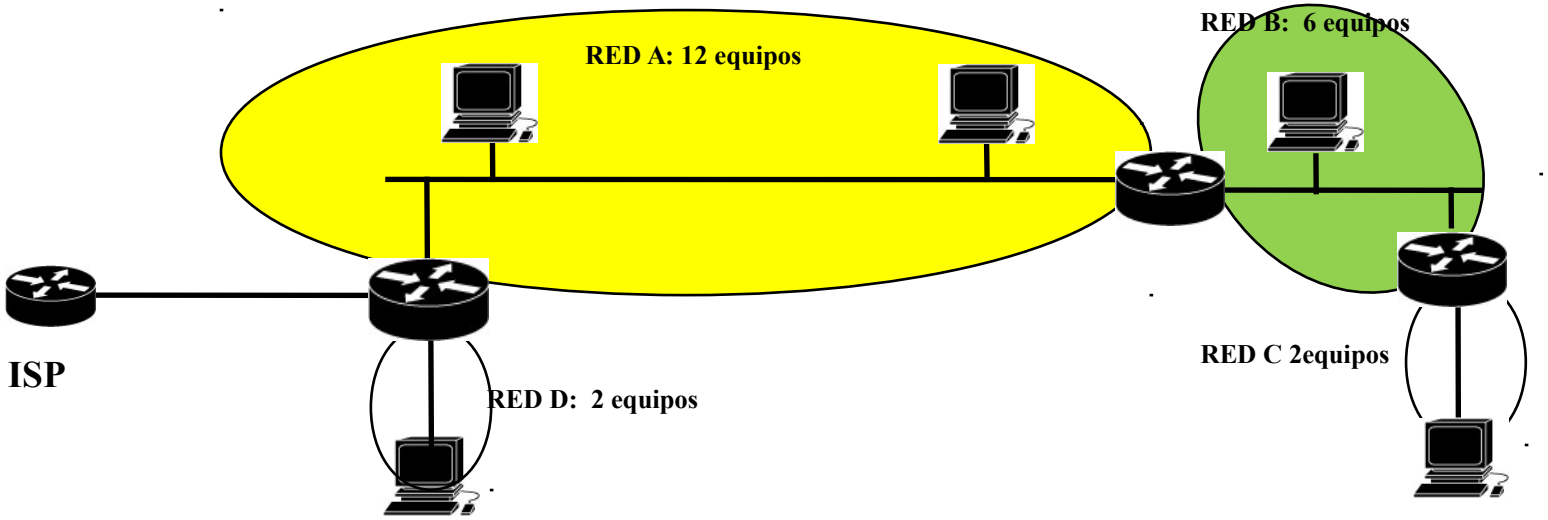
VLSM (Variable Length Subnet Mask)

- **Uso más eficiente del espacio de direcciones**
  - **Posibilidad de asignar un número de direcciones acorde al número de equipos de la subred**

- **Requerimientos**
  - **Algoritmo “longest match prefix” para reenvío**
  - **Protcolos de ruteo que intercambien máscaras de red**
- **Reglas de asignación de direcciones**
  - **El espacio de direcciones asignado bajo una máscara no puede ser asignado bajo otra máscara (prefijo más largo).**

Ejemplo de asignación de direcciones conVLSM

- **Espacio de 32 direcciones IP asignado: 201.2.3.0/27**
- **Redes a definir: 12, 5, 2 y 2 equipos cada una**



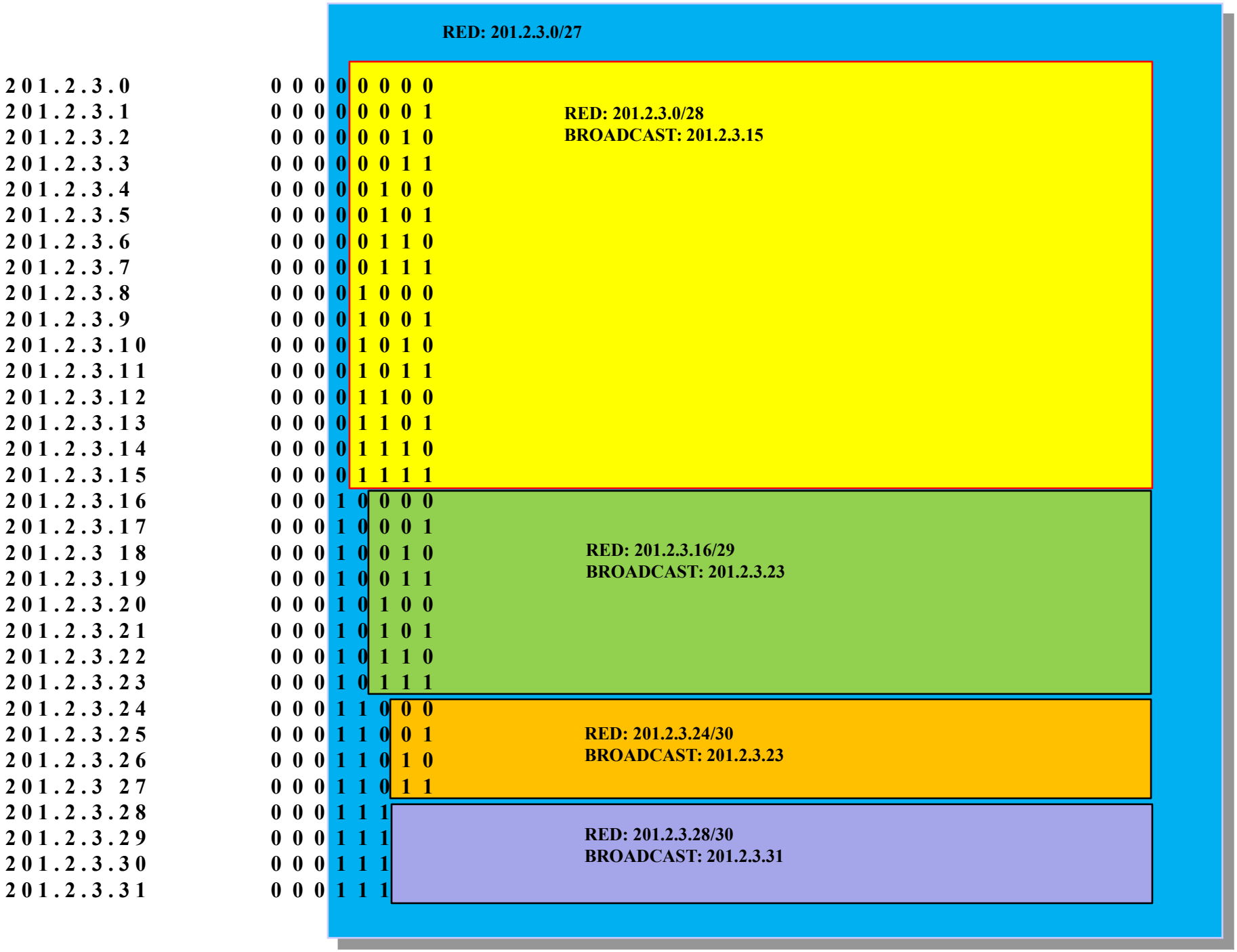
Asignacion de direcciones con VLSM

Considerar

- **No dividir en clases de direcciones:**
  en lugar de clase C 201.2.3.0
  bloque de 256 direcciones IP, a partir de la dirección 201.2.3.0
- **Se asignan bloques de 2**k direcciones, identificados por una red de prefijo (32 - k)**
- **La primera dirección es multiplo de 2**k, es la direccion de red**
- **El bloque de direcciones asignado, puede dividirse recursivamente, hasta llegar a bloques de prefijo 30**

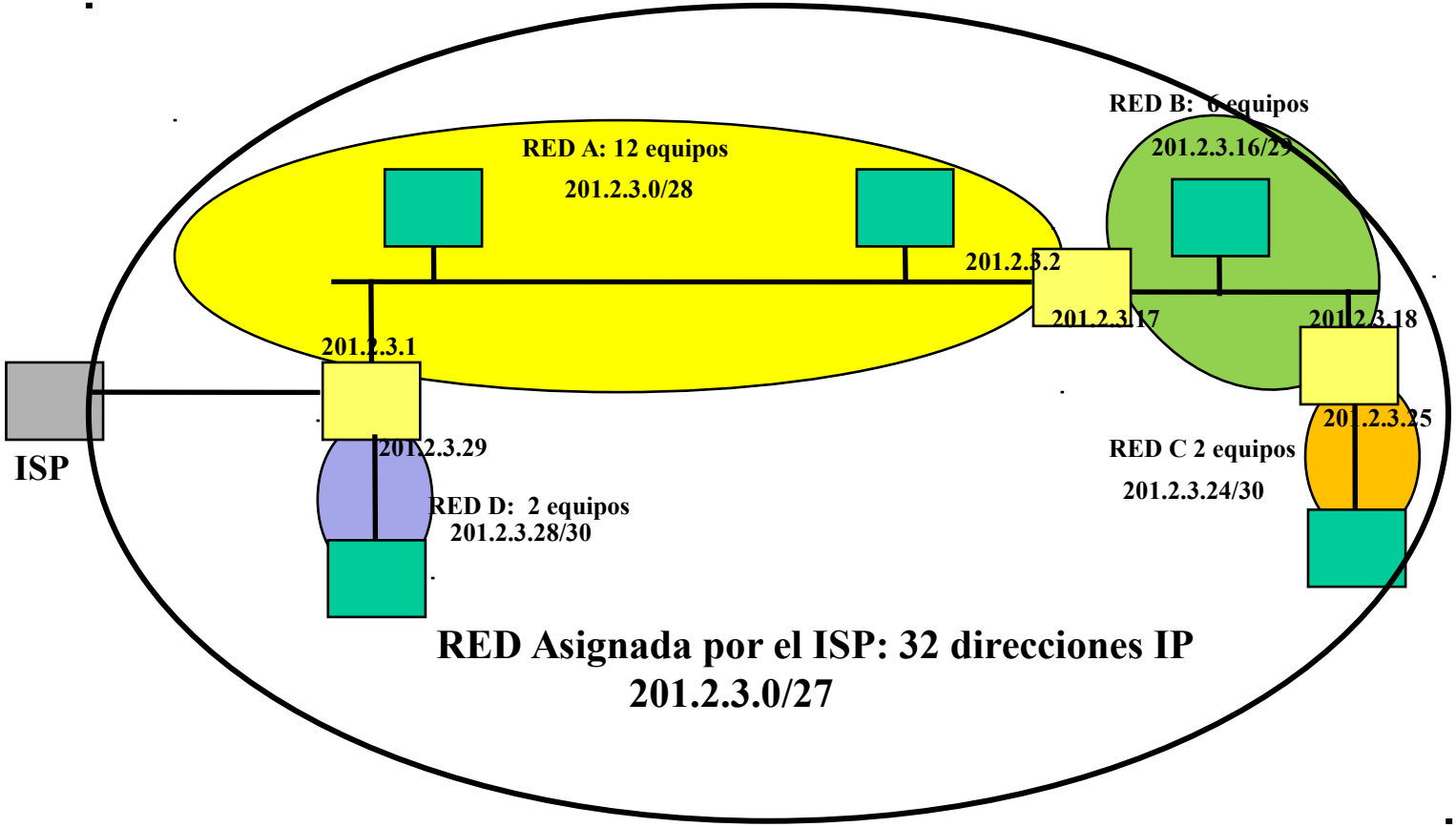
Procedimiento a seguir:

- **Ordenar las redes de mayor a menor cantidad de direcciones necesarias**
- **Desde la primer direccion del bloque, asignar una cantidad de direcciones contiguas a la primera red.Esta cantidad debe ser la minima potencia de dos tal que alcance para las direcciones que se necesitan en esa red**
- **Desde la siguiente direccion libre, realizar el mismo procedimiento para la siguiente red**
- **Tener en cuenta, si es posible, la topología de la red (de toda mi intranet)**


RED: 201.2.3.0/27
201.2.3.0 0 0 0 0 0 0 0 0
201.2.3.1 0 0 0 0 0 0 0 1
201.2.3.2 0 0 0 0 0 0 1 0
201.2.3.3 0 0 0 0 0 0 1 1
201.2.3.4 0 0 0 0 0 1 0 0
201.2.3.5 0 0 0 0 0 1 0 1
201.2.3.6 0 0 0 0 0 1 1 0
201.2.3.7 0 0 0 0 0 1 1 1
201.2.3.8 0 0 0 0 1 0 0 0
201.2.3.9 0 0 0 0 1 0 0 1
201.2.3.10 0 0 0 0 1 0 1 0
201.2.3.11 0 0 0 0 1 0 1 1
201.2.3.12 0 0 0 0 1 1 0 0
201.2.3.13 0 0 0 0 1 1 0 1
201.2.3.14 0 0 0 0 1 1 1 0
201.2.3.15 0 0 0 0 1 1 1 1
RED: 201.2.3.0/28
BROADCAST: 201.2.3.15
201.2.3.16 0 0 0 1 0 0 0 0
201.2.3.17 0 0 0 1 0 0 0 1
201.2.3 18 0 0 0 1 0 0 1 0
201.2.3.19 0 0 0 1 0 0 1 1
201.2.3.20 0 0 0 1 0 1 0 0
201.2.3.21 0 0 0 1 0 1 0 1
201.2.3.22 0 0 0 1 0 1 1 0
201.2.3.23 0 0 0 1 0 1 1 1
RED: 201.2.3.16/29
BROADCAST: 201.2.3.23
201.2.3.24 0 0 0 1 1 0 0 0
201.2.3.25 0 0 0 1 1 0 0 1
201.2.3.26 0 0 0 1 1 0 1 0
201.2.3 27 0 0 0 1 1 0 1 1
RED: 201.2.3.24/30
BROADCAST: 201.2.3.23
201.2.3.28 0 0 0 1 1 1
201.2.3.29 0 0 0 1 1 1
201.2.3.30 0 0 0 1 1 1
201.2.3.31 0 0 0 1 1 1
RED: 201.2.3.28/30
BROADCAST: 201.2.3.31

Ejemplo de asignación de direcciones conVLSM

- **Espacio de 32 direcciones IP asignado: 201.2.3.0/27**
- **Redes a definir: 12, 5, 2 y 2 equipos cada una**



CIDR (Classless Inter Domain Routing)

- **Crecimiento no previsto de la Internet**
- **Agotamiento de las direcciones clase B (sólo hay 16382)**
- **A muchas organizaciones no les basta con una dirección clase C (254 hosts)**
- **Solución a largo plazo (2005): IPv6**
- **Solución a corto plazo: Asignación de grupos de direcciones clase C a los usuarios**
  - **Problemas**
    - **Crecimiento inmanejable de tablas de ruteo (memoria y proceso)**
    - **Consumo excesivo de vínculos de transmisión debido a la propagción de información de ruteo**
  - **Solución a corto plazo: CIDR, que permite la asignación “eficiente” de las direccionesde red clase C restantes**

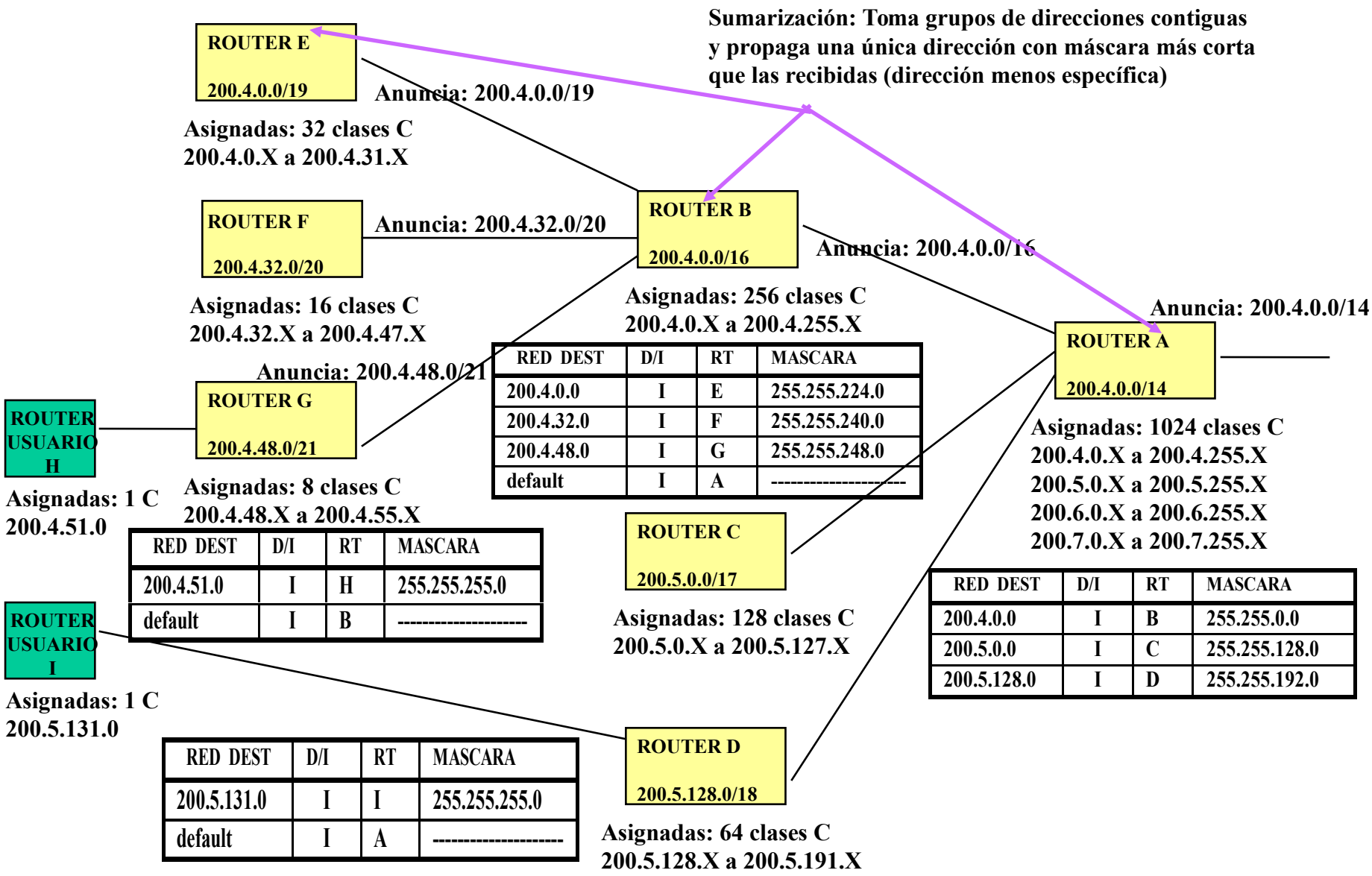
CIDR

- **CIDR (RFC 1519, Nov 1992) propone:**
  - **Asignación jerárquica de grupos de direcciones de clase C**
  - **Direcciones classless: la división entre la parte de la dirección que corresponde a la red y al host es variable, indicada por una máscara (p.e. 200.2.2.2/24)**
  - **Los routers pueden "resumir" información respecto de un grupo de direcciones y propagar la información resumida (aggregation)**
  - **En las tablas de ruteo, se almacena la información resumida**
  - **Los protocolos de ruteo más nuevos lo soportan (BGP-4, OSPF, etc)**
  - **Los routers soportan el mecanismo de matching más específico (longest match) ya que es el utlizado en subnetting**

Asignación propuesta para las direcciones clase C

Direcciones 194.0.0.0 a 195.255.255.255 Europa
Direcciones 198.0.0.0 a 199.255.255.255 América del Norte
Direcciones 200.0.0.0 a 201.255.255.255 América Central y América del Sur
Direcciones 202.0.0.0 a 203.255.255.255 Asia y el Pacífico

CIDR



RED DEST	D/I	RT	MASCARA
200.4.0.0	I	E	255.255.224.0
200.4.32.0	I	F	255.255.240.0
200.4.48.0	I	G	255.255.248.0
default	I	A	----------

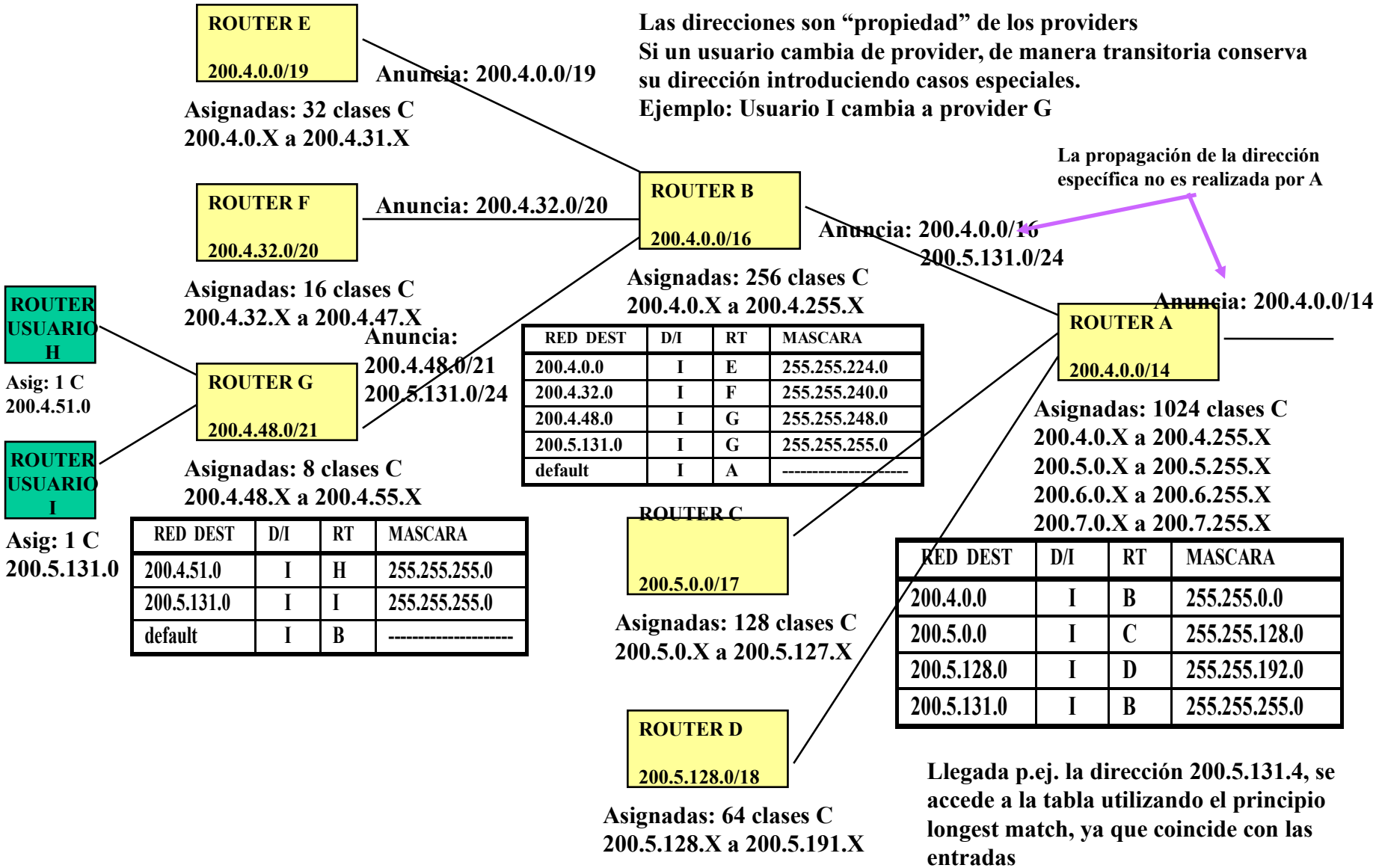
RED DEST	D/I	RT	MASCARA
200.4.51.0	I	H	255.255.255.0
default	I	B	----------

RED DEST	D/I	RT	MASCARA
200.4.0.0	I	B	255.255.0.0
200.5.0.0	I	C	255.255.128.0
200.5.128.0	I	D	255.255.192.0

RED DEST	D/I	RT	MASCARA
200.5.131.0	I	I	255.255.255.0
default	I	A	----------

* Las entradas en las tablas de ruteo se muestran parcialmente

CIDR



RED DEST	D/I	RT	MASCARA
200.4.0.0	I	E	255.255.224.0
200.4.32.0	I	F	255.255.240.0
200.4.48.0	I	G	255.255.248.0
200.5.131.0	I	G	255.255.255.0
default	I	A	----------

RED DEST	D/I	RT	MASCARA
200.4.51.0	I	H	255.255.255.0
200.5.131.0	I	I	255.255.255.0
default	I	B	----------

RED DEST	D/I	RT	MASCARA
200.4.0.0	I	B	255.255.0.0
200.5.0.0	I	C	255.255.128.0
200.5.128.0	I	D	255.255.192.0
200.5.131.0	I	B	255.255.255.0

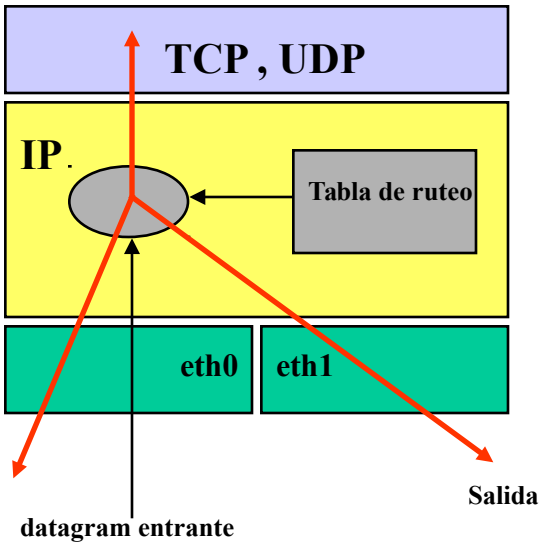
Las direcciones son “propiedad” de los providers
Si un usuario cambia de provider, de manera transitoria conserva su dirección introduciendo casos especiales.
Ejemplo: Usuario I cambia a provider G

Llegada p.ej. la dirección 200.5.131.4, se accede a la tabla utilizando el principio longest match, ya que coincide con las entradas
200.5.128.0 máscara 255.255.192.0 y
200.5.131.0 máscara 255.255.255.0

* Las entradas en las tablas de ruteo se muestran parcialmente

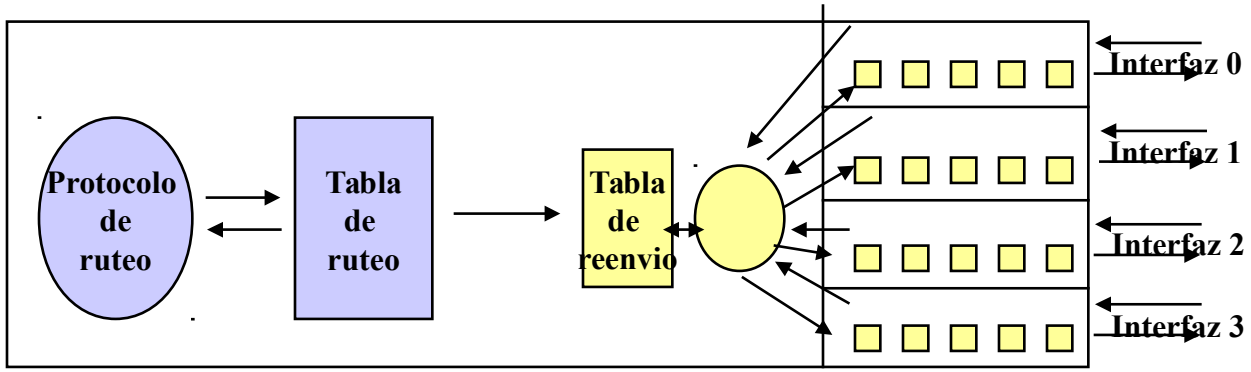
Reenvío de datagrams

- Función correspondiente al nivel IP
- Para un datagram (originado en el equipo o entrante) debe decidirse, en base a su dirección de destino, hacia qué equipo enviarlo
- La decisión se toma en base a tablas de ruteo
- Las tablas pueden ser estáticas o dinámicas (si se utiliza un protocolo de ruteo)
- Un equipo que sólo funcione como host no reenvía datagrams



Relación ruteo/reenvío

- **Ruteo**
  - **Envío de un paquete a su destino de la manera más eficiente posible**
  - **Requiere un conocimiento global de la topología de la red**
  - **Se adquiere a través de un protocolo de ruteo**
- **Reenvío**
  - **Proceso local en un router**
  - **Cada paquete, se debe enviar lo más rápidamente al siguiente router camino al destino**
  - **El reenvío se realiza en base a información provista por el componente de ruteo**



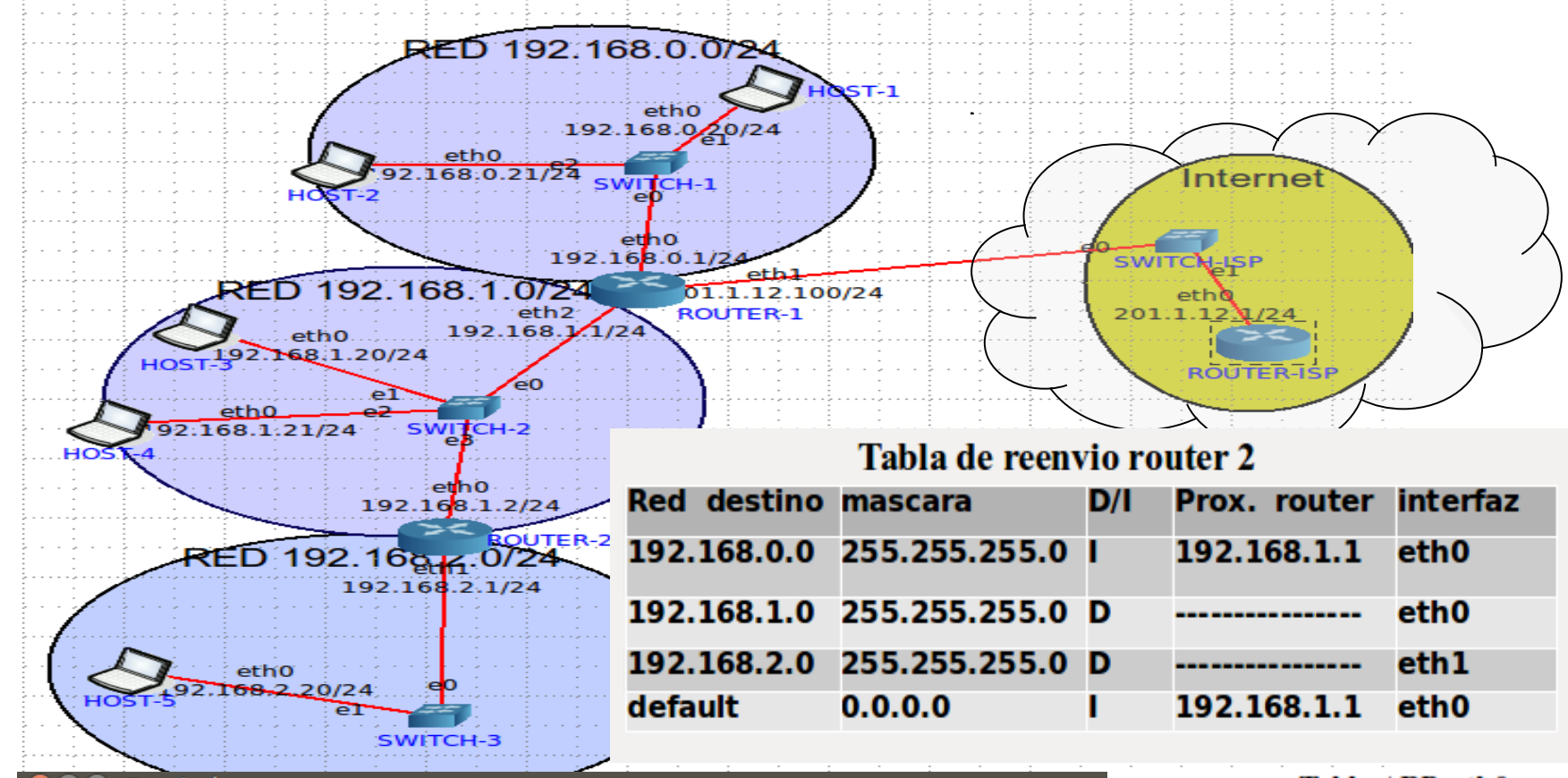


Tabla de reenvio router 2

Red destino	mascara	D/I	Prox. router	interfaz
192.168.0.0	255.255.255.0	I	192.168.1.1	eth0
192.168.1.0	255.255.255.0	D	----------	eth0
192.168.2.0	255.255.255.0	D	----------	eth1
default	0.0.0.0	I	192.168.1.1	eth0

```
Terminal
File  Edit  View  Search  Terminal  Help
root@n2:/tmp/pycore.35773/n2.conf#
root@n2:/tmp/pycore.35773/n2.conf# arp -i eth0
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.21             ether   00:00:00:aa:00:01   C                     eth0
192.168.1.20             ether   00:00:00:aa:00:00   C                     eth0
192.168.1.1              ether   00:00:00:aa:00:03   C                     eth0
root@n2:/tmp/pycore.35773/n2.conf# arp -i eth1
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.2.20             ether   00:00:00:aa:00:05   C                     eth1
root@n2:/tmp/pycore.35773/n2.conf# ip route
default via 192.168.1.1 dev eth0
192.168.0.0/24 via 192.168.1.1 dev eth0
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.2
192.168.2.0/24 dev eth1  proto kernel  scope link  src 192.168.2.1
root@n2:/tmp/pycore.35773/n2.conf#
root@n2:/tmp/pycore.35773/n2.conf#
```

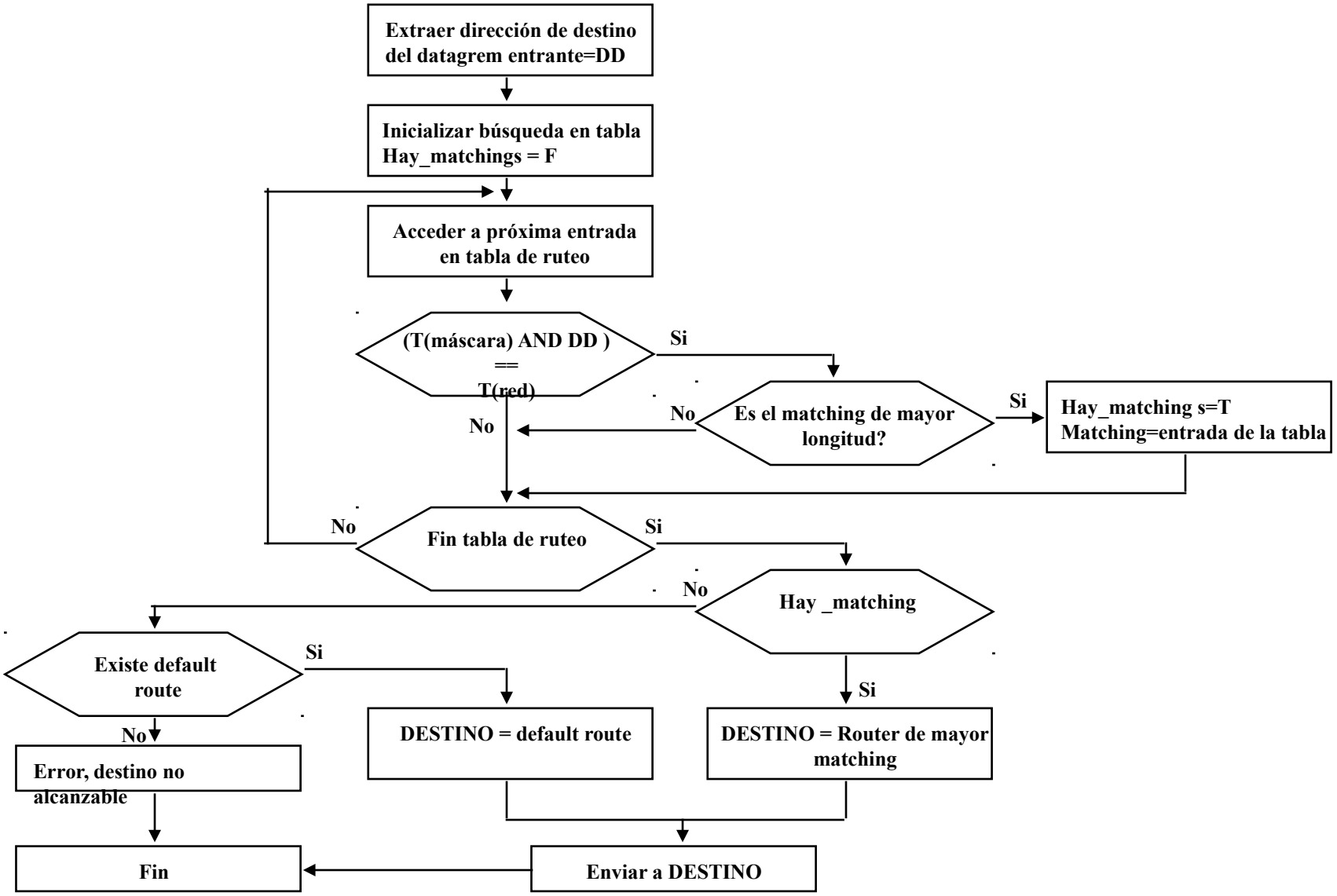
Tabla ARP eth0

Direccion IP	Dir. ethernet
192.168.1.1	00:00;00:00:00::aa
192.168.1.20	00:00:00:00:00::55
192.168.1.21	00:00:00:00:00::66

Tabla ARP eth1

Direccion IP	Dir. ethernet
192.168.2.20	00:00;00:00:00::99

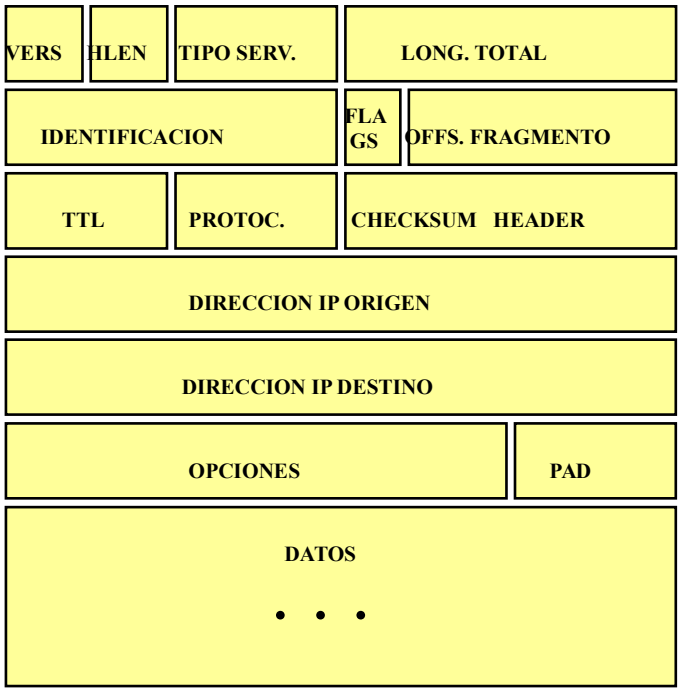
Algoritmo de búsqueda en tablas de ruteo con principio longest match prefix



IP

- **Definido en RFC 760, Ene 1980**
- **Protocolo de nivel de red (interred)**
- **No orientado a la conexión**
- **No confiable**
- **Best effort**
- **Provee una vista homogénea de la diversidad de subredes a los niveles superiores (TCP, UDP)**
- **Versiones en uso**
  - **IPv4**
  - **IPv6, que mejora a v4 :**
    - **menor overhead de proceso en routers**
    - **cantidad de direcciones y ruteo**
    - **flujos (flows) (QOS)**
    - **Seguridad**
    - **Autoconfiguracion**

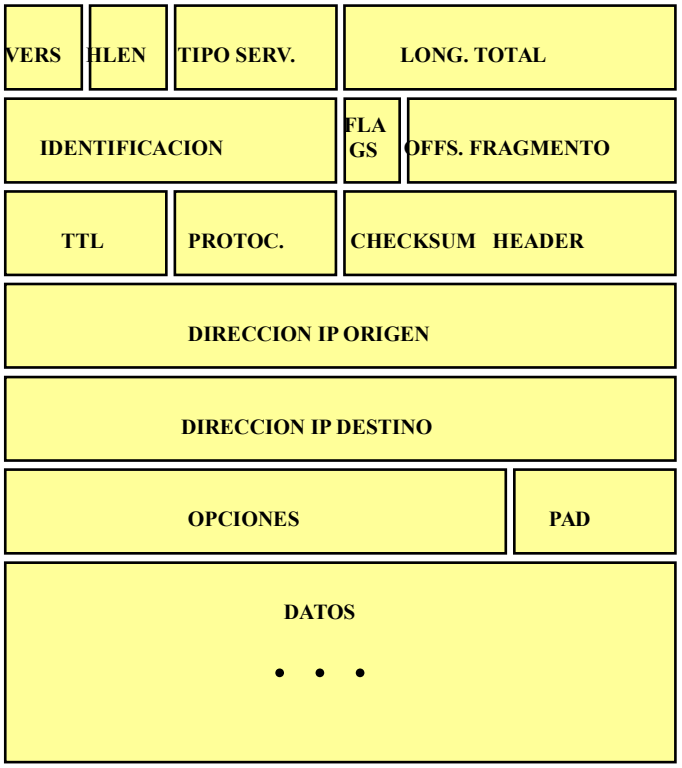
Formato de frame IP



Encabezamiento
- **longitud variable 20 a 60 bytes**
- **procesado en los routers**
- **opciones**
- **byte network order**

- **Datos**
- **procesado por niveles superiores**
- **lingitud variable**

Formato de frame IP

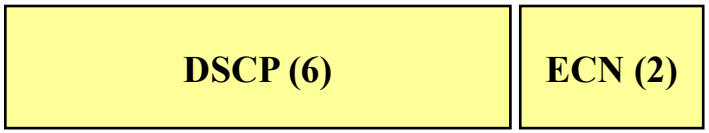


VERS(4): Versión del protocolo. Corriente: 4
HLEN(4): Longitud del header en grupos de 32 bits. Mínimo 5 (40 bytes) (sin opciones)
TIPO SERV(8): Indica el tratamiento que debe darse al datagram en los routers
LONG. TOTAL(16): Longitud total del datagram en bytes
IDENTIFICACION(16): Identificación única para el datagram (en el caso de fragmentos, todos llevan igual identificación)
FLAGS(3):Un bit no usado, Indicación de último fragmento e indicación de posibilidad de fragmantar el datagram
OFFS. FRAGMENTO(13): Indica la posición que ocupa un fragmento dentro del datagram
TTL(8): Time to live, decrementado en uno o en el tiempo de demora en segundos en cada router
CHEKSUM HEADER(16): Sólo para el header.
Inicialmente en 0. Grupos de 16 bits
Sumados en complemento a 1
Se toma el complemento a 1 del resultado
PROTOCOLO(8): Indica protocolo encapsyulado en el datagram (TCP, UDP, etc)
DIRECCIONES(32): No se cambian en los rotures
Destino: para ruteo del datagram
Origen: para mensajes de control (ICMP)
OPCIONES: Número variable con formato especial.
PAD: Completa campo de opciones a múltiplo de 32 bits.

Tipo de servicio

- **DSCP (Differentiated Services Code Point)**
  - **Valor configurado por emisor de acuerdo a la QoS requerida**
  - **Indica el tratamiento que recibirá el paquete en cada router**

- **ECN (Explicit Congestion Notification)**
  - **Utilizado para control de congestión**
  - **01 y 10: Paquete usa ECN**
  - **00 Paquete no usa ECN**
  - **11 Router indica congestión**



Redes con distintas longitudes maximas de paquetes (1)

- **Diferente longitud de paquetes en cada subred, determinada por**
  - **Carácterísticas del soporte de transmisión (TDM, Ethernet)**
  - **Tasa de errores (paquetes menores para disminuir paquetes en error)**
  - **Protocolos (Long máxima de frame)**
  - **Capacidades (buffers) en los sistemas operativos**
  - **Demora de tránsito de los paquetes en la red**
  - **Distribución equitativa del uso de la capacidad de transmisión**
  - **Límites impuestos por razones de compatibilidad**

Redes con distintas longitudes maximas de paquetes (2)

- **Alternativas**
  - **Uso de la menor longitud de paquete permitida por las subredes (descubrimiento de MTU -Maximun transmission unit-))**
    - **El origen debe conocer las longitudes máximas de cada subred que se utilizará para llegar al destino (ICMP)**
    - **No sobrecarga a los routers**
    - **Muchos routers restringen el paso de ICMP**
    - **El camino seguido por los paquetes puede cambiar**
  - **Implementación de fragmentación y reensamblado**
    - **El host origen, o un router, al recibir un paquete demasiado largo para la subred de salida, lo divide en varios paquetes de menor longitud (fragmentación)**
    - **Alternativas de reconstrucción del paquete original**
      - **Router de salida de la subred (fragmentación intranet o transparente)**
      - **Host de destino (fragmentación internet o no transparente)**

Descubrimiento de MTU

MTU (Maximun Transfer Unit): cantidad máxima de bytes que puede llevar como payload un protocolo

MTU de una interfaz de red: MTU del vinculo asociado a la interfaz

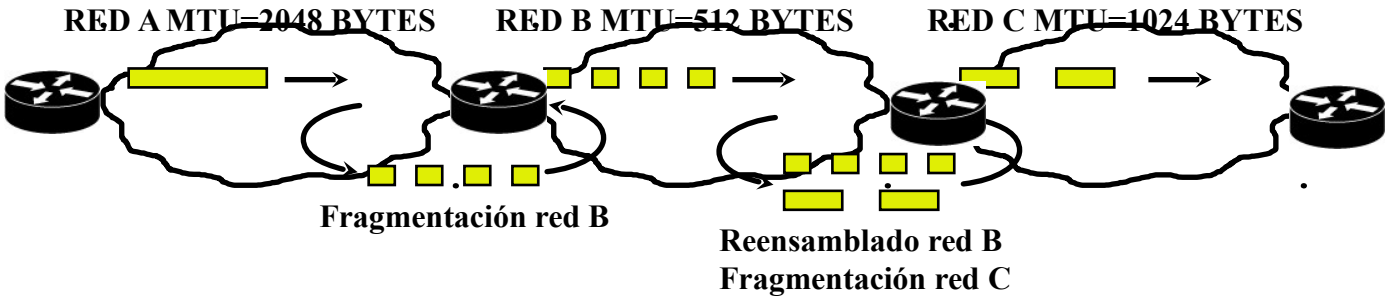
MTU de un camino en la red: el mínimo de los NTU de los vinculos que compònen el camino

En niveles inferiores, puede depender del hardware (p.ej. Ethernet 1500 bytes)

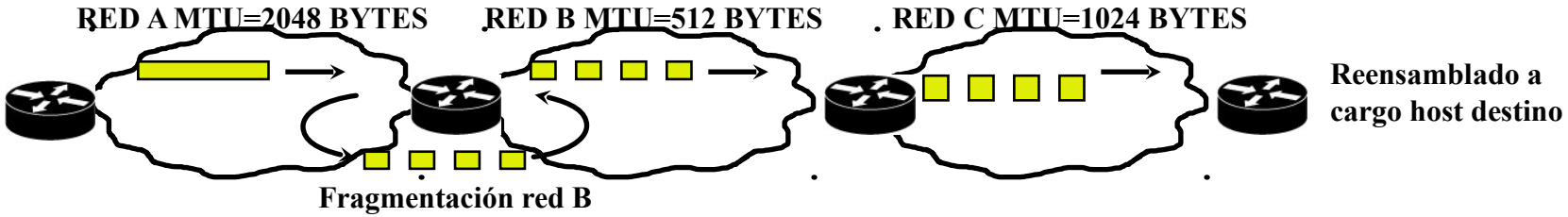
IP debe funcionar sobre diferentes tipos de hardware, soporta datagrams de hasta 65536 bytes, si no se desea que IP fragmente, debe hacerse descubrimiento de MTU (del camino origen-destino) y no entregar a IP payloads mayores

Tipos de fragmentación

Fragmentación intranet



Fragmentación internet



Tipos de fragmentación

- **Fragmentación Intranet (transparente)**
  - **El router de salida de la subred debe ser el mismo para todos los fragmentos de un paquete**
  - **Overhead debido a la posible repetición del proceso de fragmentación y reensamblado del paquete en las distintas subredes**
  - **Recursos en los routers para el proceso de reensamblado (buffers, timers)**
  - **Demoras en el tiempo de llegada de un paquete a destino debido a que en cada router debe reensamblarse**

- **Fragmentación Internet (no transparente)**
  - **Menor overhead respecto del proceso a que es sometido el paquete**
  - **Posible sub uso de las subredes (pequenos fragmentos de un paquete más largo transportados por una red con un MTU apropiado al paquete original)**
  - **Overhead en las líneas debido a que cada fragmento es considerado un paquete independiente**
  - **Cada host debe ser capaz de llevar acabo el proceso de reensamblado**

Fragmentación

Identificación de los fragmentos

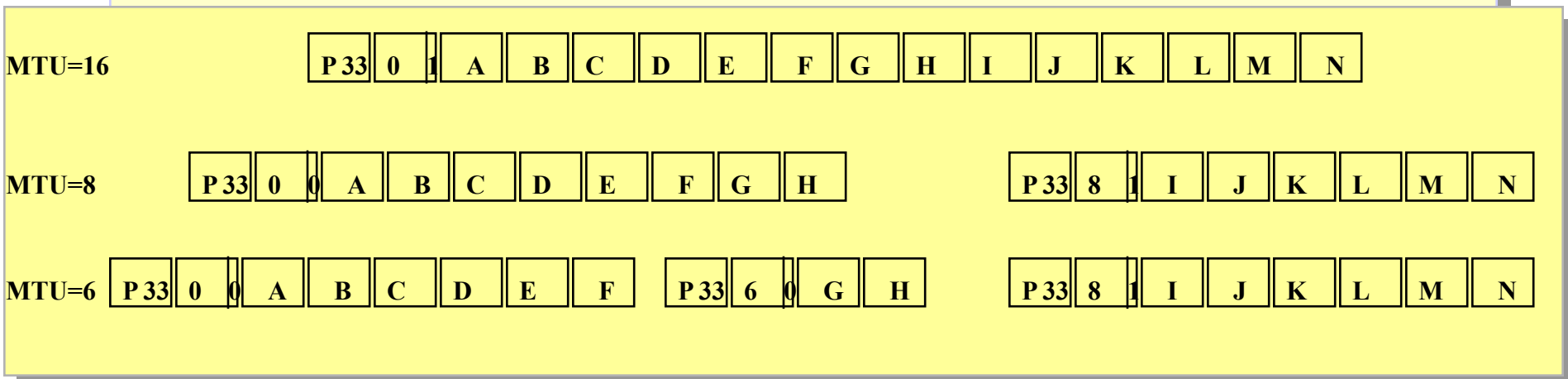
Debe soportar fragmentaciones sucesivas

Debe soportar repetición de fragmentos (posiblemente diferentes) del mismo paquete cuando se producen errores y se retransmite punta a punta

Esquema de numeración de los fragmentos

Definición de una unidad compatible (transportable) por todas las subredes, por ejemplo, byte

En cada fragmento, se índica la identificación del paquete y el número (dentro del paquete original) de la primera unidad elemental contenida en el fragmento



Fragmentación en IPv4 e IPv6

- **IPv4**
  - **Fragmentación en el origen**
  - **Fragmentacion en routers intermedios**

  Reensamblado en el host destino
- **IPv6**
  - **Fragmerntación en el origen**
  - **Los routers intermedios no fragmentan**
  - **Reensamblado en el destino**

Fragmentación IPv4:

En cada fragmento se copia header, cambian campos Flag, Long total y fragment offset

Se copian o no las opciones, dependiendo del bit de copia

Reensamblado:

Se reconoce los fragmentos por el campo identificación

Se reconoce el fin del datagram por el bit de flag de último fragmento

Se rearma el dg en base a los fragment offsets

Timer para reensamblado

Fragmentación IP

- **MTU (Maximum transmission Unit): Diferente para cada subred**
- **Fragmentación: en un router tal que MTU de salida < long. dg**
- **Puede refragmentarse un fragmento**
- **Reensamblado en el destino**

- **Fragmentación:**
  - **En cada fragmento se copia header, cambian campos Flag, Long total y fragment offset**
  - **Se copian o no las opciones, dependiendo del bit de copia**

- **Reensamblado:**
  - **Se reconoce los fragmentos por el campo identificación**
  - **Se reconoce el fin del datagram por el bit de flag de último fragmento**
  - **Se rearma el dg en base a los fragment offsets**
  - **Timer para reensamblado**

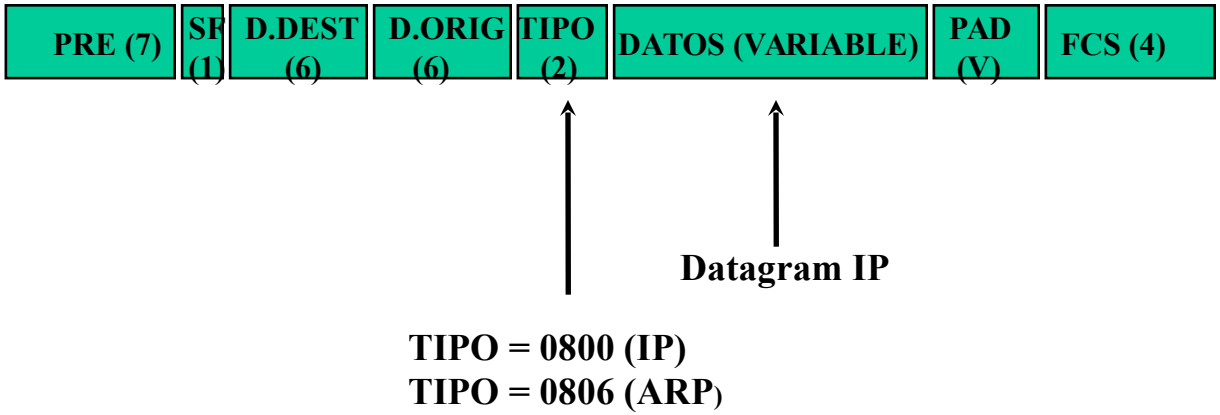
Relación de IP con la subred

- **Tratados en RFCs, por ejemplo “IP over Ethernet”**

- **Aspectos mas importantes a considerar:**
  - **Encapsulación de datagrams IP en frames de la subred**
    - **Por ejemplo, IP en Ethernet**
  - **Resolución de direcciones (mapeo de direcciones IP en direcciones de subred)**
    - **procedimiento que asocia una dirección IP a una dirección de subred**
    - **Deben ser métodos escalables (las subredes pueden crecer)**
    - **Deben presentar flexibilidad (las asociaciones pueden cambiar)**
  - **Mapeo de direcciones Multicast y Broadcast en la subred**
    - **Por ejemplo 255.255.255.255 IP a FF:FF:FF:FF:FF:FF**

IP sobre Ethernet

- **Definida en RFC 894 (Apr 1984)**
- **Tipo de protocolo: 800 hex IP; 806 hex ARP**
- **Se utiliza relleno para lograr la longitud mínima del frame ethernet**
- **Propone la resolución de direcciones a través de tablas estáticas o procedimiento de descubrimiento (ARP)**
- **Mapeo de direcciones broadcast limitado de IP en direcciones broadcast Ethernet**



ARP: Address Resolution Protocol

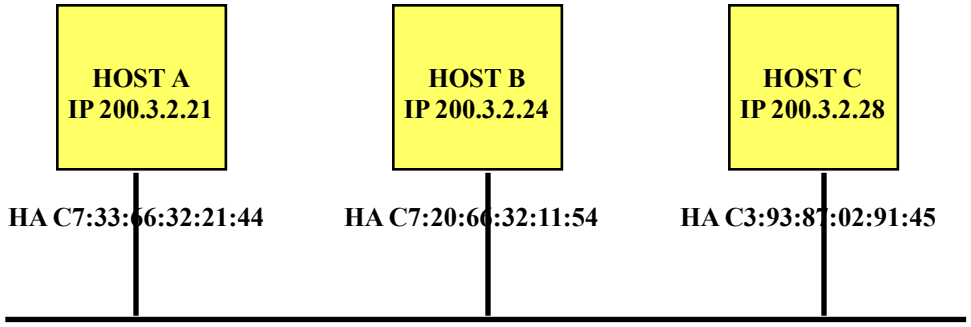
- **Surge como necesidad en las primeras redes Ethernet (broadcast múltiple acceso) (RFC 826, Nov. 1982)**

- **Una maquina conoce:**
  - **Su dirección IP (por ejemplo almacenada en disco)**
  - **Su dirección de hardware (grabada en la placa Ethernet)**
  - **La dirección IP del equipo a quien desea enviar un datagram**

  - **Para poder enviar el datagram, debe encapsularlo en un frame Ethernet, es decir, necesita conocer la direccion de hardware del equipo destino**
  - **Lo resuelve a través del protocolo ARP**

- **Características del medioambiente para ARP:**
  - **Direcciones de subred en hardware, de mayor longitud que las direcciones IP**
  - **Redes dinámicas, los equipos se conectan y desconectan, y cambian sus direcciones (placas) de subred**

ARP: Address Resolution Protocol

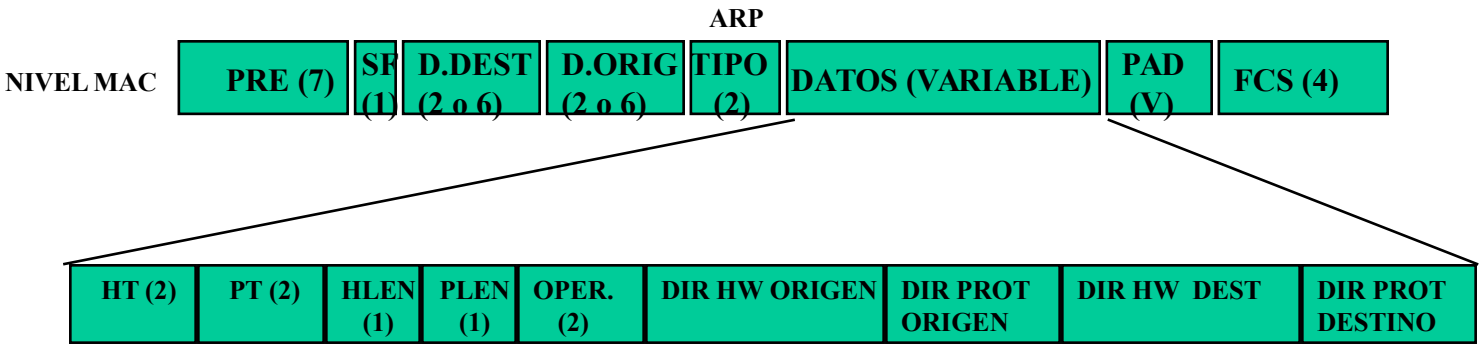


Operación del ARP

Host A (200.3.2.21) desea enviar datagram a Host B (200.3.2.24)
No conoce la dirección Ethernet de B

- **A (su ARP) envía un frame Ethernet, broadcast, con un frame ARP encapsulado preguntando: Que dirección Ethernet tiene el equipo cuya dir IP es 200.3.2.24 (ARP REQUEST)**
- **B (el equipo que reconoce su dir IP) responde(su ARP) enviando su dir Ethernet (ARP REPLY)**
- **A coloca el mapping 200.3.2.24 / C7:20:66:32:11:54 en su tabla ARP**
- **A está en condiciones de enviar el dg a B**

ARP: Formato de frame



HT: Tipo de hardware de la subred (Ethernet = 1)
PT: Tipo de protocolo (IP = 0800 hex)
HLEN: Longitud de la dirección de subred
PLEN: Longitud de la dirección del protocolo
OPER: Tipo de PDU
- **ARP REQUEST**
- **ARP RESPONSE**
- **RARP REQUEST**
- **RARP RESPONSE**

DIR ORIGEN: Las del equipo que origina el request
DIR DESTINO: Las del equipo que contesta c/reply

El reply será enviado a la dirección Ethernet origen que va en el ARP, no a la dirección origen MAC

Frames request y replay intercambiados cuando equipo 1 hace un request a equipo 2

Formato:
<MAC dest, MAC orig,oper,H orig, P orig, H dest, P dest>

Request:
<M Broad, M1, request, H1, IP 1, ??, IP 2>

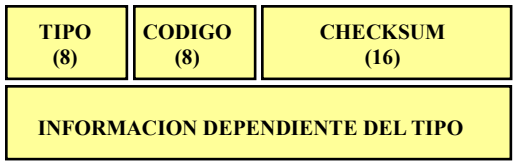
Reply
<H1, H2, reply, H1, IP1, H2, IP2>

Caracteristicas de ARP

- **Los ARP en cada equipo almacenan una tabla dinámica con los mappings**
- **Las entradas se eliminan luego de un cierto tiempo**
- **Se utilizan los ARP request (broadcast) para generar entradas en las tablas para las direcciones origen**
- **Gratuitous ARP (ARP preguntando por su propia dirección)**
  - **Permite determinar direcciones IP duplicadas**
  - **Permite que los hosts actualicen sus tablas ARP**
- **Control de envío de ARP requests**

ICMP (Internet Control Message protocol)

- **Internet Control Message Protocol (RFC 792, Sep 1981)**
- **Objeto: Nivel IP de un router o host informa a nivel IP del origen de un datagram acerca de problemas**
- **Encapsulado en IP (Protocolo = 1), parte necesaria de toda implementación IP**
- **No se generan paquetes ICMP sobre**
  - **condiciones de error producidas por ICMP**
  - **datagrams multicast o broadcast**
  - **fragmentos de datagrams IP, excepto el primero**
- **Formato general**

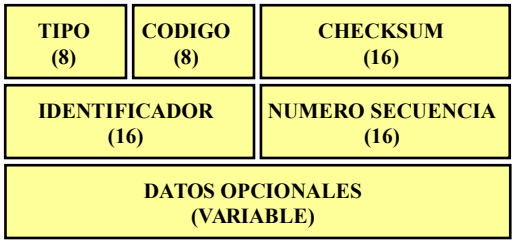


Tipo: Identifica el tipo de mensaje
Codig": Información adicional, dependiendo del tipo
Checksum: sólo para ICMP, calculado como en IP

TIPOS
0: Echo reply
3: Destination Unreachable
4: Source Quench
5: Redirect
8: Echo Request
11:Time Exceeded
12:Parameter Problem
13:Timestamp Request
14:Timestamp Reply
15:Information Request (obsoleto)
16:Information Reply (obsoleto)
17:Address Mask Request
18:Address Mask Reply

ICMP

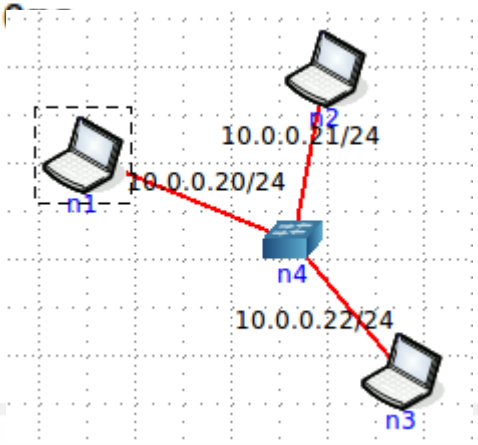
- **Echo Request y Reply (tipos 8 y 0)**
  - **Un equipo envía un request; el destino envía el reply con los mismos datos**
  - **Se comprueba el funcionamiento del destino y nodos intermedios**
  - **Campos Identificador y secuencia, para uso del emisor, se devuelven iguales**
  - **Codigo = 0**
  - **Utilizado por ping**



Ejemplo ARP – ICMP (1)

```
Terminal
File  Edit  View  Search  Terminal  Help
root@n1:/tmp/pycore.37224/n1.conf# ping 10.0.0.21 -c 1
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=64 time=0.149 ms

--- 10.0.0.21 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0
rtt min/avg/max/mdev = 0.149/0.149/0.149/0.000 ms
root@n1:/tmp/pycore.37224/n1.conf#
```



No.	Time	Source	Destination	Protocol	Length	Info
5	34.046521000	00:00:00_aa:00:00	Broadcast	ARP	42	Who has 10.0.0.21? Tell 10.0.0.20
6	34.046590000	00:00:00_aa:00:01	00:00:00_aa:00:00	ARP	42	10.0.0.21 is at 00:00:00:aa:00:01
7	34.046597000	10.0.0.20	10.0.0.21	ICMP	98	Echo (ping) request id=0x001c, seq=1/256, ttl=64 (reply i
8	34.046624000	10.0.0.21	10.0.0.20	ICMP	98	Echo (ping) reply id=0x001c, seq=1/256, ttl=64 (request

Frame 5: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

Ejemplo ARP – ICMP (2)

No.	Time	Source	Destination	Protocol	Length	Info
5	34.046521000	00:00:00_aa:00:00	Broadcast	ARP	42	Who has 10.0.0.21? Tell 10.0.0.20
6	34.046590000	00:00:00_aa:00:01	00:00:00_aa:00:00	ARP	42	10.0.0.21 is at 00:00:00:aa:00:01
7	34.046597000	10.0.0.20	10.0.0.21	ICMP	98	Echo (ping) request id=0x001c, seq=1/256, ttl=64 (reply i
8	34.046624000	10.0.0.21	10.0.0.20	ICMP	98	Echo (ping) reply id=0x001c, seq=1/256, ttl=64 (request

▸ Frame 5: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

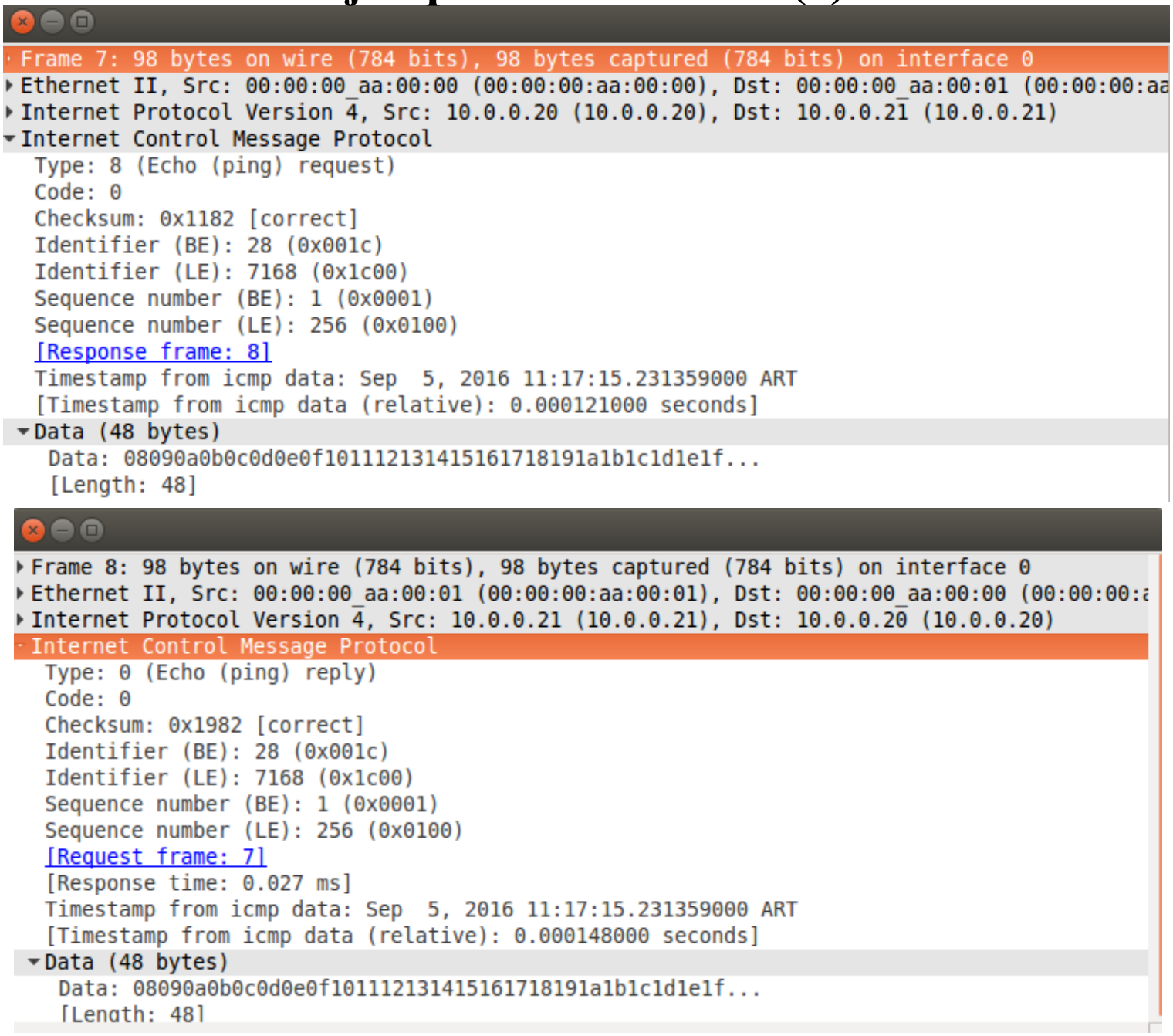
5 34.046521000 00:00:00_aa:00:00 Broadcast ARP 42 Who has 10.0.0.21? Tell 10.0.0.20

```
▸ Frame 5: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▸ Ethernet II, Src: 00:00:00_aa:00:00 (00:00:00:aa:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: 00:00:00_aa:00:00 (00:00:00:aa:00:00)
    Sender IP address: 10.0.0.20 (10.0.0.20)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.0.0.21 (10.0.0.21)
```

6 34.046590000 00:00:00_aa:00:01 00:00:00_aa:00:00 ARP 42 10.0.0.21 is at 00:00:00:aa:00:01

```
▸ Frame 6: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▸ Ethernet II, Src: 00:00:00_aa:00:01 (00:00:00:aa:00:01), Dst: 00:00:00_aa:00:00 (00:00:00:aa
▾ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: 00:00:00_aa:00:01 (00:00:00:aa:00:01)
    Sender IP address: 10.0.0.21 (10.0.0.21)
    Target MAC address: 00:00:00_aa:00:00 (00:00:00:aa:00:00)
    Target IP address: 10.0.0.20 (10.0.0.20)
```

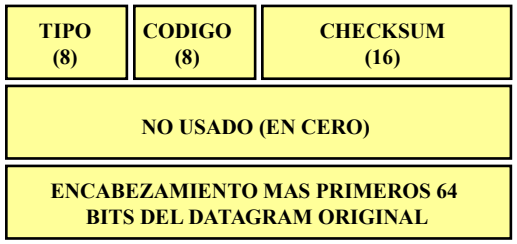
Ejemplo ARP – ICMP (3)



ICMP

- **Destination Unreachable (tipo 3)**
  - **Enviado por un router o host (destino) que no puede entregar el dg.**
  - **Códigos 0 a 5**

- **Time Exceeded (tipo 11)**
  - **Enviado por un router si TTL llega a 0**
  - **Códigos 0 a 1**



Código Destination Unreachablë:

0: Network Unreachable
1: Host Unreachable
2: Protocol Unreachable
3: Port Unreachable
4: Fragmantación y DF set
5: Falla en Source Route

Códigos en Time Exceeded:

0: TTL excedido (0)
1: Tiempo de reensamblado (fragmentación) excedido

Netfilter

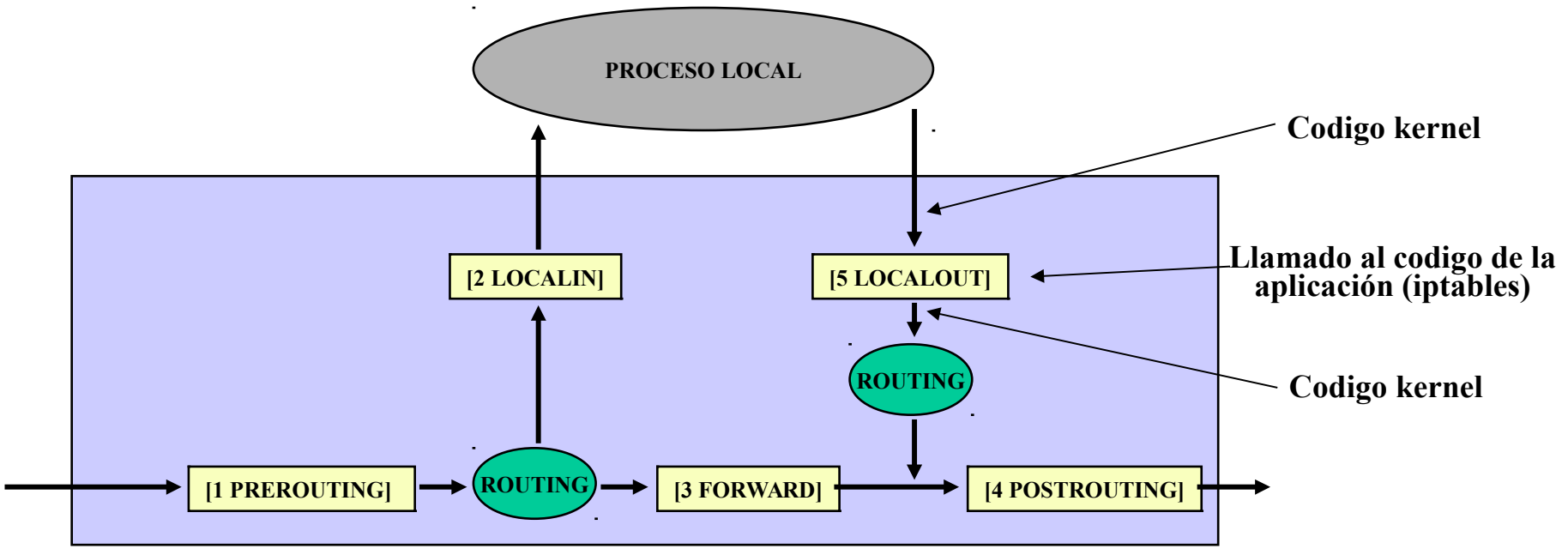
Netfilter:

- **Sistema de intercepción de paquetes en su camino por el Kernel**
- **Soporte para la implementación de diferentes funciones**
  - **NAT (p.ej, IPTables)**
  - **Filtrado de paquetes(p.ej, IPTables)**
  - **Backdoors**
  - **Etc.**

Puntos de intercepción: HOOKS

- **Los Hooks dependen del protocolo (IPv4, IPv6, etc)**
- **Cada Hook puede tener varios módulos registrados**
- **Funciones de intercepción de paquetes: permiten inspeccionar, modificar o descartar paquetes**
  - **Ubicación**
    - **Kernel (módulos)**
    - **Espacio usuario (ip_queue): tratamiento asincrónico**

Hooks Netfilter



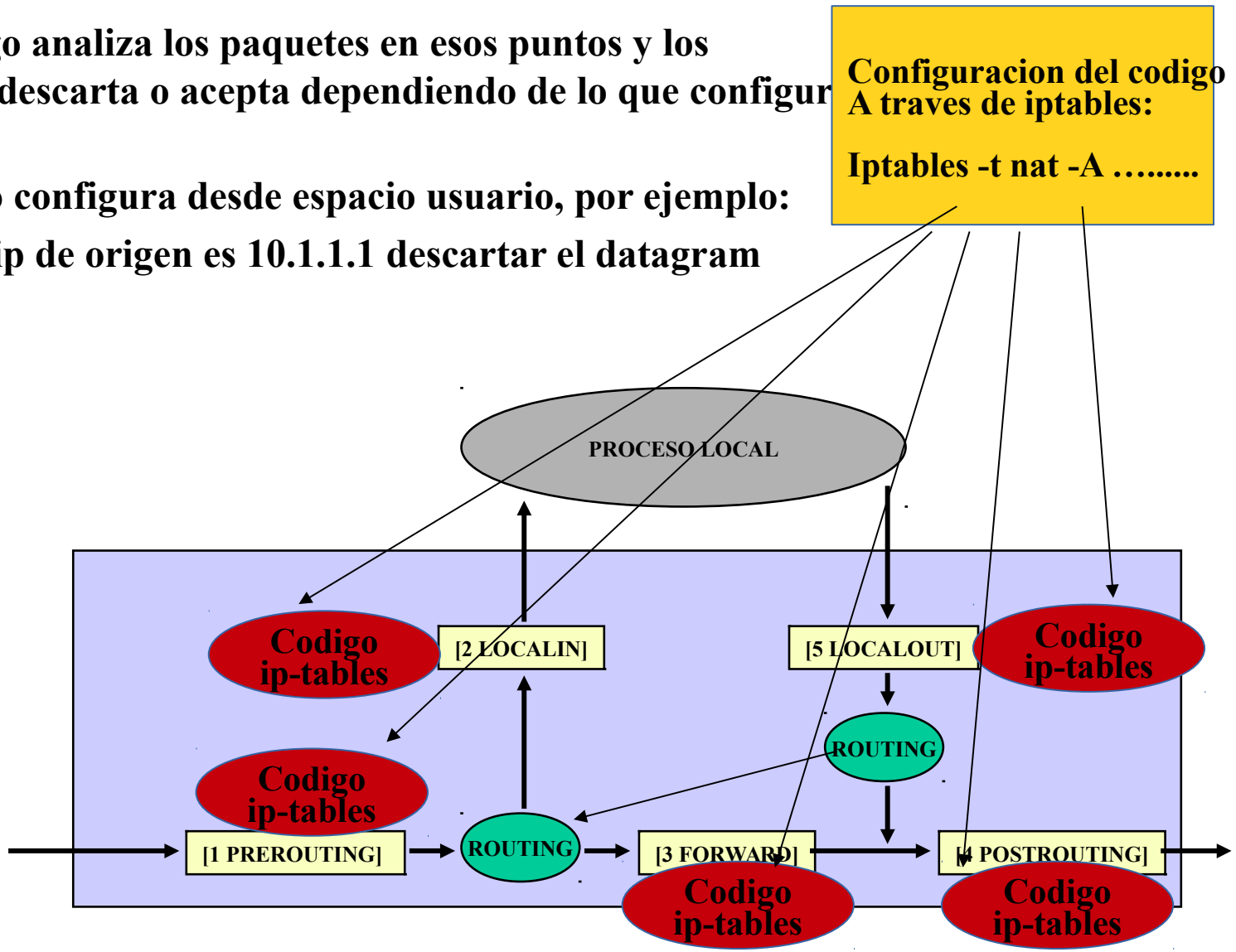
IPTABLES

- **Sistema de reglas de filtrado de paquetes**
- **Basado en el soporte Netfilter**
- **Configuración y cambio de reglas desde espacio usuario**
- **Funciones principales**
  - **Filtrado de paquetes**
  - **Modificación de paquetes**
  - **Identificación de “conexiones”**
  - **Network Address Translation**
  - **Interacción con Traffic Control**

IPTABLES

- Ip-tables se implementa a traves de codigo que es llamado por los hooks de netfilter
- Este codigo analiza los paquetes en esos puntos y los modifica, descarta o acepta dependiendo de lo que configur el usuario
- El usuario configura desde espacio usuario, por ejemplo:

  Si la ip de origen es 10.1.1.1 descartar el datagram



Componentes de IPTABLES - Tablas

- Cada tabla esta asociado con una clase de proceso que se desea hacer sobre los paquetes.
- Este proceso puede que se realice en uno o varios de los hooks
- Una tabla esta compuesta de varias cadenas, en el mismo y/o en diferentes hooks
- Existen tablas predefinidas y otras que pueden crearse (como LKM)
- Tablas predefinidas por iptables
  - FILTER: Se decide si un paquete es eliminado o continua su camino
  - RAW: Se utiliza para marcado en connection tracking
  - MANGLE: Modificacion de ciertos campos del paquete
  - NAT: Para funciones de NAT (origen y destino)

Componentes de IPTABLES - Cadenas

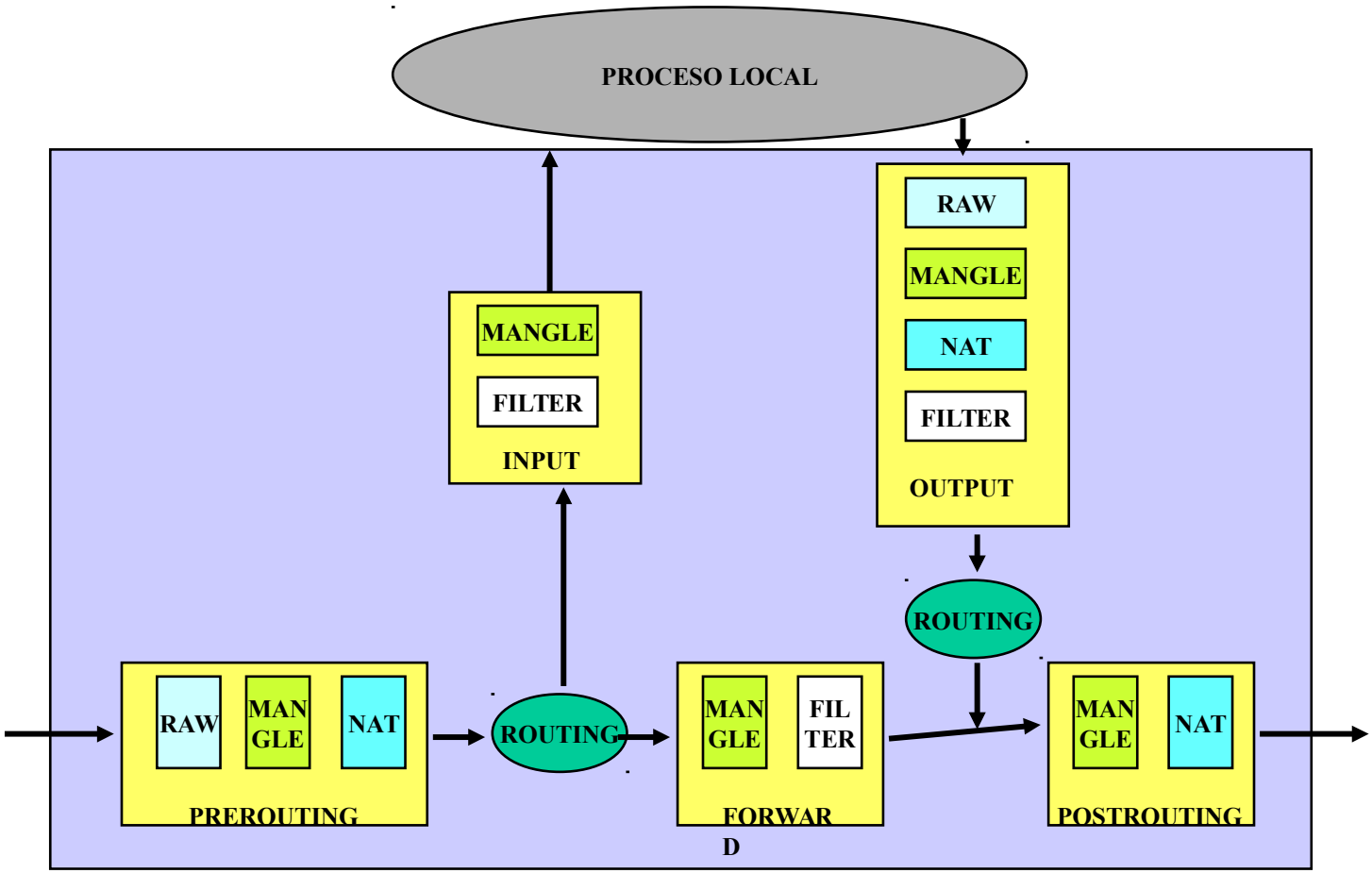
- Conjuntos de reglas ordenadas agrupadas lógicamente
- Al final de las reglas, se especifica una politica por defecto
- Cada paquete es procesado en orden (regla por regla)
- El paquete puede ser descartado, lo que lo elimina del sistema
- Si no es descartado, se le aplica la politica por defecto de la cadena

Tipos de cadenas

- Predefinidas
- Definidas por el usuario (desde la línea de comandos iptables)
- Sin política por defecto

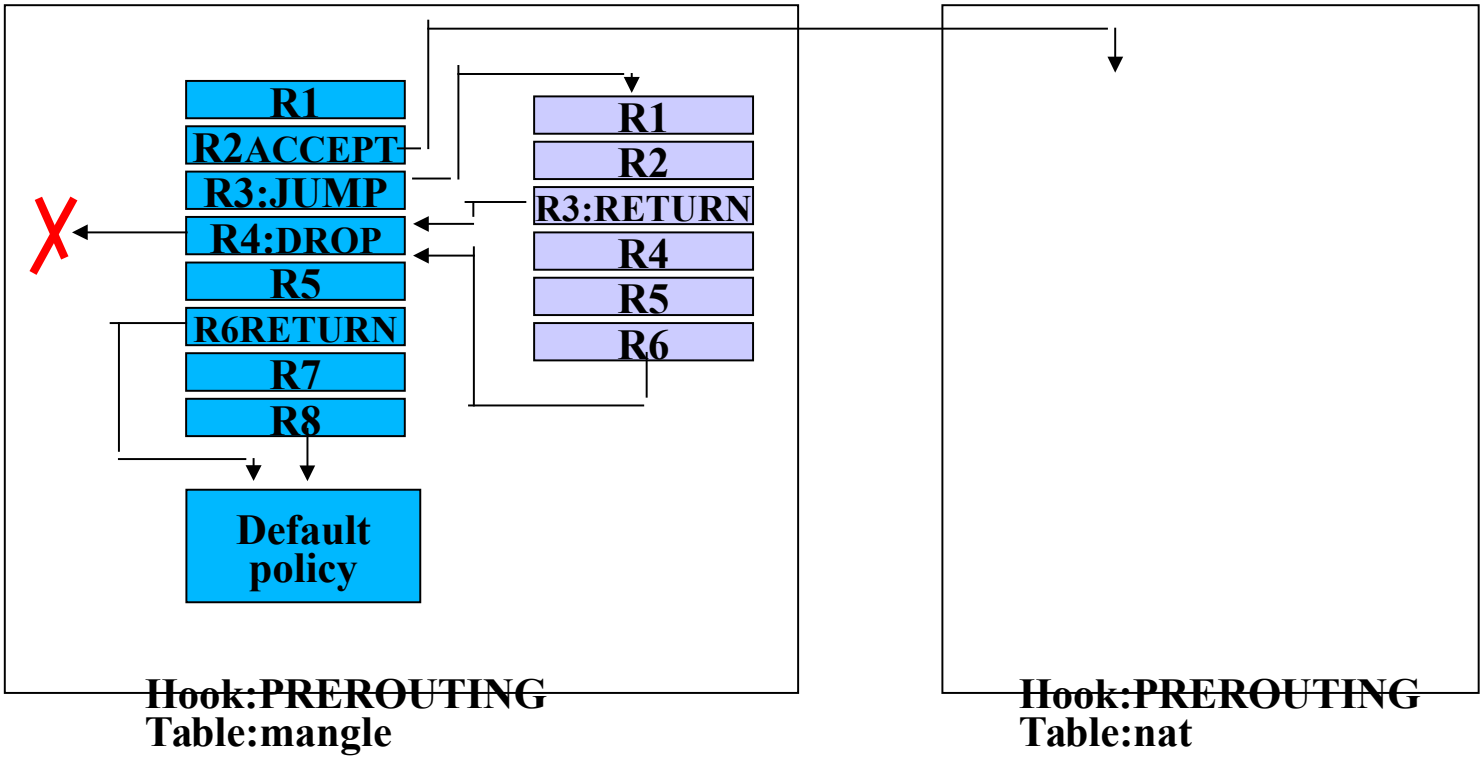
En cada hook se pueden definir multiples cadenas, que se ejecutan en un cierto orden

IPTABLES: Cadenas y tablas



IPTABLES: Cadenas y tablas (1)

- **Tablas: compuestas por cadenas**
  - **built-in (dependiente del hooks) (con default policy)**
  - **Definidas por el usuario (sin default policy)**



Reglas IPTABLES

si (C1 && C2 . . . && … && Cn == TRUE) target

Si (direccion de origen ==10.0.0.1 y port =80) descartar

Condiciones (matchings)

-p, --protocol [!] protocol
-s, --source [!] address[/mask]
-d, --destination [!] address[/mask]
-i, --in-interface [!] name
-o, --out-interface [!] name
[!] -f, --fragment
-c, --set-counters PKTS BYTES

-j, --jump target
-g, --goto chain

matching extensions (LKM)

Targets (acciones)

ACCEPT
DROP
QUEUE
RETURN

Cadena definida por el usuario

Target extensions (LKM)

Comando IPTABLES: manejo de reglas y cadenas

iptables [-t table] -[AD] chain rule-specification [options]
iptables [-t table] -I chain [rulenum] rule-specification [options]
iptables [-t table] -R chain rulenum rule-specification [options]
iptables [-t table] -D chain rulenum [options]

iptables [-t table] -[LFZ] [chain] [options]
iptables [-t table] -N chain
iptables [-t table] -X [chain]
iptables [-t table] -P chain target [options]
iptables [-t table] -E old-chain-name new-chain-name

options
-v
-n
-x
--line-numbers
--modprobe=command

Contadores de bytes y de paquetes
- **Por regla: Se incrementan cada vez que un paquete hace matching con la regla**
- **Por cadena: Se incrementan cada vez que un paquete hace matching con la política por defecto de la cadena**
- **Se ponen a cero con -Z**

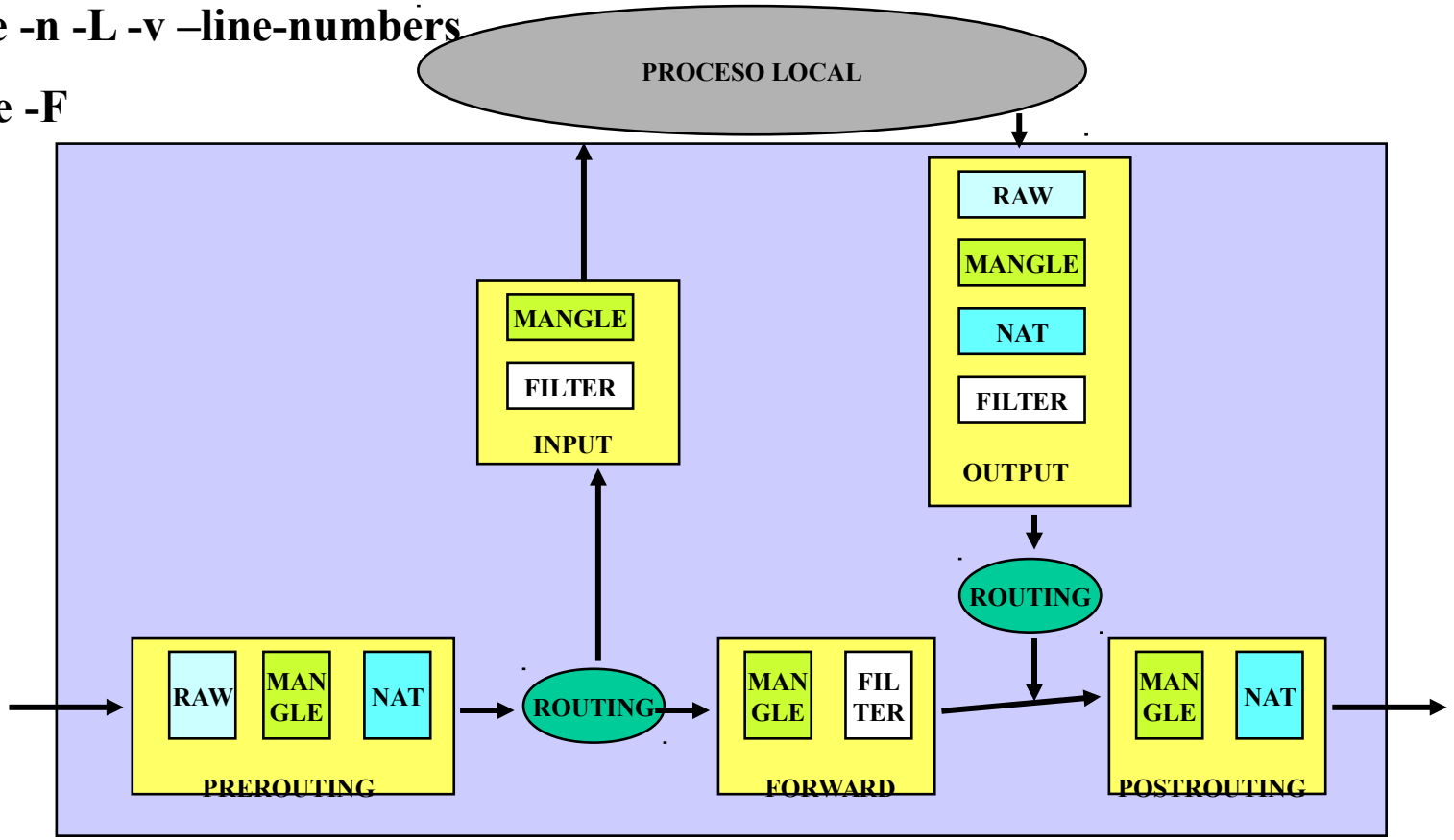
Ejemplo de reglas iptables

iptables -t filter -A OUTPUT -p tcp -j DROP

iptables -t mangle -A POSTROUTING -m ttl --ttl-gt 64 -j DROP

iptables -t mangle -n -L -v –line-numbers

Iptables -t mangle -F



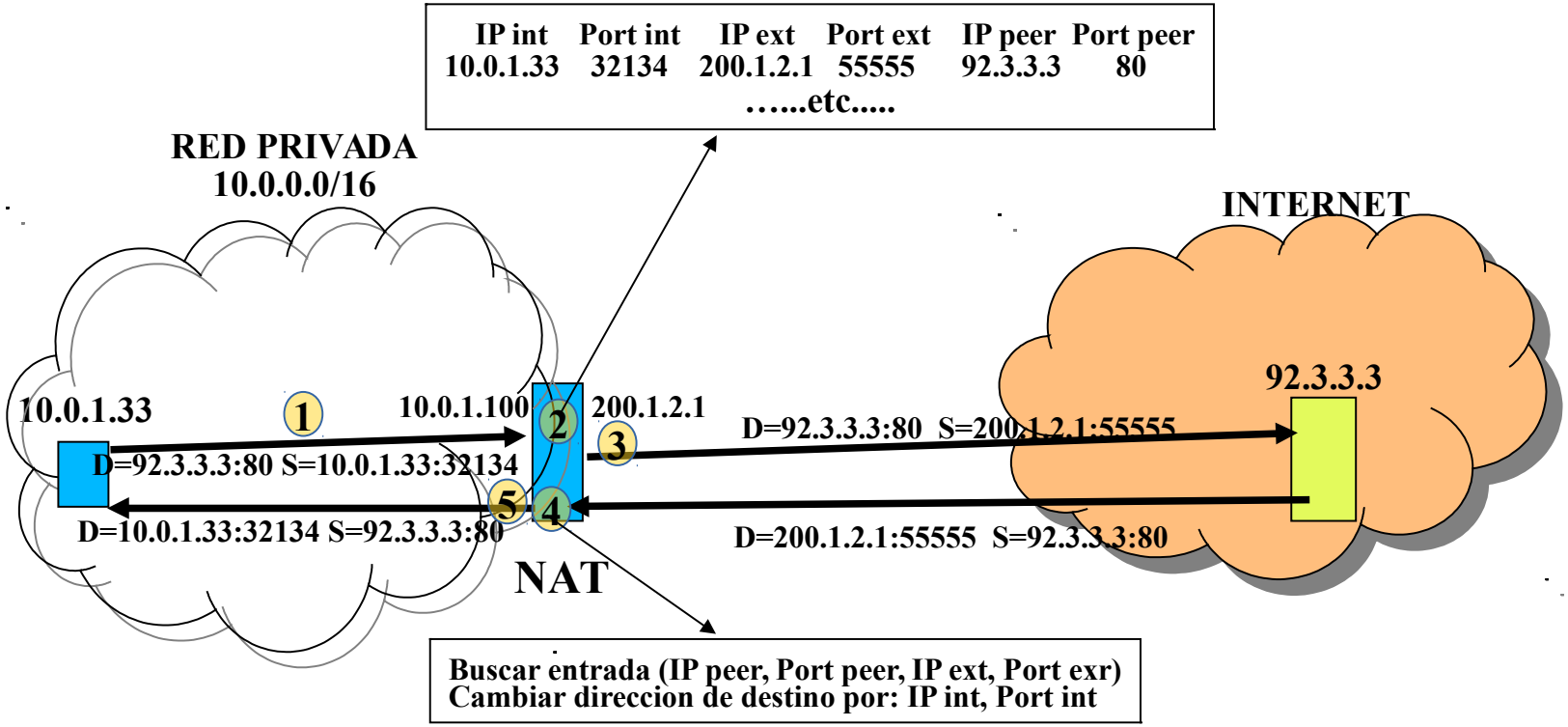
NAT (Network address translation)

La función NAT se sitúa en el router de salida de una intranet hacia la Internet

Consiste en cambiar las direcciones (y ports) de los paquetes que salen o entran en la intranet

Surge debido a la escasez de direcciones IPv4

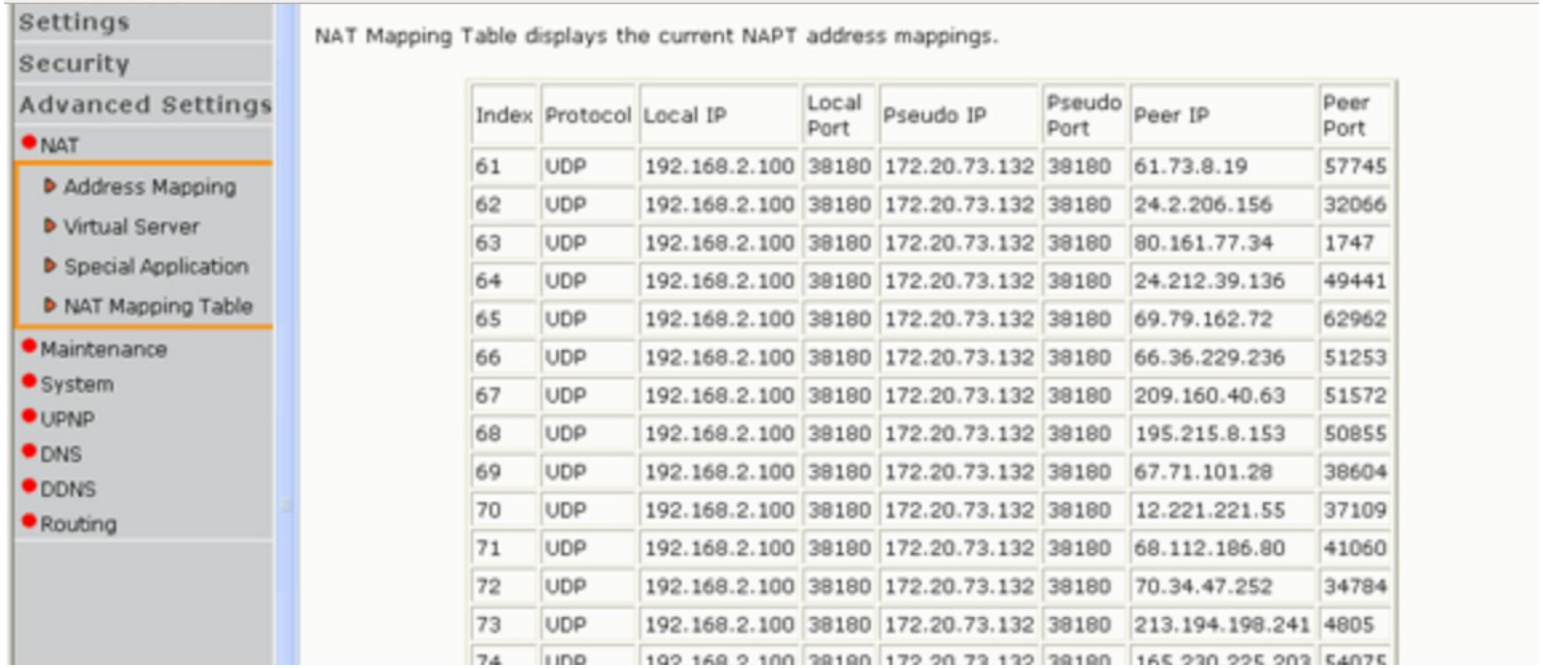
Funcionamiento simplificado de un NAT



Caso particular de NAT, se reemplaza direccion y port en el envio y en el arribo se exige que coincidan todos los campos
Si enviamos un segundo paquete al mismo destino y con el mismo port, no entra en la tabla de mapping (solo modifica)

1- Envio del host interno al externo
2- Creacion de una entrada enla tabla interna de mapping, selección de nueva IP y (en ciertos casos) nuevo port de origen
3-Envio del paquete modificado
4-Busqueda de entrada existente en la tabla, para filtrar y realizar conversion de direccion de destino
5-Envio del paquete modificado

Ejemplo de tabla de mapping en un NAT



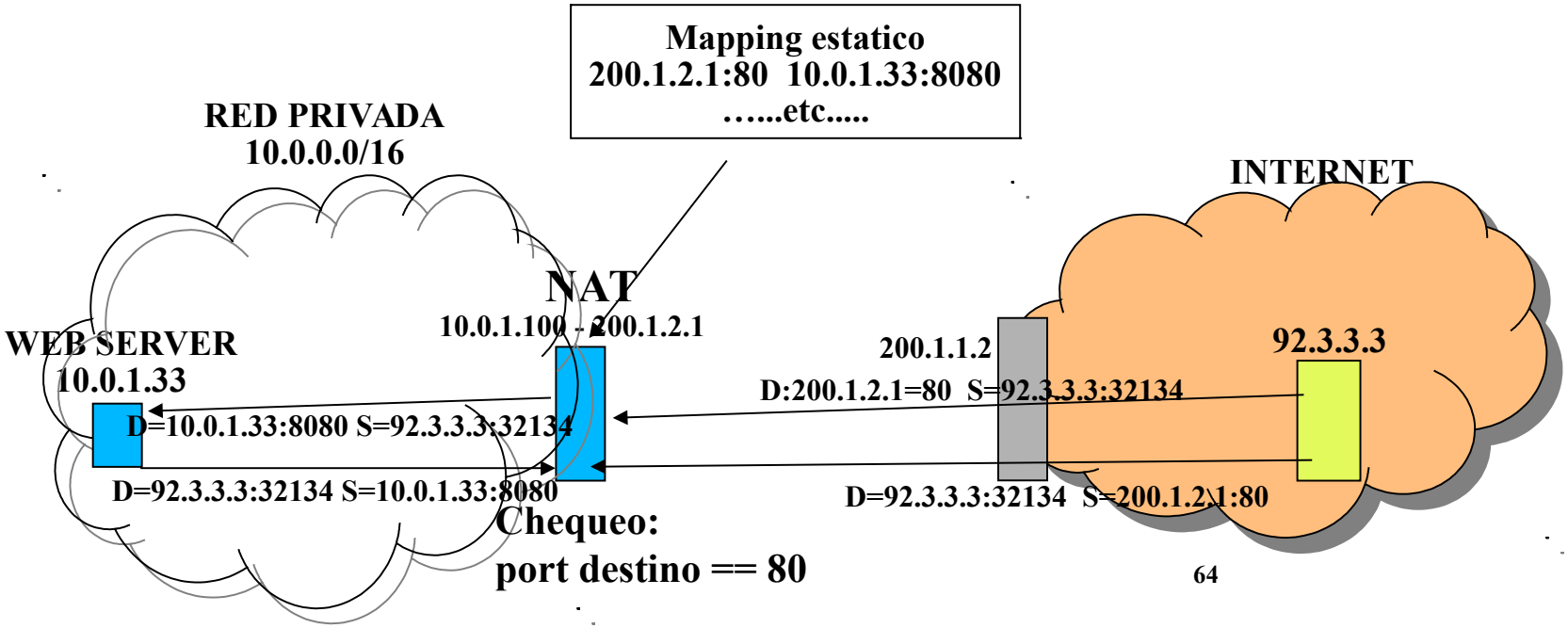
NAT Mapping Table displays the current NAPT address mappings.

Index	Protocol	Local IP	Local Port	Pseudo IP	Pseudo Port	Peer IP	Peer Port
61	UDP	192.168.2.100	38180	172.20.73.132	38180	61.73.8.19	57745
62	UDP	192.168.2.100	38180	172.20.73.132	38180	24.2.206.156	32066
63	UDP	192.168.2.100	38180	172.20.73.132	38180	80.161.77.34	1747
64	UDP	192.168.2.100	38180	172.20.73.132	38180	24.212.39.136	49441
65	UDP	192.168.2.100	38180	172.20.73.132	38180	69.79.162.72	62962
66	UDP	192.168.2.100	38180	172.20.73.132	38180	66.36.229.236	51253
67	UDP	192.168.2.100	38180	172.20.73.132	38180	209.160.40.63	51572
68	UDP	192.168.2.100	38180	172.20.73.132	38180	195.215.8.153	50855
69	UDP	192.168.2.100	38180	172.20.73.132	38180	67.71.101.28	38604
70	UDP	192.168.2.100	38180	172.20.73.132	38180	12.221.221.55	37109
71	UDP	192.168.2.100	38180	172.20.73.132	38180	68.112.186.80	41060
72	UDP	192.168.2.100	38180	172.20.73.132	38180	70.34.47.252	34784
73	UDP	192.168.2.100	38180	172.20.73.132	38180	213.194.198.241	4805
74	UDP	192.168.2.100	38180	172.20.73.132	38180	165.230.225.203	54075

Tipos de NAT

Full Cone NAT (NAT estatico)

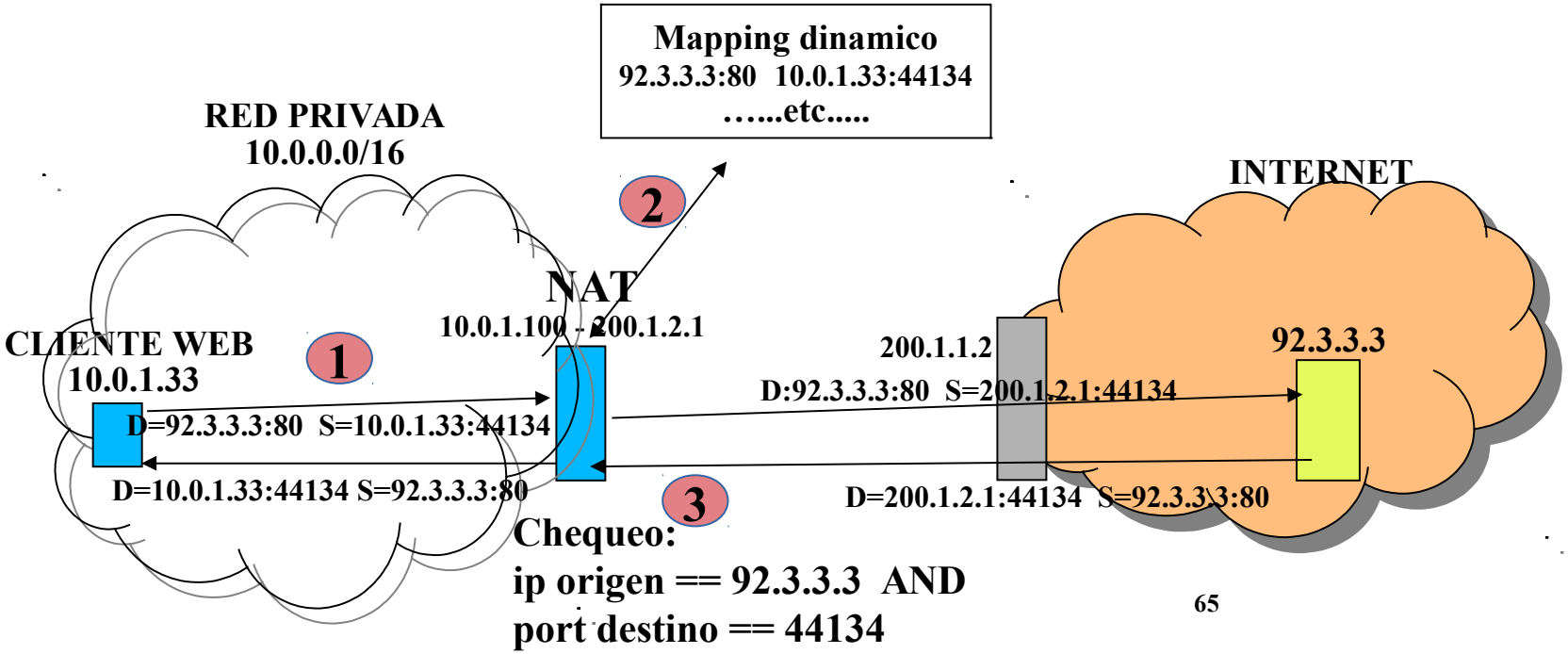
- Mapping estatico
- Port permanentemente abierto
- Una direccion publica por cada privada en uso
- Los ports pueden diferir
- Simple pero inseguro
- Acepta paquetes de cualqier IP si estan dirigidos a la IP y port definidos



Tipos de NAT (1)

Restricted Cone NAT (NAT dinamico)

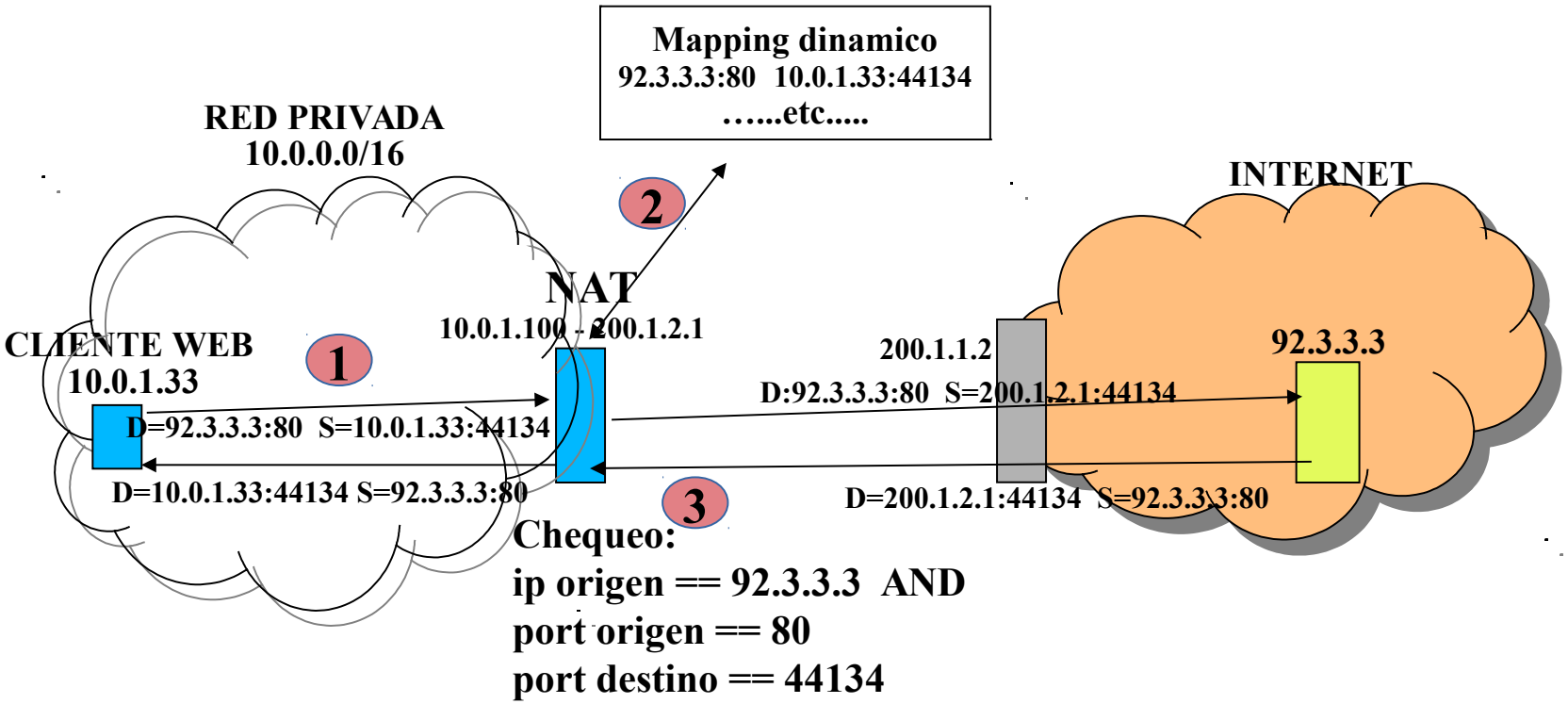
- Mapping dinamico (realizado cuando el host interno envia a uno externo)
- El port se abre solo desde el interior: No es posible contactar desde el exterior a un equipo interno
- Los ports externos e internos pueden diferir



Tipos de NAT (2)

Port Restricted Cone NAT (Nat dinamico)

Similar a Restricted Core NAT pero chequea ademas el port de origen



Tipos de NAT (3)

Symmetric NAT (Nat dinamico)

Similar a Port Restricted Core NAT

Para cada conexión del equipo interno, genera un port de salida aleatorio

Netstat-nat y stun

Mapping Symmetric

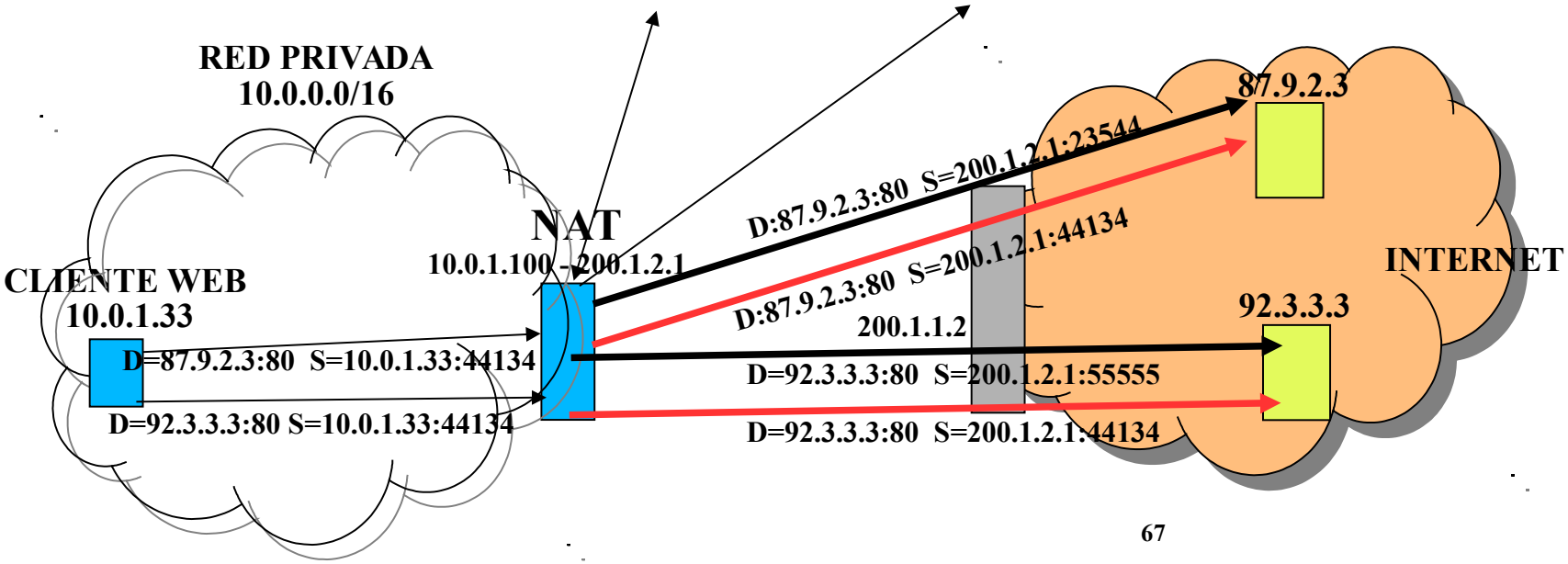
IP local	P_loc	IP_ext	P_ext	IP_peer	P_peer
10.0.1.33	44134	200.1.2.1	23544	87.9.2.3	80
10.0.1.33	44134	200.1.2.1	55555	92.3.3.3	80

……etc.....

Mapping no simetrico

IP local	P_loc	IP_ext	P_ext	IP_peer	P_peer
10.0.1.33	44134	200.1.2.1	44134	87.9.2.3	80
10.0.1.33	44134	200.1.2.1	44134	92.3.3.3	80

……etc.....



Ventajas y desventajas de NAT

- **Desventajas**
  - **Necesidad de Application Level Gateways (ALG)**
    - **Especificos para cada protocolo**
    - **Aplicaciones que transportan direcciones IP (SNMP)**
    - **Aplicaciones con conexiones interrelacionadas (p.ej FTP)**
  - **Performance**
    - **Recalculo de checksums si modifica direcciones**
    - **Recalculo de TCP checksum si se modifican ports**
    - **Controles adicionales si se fragmenan paquetes**
  - **Perdida de visibilidad de los equipos internos**
  - **Preservacion de ports 4787**
  - **Comportamiento no totalmente definido**
  - **Consideraciones sobre NATs sucesivos**
  - **Incompatibilidades con IPSec**

- **Ventajas**
  - **Soluciona temporariamente la escacez de direcciones IPv4**
  - **Elimina los costos de cambio de direccion de red**
  - **Mejora la privacidad de la red**

NAT en iptables

- Tabla “nat” de iptables
- Sólo consultada para el primer paquete que define una conexión

- Targets:
  - DNAT
    - válido en PREROUTING y OUTPUT
    - Cambia dirección y/o port de destino
  - SNAT:
    - Válido en POSTROUTING
    - Cambio dirección y/o port de origen
  - MASQUERADING:
    - Válido en POSTROUTING
    - Cambio dirección y/o port de origen
    - Sólo recomendable para asignación dinámica de IP
  - SAME
    - Trata de preservar mapping conexión-destino

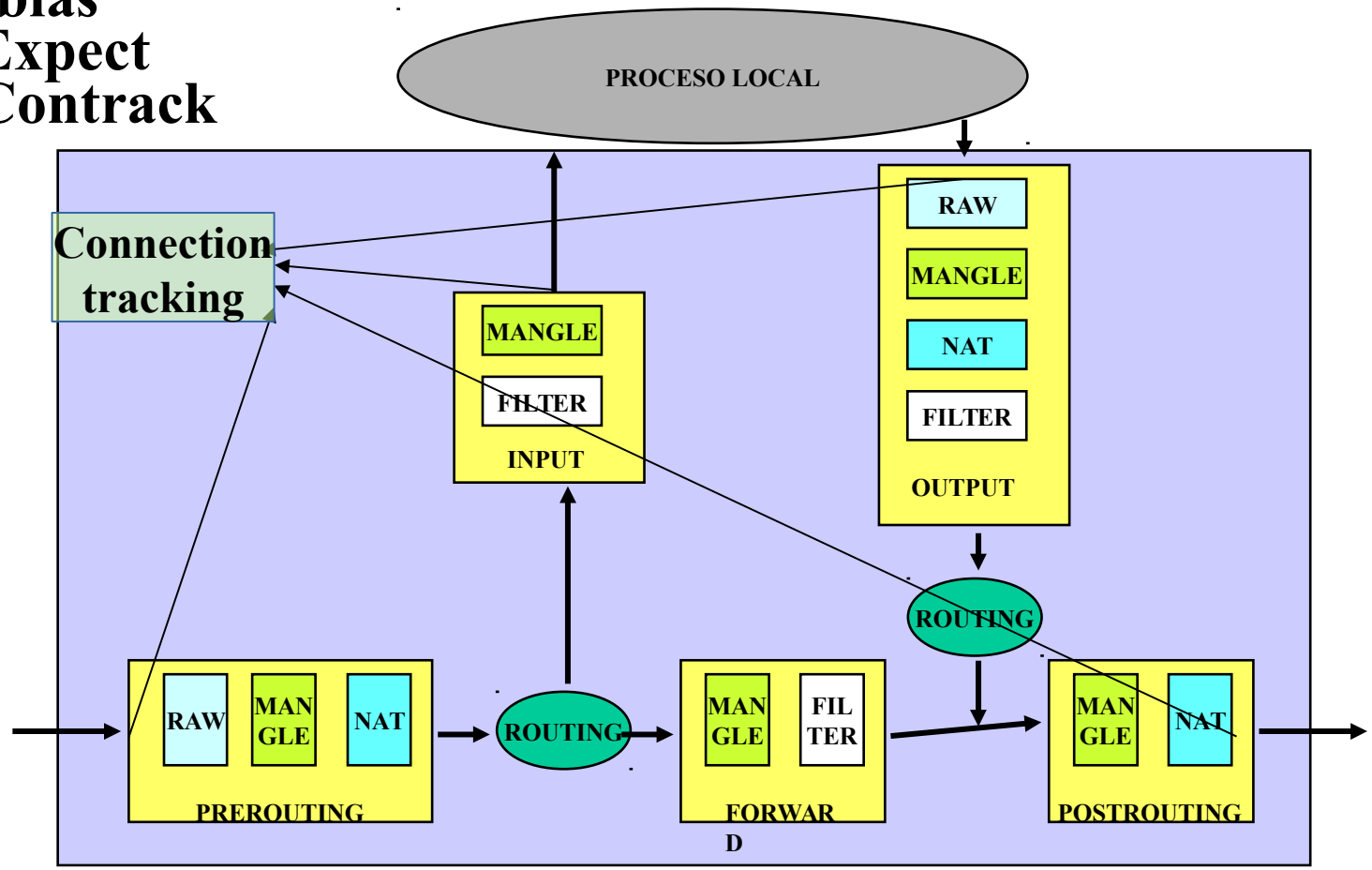
Connection tracking

- **Subsistema de identificación de “conexiones” provisto por Netfilte**
- **Detecta y monitorea conexiones**
- **Ofrece a iptables el estado de la conexión para cada paquete**
- **Compilado en el Kernel o cargado dinámicamente (LKMs)**
- **Requiere defragmentación de paquetes**
- **Uso de helpers para protocolos con mayor complejidad**
- **Requerido para**
  - **NAT**
  - **firewalls stateful**

Funcionamiento de connection tracking

- Funciones
  - Entrada: determina si hay ya un matching para la conexión
  - Salida: confirma el matching
  - Helpers: especificos para ciertos protocolos (p.ej FTP)

- Tablas
  - Expect
  - Contrack



Estados de una “conexion”

- **Conexion: flujo relacionado de paquetes (p. ej conexión TCP, etc)**

- **Estados de una “conexion:**
  - **NEW**
    se ha detectado el primer paquete de una futura conexión (p.ej el primer SYN TCP)
  - **ESTABLISHED**
    se han detectado paquetes de una misma conexión en ambos sentidos
  - **RELATED**
    Cuandi habiendo ya una conexión ESTABLISHED, se detecta un paquete de una nueva conexión felacionada a la anterior (p.ej. Una conexión de datos GTP es RELATED a la conexión de control)
  - **INVALID**
    Paquetes que no pueden ser asociados a ninguna conexión, p.ej. ver un echo reply sin haber visto un request
  - **UNTRACKED**
    Cuando se hace un marching NOTRACK en raw, el paquete y sus posibles REKATED no son vistos por CT

- **Estados utilizados por NAT y en sentencias iptables**
- **Desde espacio usuario: contrack (p.ej contrack -L expect)**

Alternativas para las funciones de mapping y filtrado

- Tipos de mapping de la direccion de origen (Di:Pi) → (De:Pe)
  - Independiente del destino
    - Cualquiera sea la direccion de destino, el mapping es el mismo
  - Dependiente de la direccion de destino
    - Se hace un mapping diferente por caa direccion de destino (no se tiene en cuenta el port
  - Dependiente del port y direccion de destino
    - Para cada direccion y port de destino, el mapping es diferebte

- Tipos de filtrado de paquetes entrantes, segun direccion de destino
  - Independiente del destino
    - Se permite entrar todos los paquetes dirigidos a una direccion externa mapeada, sin importar de donde provienen
  - Dependiente de la direccion de destino
    - Se permite entrar solo los paquetes que vienen de la direccion IP a la cual se envio, independientemente del port
  - Dependiente del port y direccion de destino
    - Se permite entrar solo los paquetes que vienen de la IP y el port al cual el host interno envio

Extensiones a matches

-p icmp --icmp-type [!] typename

Solo valida si se ha especificado -p icmp. Permite selecionar el codigo (p.ej. Echo request, etc.)

--pkt-type [unicast|broadcast|multicast]

Permite seleccionar por el tipo de envio link layer

-p tcp [!] --syn

Permite seleccionar los segmentos TCP según tengan o no SYN

Extensiones a targets

REJECT

Termina con el proceso del paquete yy envía ICMP

Opciones: --reject-with type

BALANCE

Permite distribuir los paquetes en forma round robin entre varias direcciones de destino

Opciones: --to-destination ipaddr-ipaddr

TTL

Modifica el valor de ttl de los paquetes

Válido solo en tabla mangle

Opciones: --ttl-set valor; --ttl-inc valor; --ttl.dec valor