UNIDAD 2

PROTOCOLO IP

CC4303 - Redes

CONTENIDOS DE LA UNIDAD

- Concepto de CATENET
- Direcciones IP (v4 y v6)
- Paquete IP (v4 y v6)
- Redes
- Manejo de Errores
- Ruteo y Fragmentación

CONCEPTO DE CATENET

- Conmutación de Circuitos vs Conmutación por Paquetes.
- Principios de TCP/IP

CONMUTACIÓN DE CIRCUITOS VS CONMUTACIÓN POR PAQUETES (1)

- Diferencias entre las redes IP y las redes de telecomunicaciones clásicas.

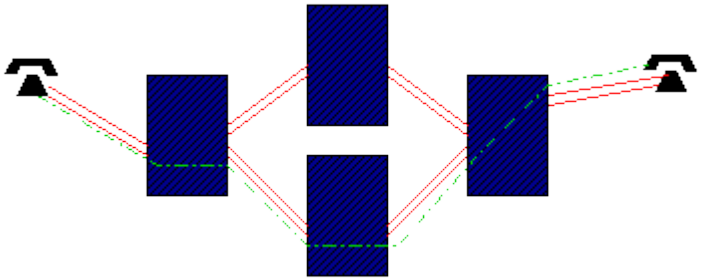
Características	Computación	Telecom
Puntas	Computadores	Teléfonos TV, tontos
Usuarios	Aplicaciones	Personas
Interfaz	Open/Close Read/Write	Interrupción Señal
Servicios	Usuarios	Carrier
Redes	Tontas	Inteligentes
Cuello Botella	Routers	Switches
Tráfico	Datos	Audio/Video (isócronos)
Regulación	Ninguna	Gobierno
Crecimiento	Desde abajo	Desde arriba

CONMUTACIÓN DE CIRCUITOS VS CONMUTACIÓN POR PAQUETES (2)

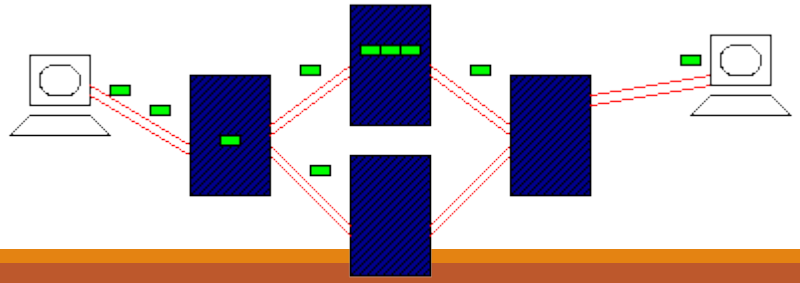
- Tradicionalmente:
  - Redes de Datos: Mail, FTP.
  - Redes de Telecomunicaciones: Audio, Video análogos.
- Actualmente
  - Redes de Datos: Web, Multimedia, Audio, Video.
  - Redes de Telecomunicaciones: Audio y Video Digitales
- Convergencia o Colisión?

CONMUTACIÓN DE CIRCUITOS VS CONMUTACIÓN POR PAQUETES (3)

- Conmutación de Circuitos



- Conmutación por Paquetes



PRINCIPIOS DE TCP/IP(1)

- El nivel de red de TCP/IP tiene como misiones el ruteo, el manejo de congestión y de errores.
- Ahora el problema radica en poder conectar una máquina con otra pasando por múltiples redes y enlaces, incluso cambiando la representación física de los paquetes y el encapsulamiento de una red a otra.
- El protocolo que opera en esta capa y se encarga de lo anterior es el protocolo IP (Internet Protocol).

PRINCIPIOS DE TCP/IP(2)

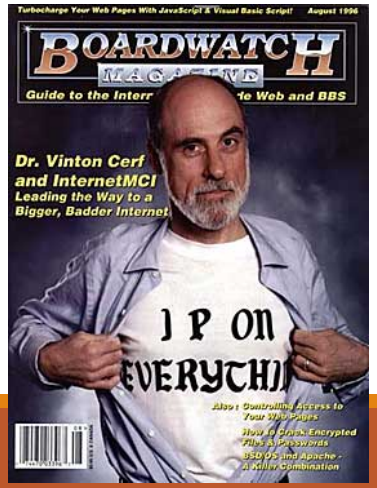
- IP tiene dos principios básicos:
- *1.- End-to-end Argument*:
  - La inteligencia y el manejo de las conexiones va en las puntas.
  - Nunca hacemos algo en los nodos intermedios, si podemos hacerlo en el origen y/o en el destino.
  - De este modo, todos los costos por saturación, estado por conexión, etc, los manejan los nodos interesados en la conexión y no el resto.

PRINCIPIOS DE TCP/IP(3)

- IP tiene dos principios básicos:
- *1.- End-to-end Argument*:
  - Lo anterior permite no mantener el estado de las conexiones en la red misma, sino sólo en el origen y el destino.

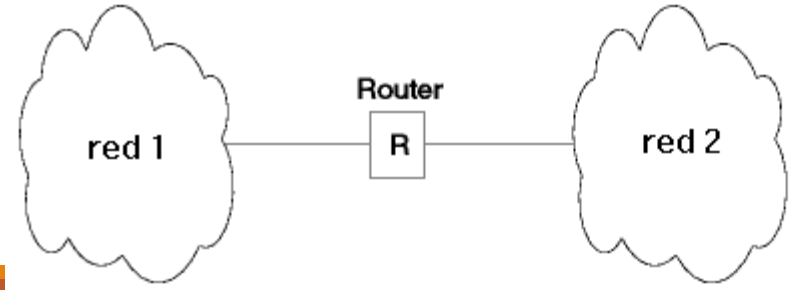
PRINCIPIOS DE TCP/IP(4)

- IP tiene dos principios básicos:
- 2. - IP sobre todas las cosas:
  - La idea es definir un protocolo independiente de la red física, que logre pasar a través de todos los medios y no dependa de ninguno en particular.



PRINCIPIOS DE TCP/IP(5)

- IP se basa en el concepto de una inter-red, que es una red de redes. Es decir, no conecto un computador con otro, sino una red con otras (CATENET): red debe tener un identificador único
- Esto me permite factorizar una buena cantidad de información, puesto que solo debo identificar la red destino para rutear.



PRINCIPIOS DE TCP/IP(6)

- Supuestos sobre una red física
  - Conecta a todos los computadores que están en la misma red física (red local)
  - Permite enviar bytes, en alguna codificación, de un computador a cualquier otro
  - Los computadores se identifican con una dirección física única en la red (conjunto de bits)
  - Puede soportar broadcast: mensaje a todos los computadores conectados
  - Un computador puede estar conectado a varias redes físicas simultáneamente (interfaces)

PRINCIPIOS DE TCP/IP(7)

- El Router (que es un computador que conecta dos redes entre sí) se encarga del cambio de formato físico y del ruteo entre ambas redes. Para lograr hacer esto requerimos las siguientes características para IP:
  - 1.- Espacio de nombres (direcciones IP)
    - Únicos en toda la Inter-red
    - Independientes de la Red
    - Traducibles a direcciones físicas
    - Identifican la red y al dispositivo (ej: multi-interfaz => múltiples direcciones IP

PRINCIPIOS DE TCP/IP(8)

- El Router (que es un computador que conecta dos redes entre sí) se encarga del cambio de formato físico y del ruteo entre ambas redes. Para lograr hacer esto requerimos las siguientes características para IP:
  - 2.- Ruteo de los datos en base al “nombre” IP del destino (red+dispositivo)
  - 3.- Paso de los datos por los routers sin alteración
  - 4.- División de los datos en paquetes independientes

DIRECCIONES IP

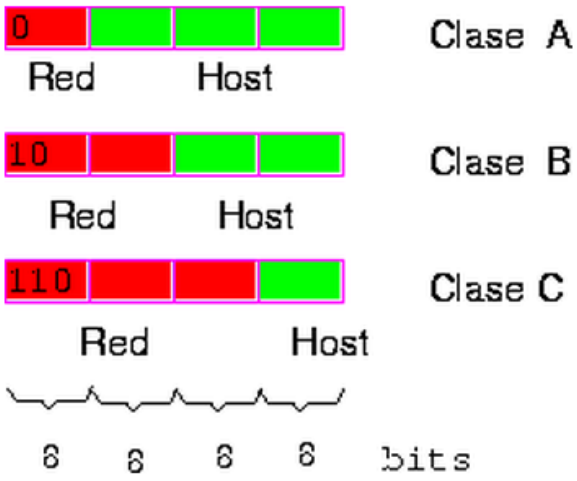
- Direcciones IPv4
- Direcciones IPv6
- Traducción a Dirección Física

DIRECCIONES IPV4 (1)

- Para el nivel red, se define un espacio de direcciones de 32 bits, que serán usados en forma única para identificar cada computador conectado a la Inter-red.
- Para permitir ruteo fácil en base a la dirección IP, se dividen los bits en una dirección de red (bits superiores) y una dirección de host (bits inferiores).

DIRECCIONES IPV4 (2)

- La notación de las direcciones IPv4 es en base a cuatro decimales separados por un punto (cada decimal codifica un byte: 0 - 255).
  - Por ejemplo: 146.83.4.11 (IPv4)
- Inicialmente, las direcciones se separaron en clases A, B y C. La idea era que la separación entre bits de red y bits de host es en un byte distinto para cada clase.



DIRECCIONES IPV4 (3)

- Al rutear, se debe separar el prefijo de red del sufijo de host. Luego, se rutea en base al prefijo.
- Sólo cuando ya estoy en la red correcta, utilizo el sufijo de host para encontrar la máquina destino en la red local.
- El número de bits del sufijo determina cuantos hosts puedo tener en la misma red: 256 para una clase C, 65.536 para una clase B y 16.777.216 para una clase A.
- Los prefijos de red deben ser **únicos** en todo el mundo, por lo cual son asignados centralizadamente.
- El ligar el ruteo con la dirección de la máquina implica que una máquina conectada a varias redes posee varias direcciones IP (una en cada red), lo que puede hacer que en un momento sea accesible por una dirección y no por otra.

DIRECCIONES IPV4 (4)

- Direcciones IP Reservadas:
  - 0.0.0.0 “Este” Host (origen).
  - 0.0.x.x Host en “esta” red.
  - 255.255.255.255 Broadcast local.
  - x.x.255.255 Broadcast en “esta” subred.
  - 127.x.x.x.Loopback.

DIRECCIONES IPV6 (1)

- Objetivos
  - Principales
    - Enfrentar la escasez de direcciones IPv4.
    - Enfrentar el excesivo crecimiento de la tabla de rutas global de Internet.
  - Secundarios
    - Soporte para servicios de tiempo real.
    - Soporte para seguridad.
    - Autoconfiguración.
    - Funcionalidades extendidas de ruteo, por ejemplo, para host móviles.

DIRECCIONES IPV6 (2)

- Principales Características

- Direcciones de 128 bits, lo que da un total de 3.4 * 10^{38} para un 100% de eficiencia.
- Aprox 667 * 10^{21}
- Pero estimaciones han determinado que habrían 1500 direcciones IP por metro cuadrado de superficie terrestre aun con la tasa de pérdida más pesimista posible (ver Host-Density Ratio)

  - No existen clases, pero sí agrupaciones por prefijos de bits. Algunos para mapear direcciones IPX, NSAP, IPv4. Otros para multicast, para “direcciones privadas”, etc.

DIRECCIONES IPV6 (2.5)

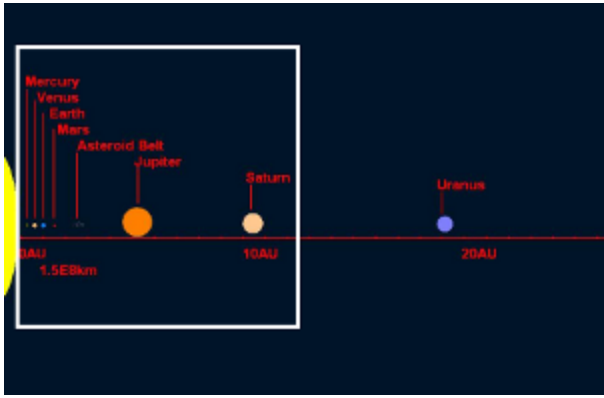
IPv4 address space

(256 / pixel)

IPv6: 10^38

Estrellas: 10^22

Átomos: 10^81



IPv6 address space

(256 / pixel)

DIRECCIONES IPV6 (3)

- Prefijos Asignados en IPv6:

Table 24. :IPv6 - Format prefix allocation

Allocation	Prefix (bin)	Start of address range (hex)	Mask length (bits)	Fraction of address space
Reserved	0000 0000	0:: /8	8	1/256
Reserved for NSAP	0000 001	200:: /7	7	1/128
Reserved for IPX	0000 010	400:: /7	7	1/128
Aggregatable Global Unicast Addresses	001	2000:: /3	3	1/8
Link-local Unicast	1111 1110 10	FE80:: /10	10	1/1024
Site-local Unicast	1111 1110 11	FEC0:: /10	10	1/1024
Multicast	1111 1111	FF00:: /8	8	1/256
Total Allocation				15%

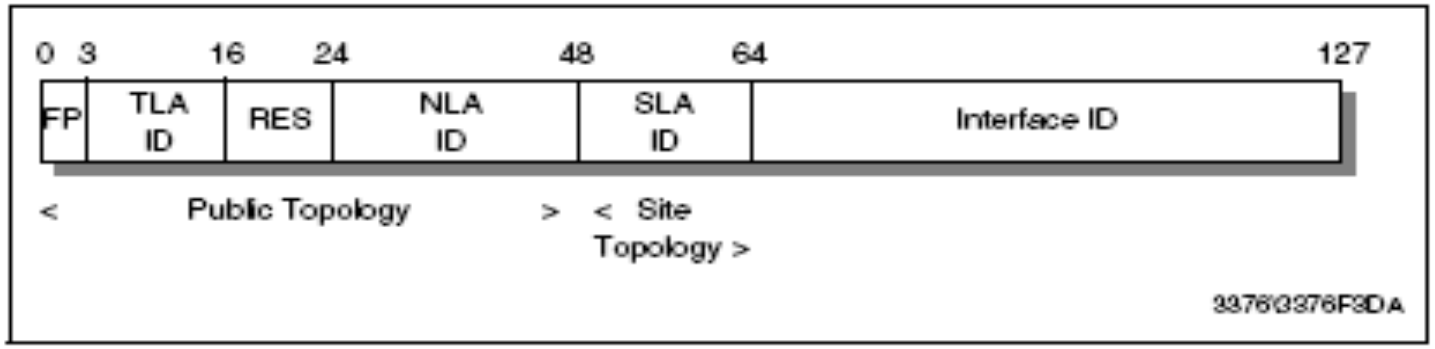
DIRECCIONES IPV6 (4)

- Prefijos Asignados en IPv6:
  - Aggregatable Global Unicast Addresses: Rango de direcciones para asignar a interfaces de red públicas y únicas
  - Site-local Unicast: Direcciones que no pueden ser publicadas en Internet. Equivalentes a las direcciones IPv4 privadas.
  - Link-local Unicast: Direcciones que sólo son válidas a nivel de la misma red física.
  - Los locales necesitan un "*zone_id*" (interfaz), ej:

fe80::5626:96ff:fedb:f347%en0

DIRECCIONES IPV6 (5)

- Jerarquía de las direcciones unicast.



3	m	n	o	p	125 - m- n- o- p
010	RegistryID	ProviderID	SubscriberID	SubnetID	InterfaceID

DIRECCIONES IPV6 (6)

- Notación de las direcciones
  - Ejemplo: 47CD:1234:4422:AC02:0022:1234:A456:0124
  - 47CD:0000:0000:0000:0000:0000:A456:0124 pueden ser escrita como 47CD:0:0:0:0:0:A456:124 y más brevemente como 47CD::A456:124
  - www.nic.cl: 2001:1398:5::6003
  - Una dirección IPv4 128.96.33.81 se mapea en IPv6 como ::FFFF:128.96.33.81
  - ::1 es localhost (como 127.0.0.1)

TRADUCCIÓN A DIRECCIÓN FÍSICA (1)

- Utilizando la parte de Red de la dirección IP se rutean los paquetes hasta la red de destino.
- Una vez que ya llegó a la red de destino hay que entregar el datagrama a su máquina destino correcta.
- Esto no es trivial, puesto que solo conozco su dirección IP, y lo que requiero ahora es su dirección física. Por supuesto, este problema también se aplica al tener que enviar paquetes a routers en mi propia red, para que ellos continúen el ruteo.

TRADUCCIÓN A DIRECCIÓN FÍSICA (2)

- En definitiva, nuestro problema general es enviar un datagrama cualquiera a una máquina de mi red, conociendo sólo su dirección IP.
- Obviamente, conozco la interfaz de red por donde debo enviar el datagrama y también conozco la forma de encapsular el datagrama en un frame físico de este tipo de red. Pero aún así, debo encontrar la dirección física del destinatario (dirección ethernet, NSAP ATM, número de Token Ring, etc). Esto es un protocolo de traducción de direcciones, que depende de la red física en cuestión.

TRADUCCIÓN A DIRECCIÓN FÍSICA (3) IPv4

- El protocolo más usado para IPv4 es ARP (*Address Resolution Protocol*) en medios que soportan *broadcast*. Broadcast de un paquete con protocolo ARP (esto no es un datagrama IP), que contiene mis datos (dirección IP y física) y la dirección IP que quiero resolver.
- Este paquete es recibido por todas las máquinas de la red, quienes verifican si tienen esa dirección IP registrada como propia o no.
- quien tiene esa dirección como propia debe responder, completando la información del paquete ARP y enviándolo directamente al origen.

TRADUCCIÓN A DIRECCIÓN FÍSICA (4) IPv4

- Al recibir la respuesta, el origen agrega a una tabla local (en memoria) el par dirección IP, dirección física.
- Este cache de ARP sirve para no preguntar por cada paquete en caso de tráfico continuo. Sin embargo, los valores deben ir expirando en el tiempo para permitir cambios de asociaciones dinámicos.

TRADUCCIÓN A DIRECCIÓN FÍSICA (5) IPv4

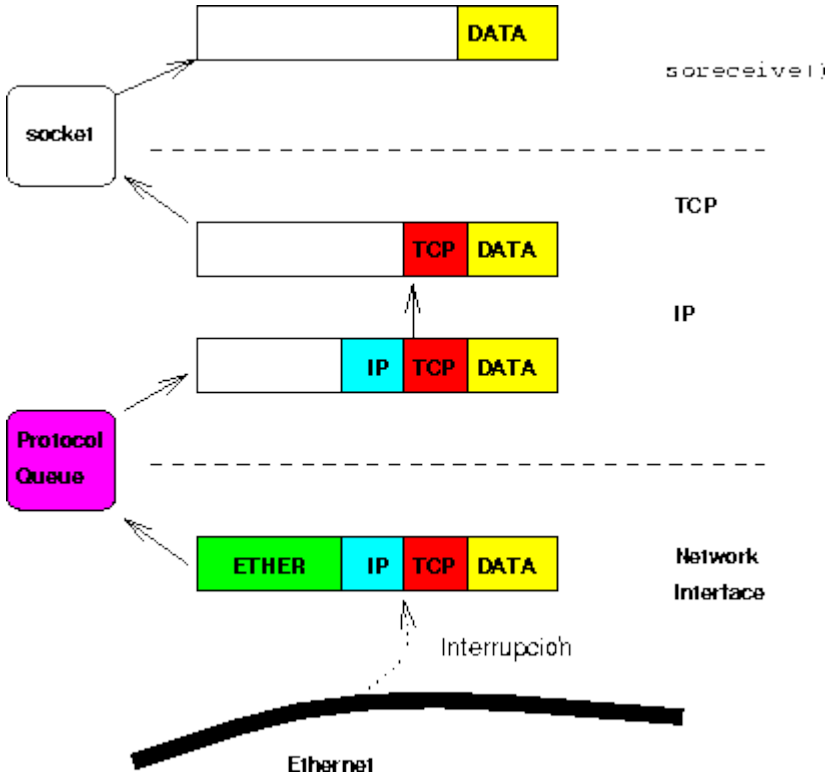
- En el caso de redes que no soportan *broadcast*, debemos tener tablas locales (mantenidas por personas) o un servidor ARP central a quien enviarle las preguntas (como un directorio).
- En este último caso, requiero que configuren a mano la dirección física de dicho servidor solamente.
- ARP es uno de los problemas de IP en redes grandes, puesto que genera broadcasts que ningún filtro de nivel físico puede parar. Tormentas de broadcasts son frecuentes en redes con más de 100 máquinas.

TRADUCCIÓN A DIRECCIÓN FÍSICA (6)

- Existen algunas redes físicas que trivializan esta operación, por ejemplo en Token Ring las estaciones están numeradas en el orden del anillo, y puedo usar el último byte de la dirección IP para almacenar el número de la estación.
- En este caso, puedo traducir directamente, sin ningún protocolo de red.

TRADUCCIÓN A DIRECCIÓN FÍSICA (7)

Al encapsular un paquete IP en un paquete físico, termina el proceso de empaquetamiento y finalmente es emitido el datagrama a la red.



TRADUCCIÓN A DIRECCIÓN FÍSICA (8)

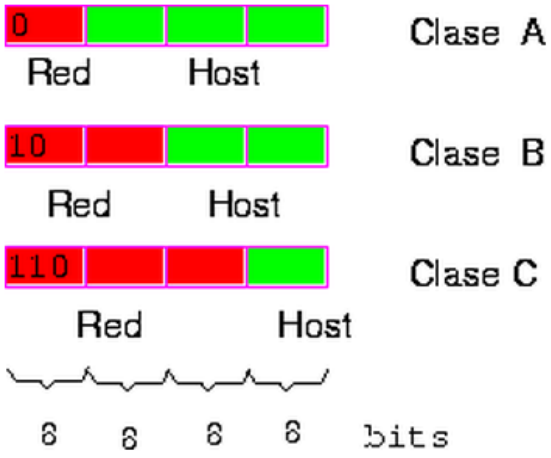
- Si el datagrama va dirigido a un host de esta red local, la dirección IP y la dirección física serán de la misma máquina.
- Sin embargo, si el datagrama va dirigido a un host de otra red, la dirección física corresponderá a la del router de esta red y no estará relacionada con el destino.

TRADUCCIÓN A DIRECCIÓN FÍSICA (9) IPv6

- En Ipv6 se busca eliminar completamente ARP y *broadcasts*
- 64 bits se reservan para la dirección de host, por lo que la MAC ethernet cabe completa ahí dentro
- O se usa multicast para evitar molestar a toda la red (ver ICMPv6 más adelante)

CIDR: SUB REDES (1)

- Recordemos que inicialmente se dividieron las direcciones IPv4 en 3 clases: Clase A, Clase B y Clase C.
- Cada clase tenía un número distinto de bytes asignado para la dirección de la Red y la dirección del Host.



SUB REDES (2)

- Con clase C se teníaun byte para el host, o sea pueden haber 256 computadores en esa red.
- Con clase B por otro lado se tenían 2 bytes para el host, es decir, 65.536 hosts.
- Como obviamente 256 computadores es poco para la mayoría de las organizaciones, casi todas pedían una red clase B.

SUB REDES (3)

- Por otro lado 65.536 computadores era mucho para una sola red. Para dividir una red corporativa en múltiples redes internas, se usa el concepto de una sub-red, que considera los primeros bits de la parte host, como una extensión de la parte red, y los últimos como el verdadero host.
- O sea, en vez de dividir la dirección en red, host; la dividimos en red, sub-red, host.

SUB REDES (4)

- El número de bits en la sub-red es variable, y se define con una máscara de red, que es un conjunto de bits.
- Los bits en 1 definen el prefijo de red (red + sub-red) y los bits en 0 corresponden a la parte host.
  - Por ejemplo, la clase B 146.83.0.0 la podemos dividir en redes con un máximo de 128 hosts cada una.
  - Para esto el primer byte completo y un bit más de la parte host de la clase B son necesarios para el prefijo de red. La máscara es entonces: 255.255.255.128.

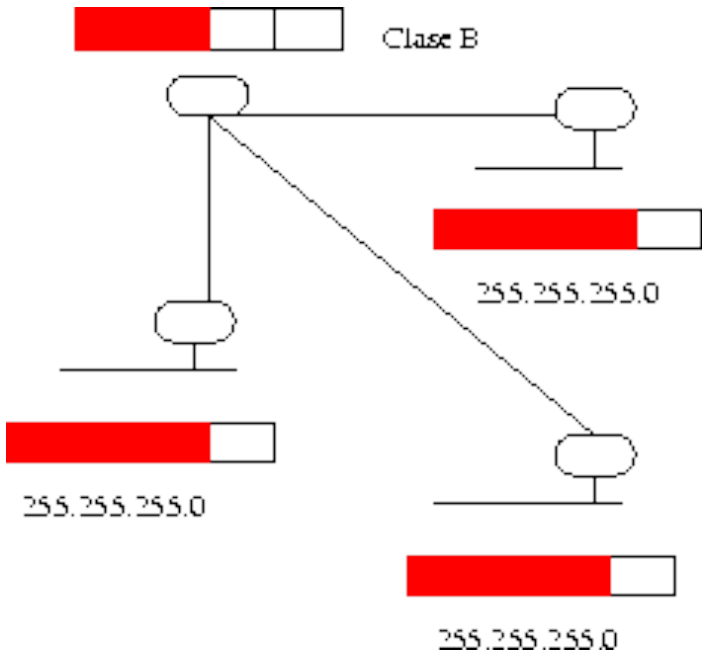
SUB REDES (5)

- Esto me permite manejar la red como una sola entrada en las tablas de rutas de Internet, pero internamente sé que son múltiples redes interconectadas.
- La máscara de sub-red no requiere ser conocida fuera de la red local, puesto que hacia afuera somos una sola red.

SUB REDES (6)

- Para separar la parte red y la parte host, entonces, primero uso el prefijo de la clase. Si corresponde a mi red, uso la máscara para terminar de extraer el prefijo de red.

Ejemplo de Sub-redes



SÚPER REDES (1)

- En el año 1992 se generó un importante proyecto de cambios en el manejo y asignación de direcciones IP en Internet.
- Básicamente, existían tres riesgos de muerte del Internet a mediano plazo:
  - 1.- Término de las direcciones clase B.
    - A esa altura casi la mitad de las direcciones clase B estaban asignadas. Al ritmo de crecimiento de ese momento, iban a terminarse en un par de años.
    - Básicamente, toda organización normal, requería de una clase B, puesto que una clase C era insuficiente.

SÚPER REDES (2)

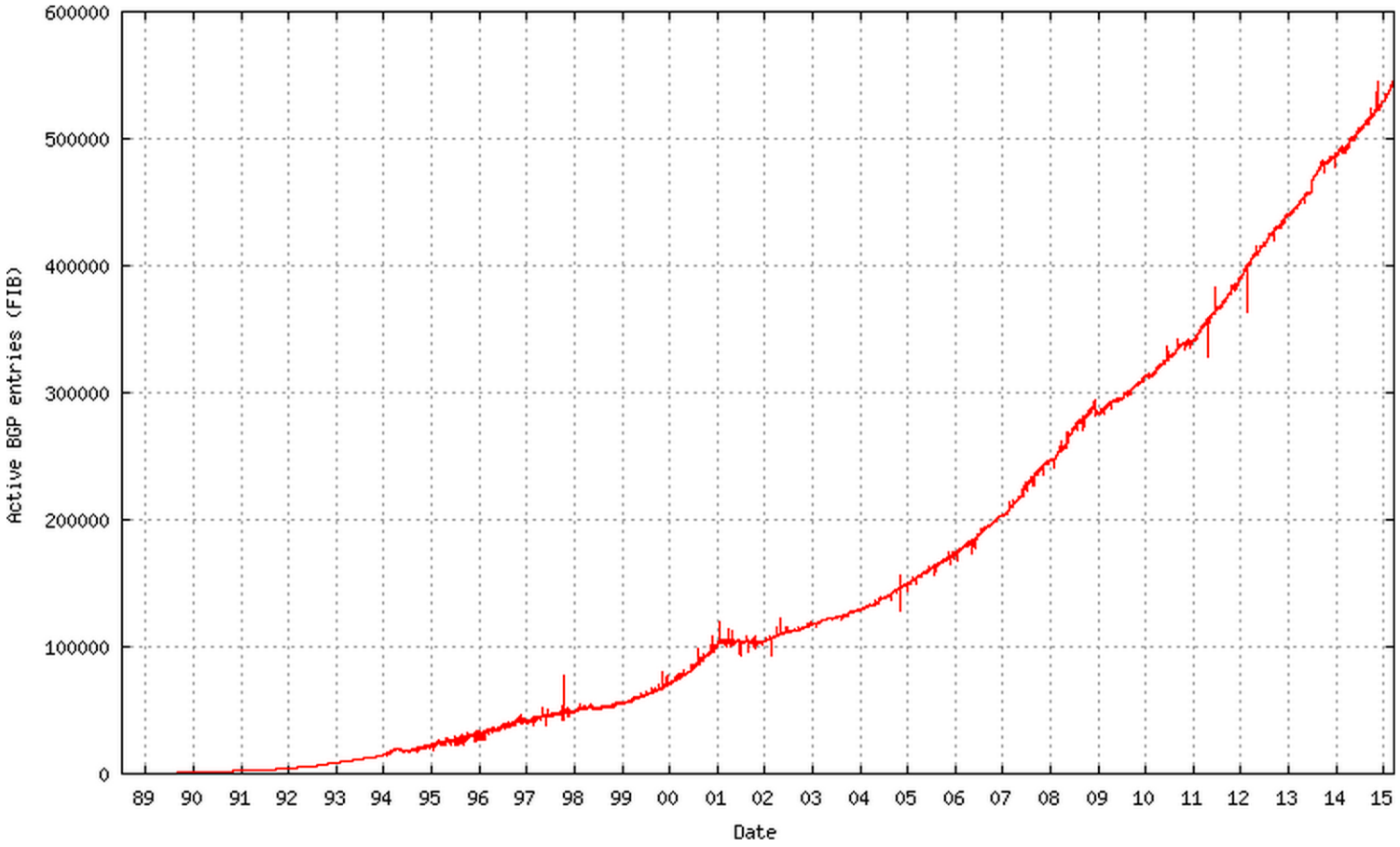
- 2.- Término de todas las direcciones IP
  - Claramente 32 bits no eran suficientes para el crecimiento de Internet.
  - La tasa de pérdida que es parte de cualquier sistema de asignación que se utilice, dado que son prefijos de red, y las redes no están nunca totalmente utilizadas.

SÚPER REDES (3)

- 3.- Explosión en las tablas de rutas
  - Los routers centrales de Internet deben mantener una tabla con una entrada para cada red conectada a Internet en el mundo entero.
  - Actualmente, hay unas 500.000 redes, y cada vez resulta más difícil manejar ese tamaño, tanto en memoria como en ancho de banda para los protocolos de actualización.
  - Se duplica cada 5 años aproximadamente
  - IPv6 va recién en 22.000 prefijos

SÚPER REDES (3)

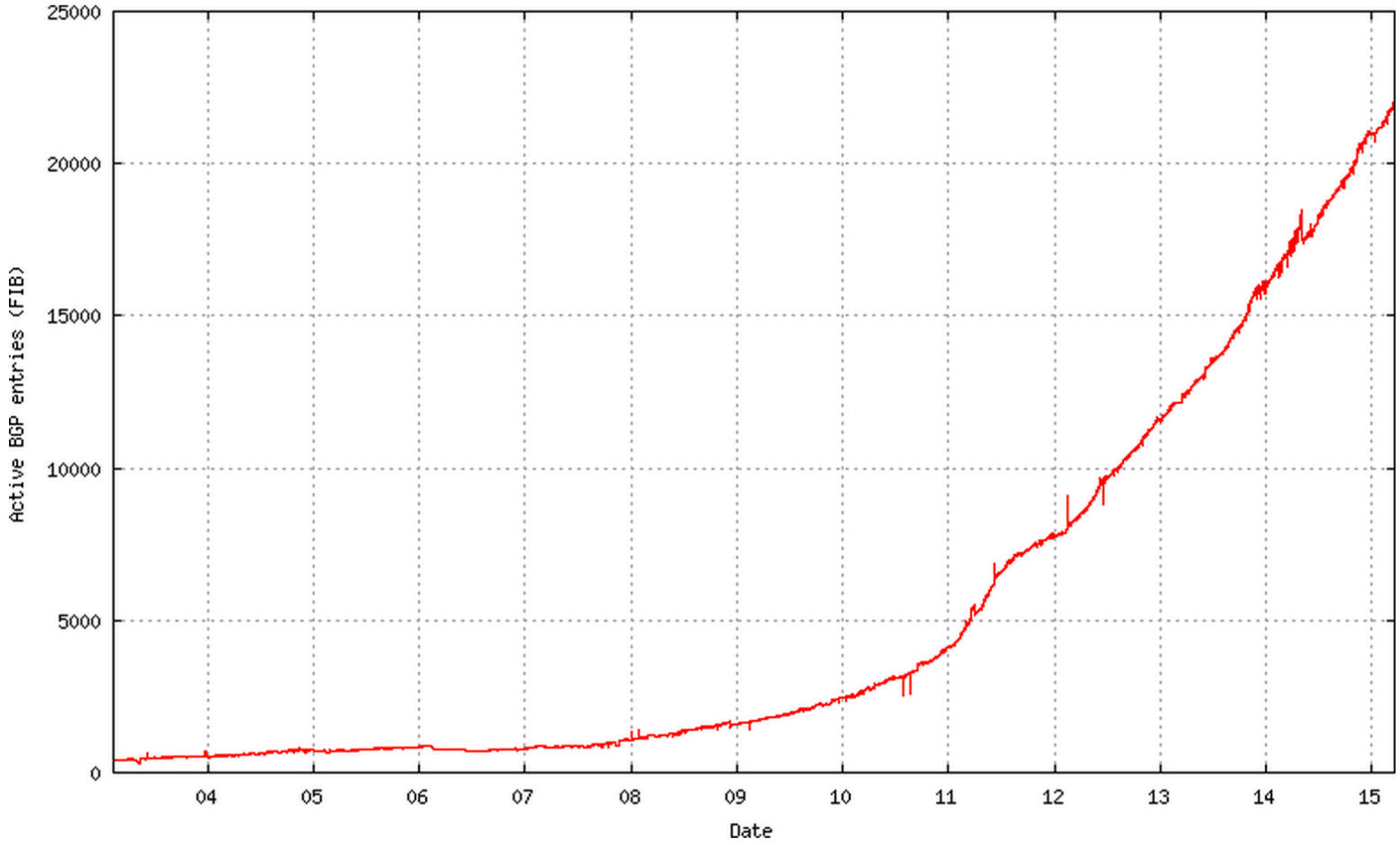
IPv4



http://bgp.potaroo.net/as2.0/bgp-active.html

SÚPER REDES (3)

IPv6



http://bgp.potaroo.net/v6/as2.0/index.html

SÚPER REDES (4)

- Para resolver los problemas 1 y 3 se decidió desplegar un nuevo esquema de asignación y de manejo de las direcciones IP, de modo de disminuir el problema y darle tiempo al desarrollo de la nueva versión de IP (IP versión 6) que resolvería el punto 2.
- Básicamente, una red clase A, B o C puede verse como una red con una máscara implícita.
- En el nuevo esquema, conocido como CIDR (Class-less IP), todas las redes se manejan con una máscara explícita para poder dividirlas en red/host y nos olvidamos de las clases.

SÚPER REDES (5)

- Por ello, podemos agrupar varias clases C contiguas, con una máscara común, extendiendo los bits de host hacia los de red, implementando lo que se conoce como súper-redes.
  - Por ejemplo, las clases C 200.0.0.0 y 200.0.1.0 pueden agruparse en una súper-red 200.0.0.0 con máscara 255.255.254.0.
- Actualmente, las máscaras ya no se anotan así, y se prefiere escribir el número de bits del prefijo de red. En el ejemplo anterior, se habla de la red 200.0.0.0/23.
- Clase A: /8, clase B: /16, clase C: /24 (/32=1IP)
- IPv6: soporta hasta un /128 = 1IP

SÚPER REDES (6)

- La existencia de súper-redes ha permitido asignar el número de redes adecuado a cada institución sin sobrevender demasiado, disminuyendo el problema 1.
- Sin embargo, si las súper-redes no son entendidas por los routers de Internet, exacerban el problema 3.
- Por ello, los nuevos protocolos de ruteo ya manejan este concepto, difundiendo redes agregadas, con un prefijo y una máscara.
- Esto me permite factorizar varias redes clases C agrupadas en una súper-red, como una sola entrada en mi tabla de rutas.

SÚPER REDES (7)

- Para poder manejar super-redes, sin embargo, se requiere un cambio mayor en la representación de las tablas de ruteo porque (al contrario que en el caso de las sub-redes) ahora requiero conocer la máscara de red en todas partes, incluso fuera de la red misma.
- En IP sin clases (CIDR) se supone que toda máquina y Router manejan tablas con el prefijo de red y la máscara asociada, de modo de poder separar la parte red y la parte host de una dirección cualquiera.

SÚPER REDES (8)

- Un punto importante es que aunque mi Router no hable CIDR y utilice el sistema antiguo igual funciona.
- Esto se logra haciendo que ese Router maneje una entrada para cada clase C de la súper-red en cuestión, sin factorización.
- Obviamente, perdemos la ventaja de CIDR, pero al menos funciona.
- En IPv6 no hay opción: debo tener la máscara

SÚPER REDES (9)

- IPv6 usa el mismo esquema de prefijos
- NIC Chile prefix: 2001:1398::/32
- Ejemplo host:

b.nic.cl → 2001:1398:274:0:200:7:4:7

Redes Privadas

- ¿Cómo crecer en IPv4 sin usar más direcciones?
- Armar redes internas IP (Intranets) usando direcciones cualesquiera
- Anunciar hacia fuera sólo una IP pública
- Poner todos los servicios en esa IP
- Traducir direcciones en el router de acceso (NAT)
- RFC1918: 10/8, 172.16/12, 192.168/16 reservadas
- ¡No tenemos idea cuántas redes privadas hay!

Redes Privadas (2)

- Necesitan un router que haga NAT
- Todo equipo actual sabe hacerlo
- Incluso en la casa de Uds
- Mala idea usar redes 10.0.0.0 o 192.168.1
- ¡Todo el mundo las usa!
- Esto ha permitido aguantar hasta hoy
- Casi sin usar IPs públicas v4

Fin de Ipv4

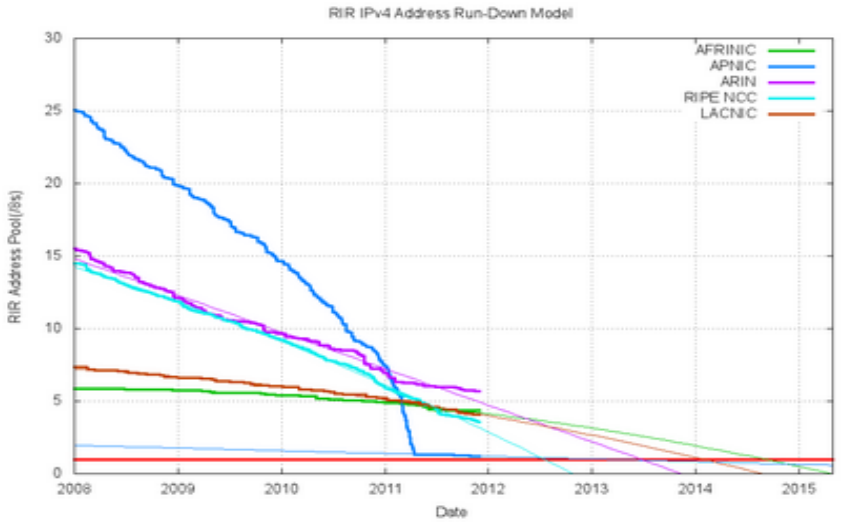
This report generated at 30-Nov-2011 07:59 UTC.

IANA Unallocated Address Pool Exhaustion:

03-Feb-2011

Projected RIR Address Pool Exhaustion Dates:

RIR	Projected Exhaustion Date	Remaining Addresses in RIR Pool (/8s)
APNIC:	**19-Apr-2011**	1.2017
RIPENCC:	**14-Jul-2012**	3.5721
ARIN:	**22-Jun-2013**	5.7067
LACNIC:	**28-Jan-2014**	4.1000
AFRINIC:	**03-Sep-2014**	4.3643



Projection of consumption of Remaining RIR Address Pools

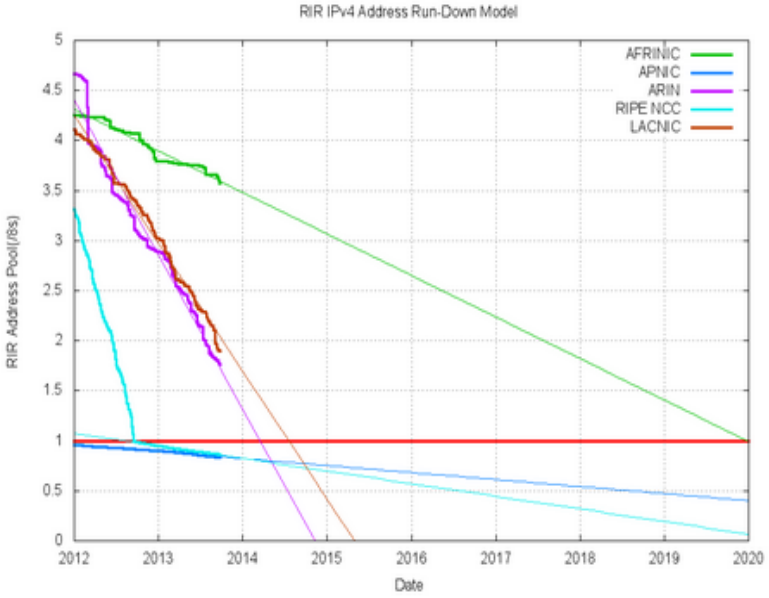
Fin de Ipv4

This report generated at 25-Sep-2013 08:21 UTC.

IANA Unallocated Address Pool Exhaustion:
03-Feb-2011

Projected RIR Address Pool Exhaustion Dates:

RIR	Projected Exhaustion Date	Remaining Addresses in RIR Pool (/8s)
APNIC:	**19-Apr-2011** (actual)	0.8329
RIPE NCC:	**14-Sep-2012** (actual)	0.8633
ARIN:	**09-Jan-2015**	1.7637
LACNIC:	**19-Apr-2015**	1.9011
AFRINIC:	**26-Aug-2022**	3.5705



Projection of consumption of Remaining RIR Address Pools

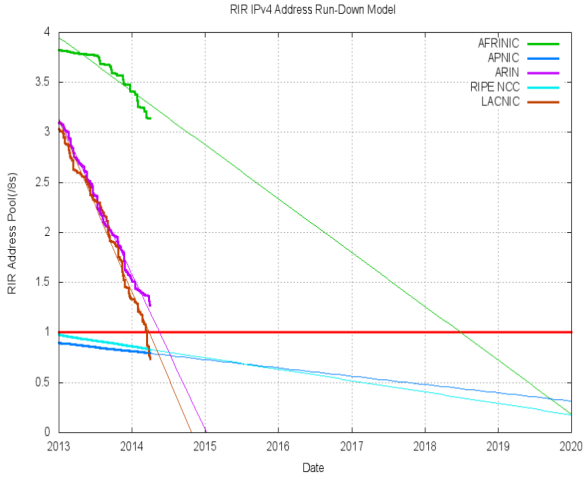
IPv4 Address Report

This report generated at 01-Apr-2014 08:4

IANA Unallocated Address Pool Exhaustion:
03-Feb-2011

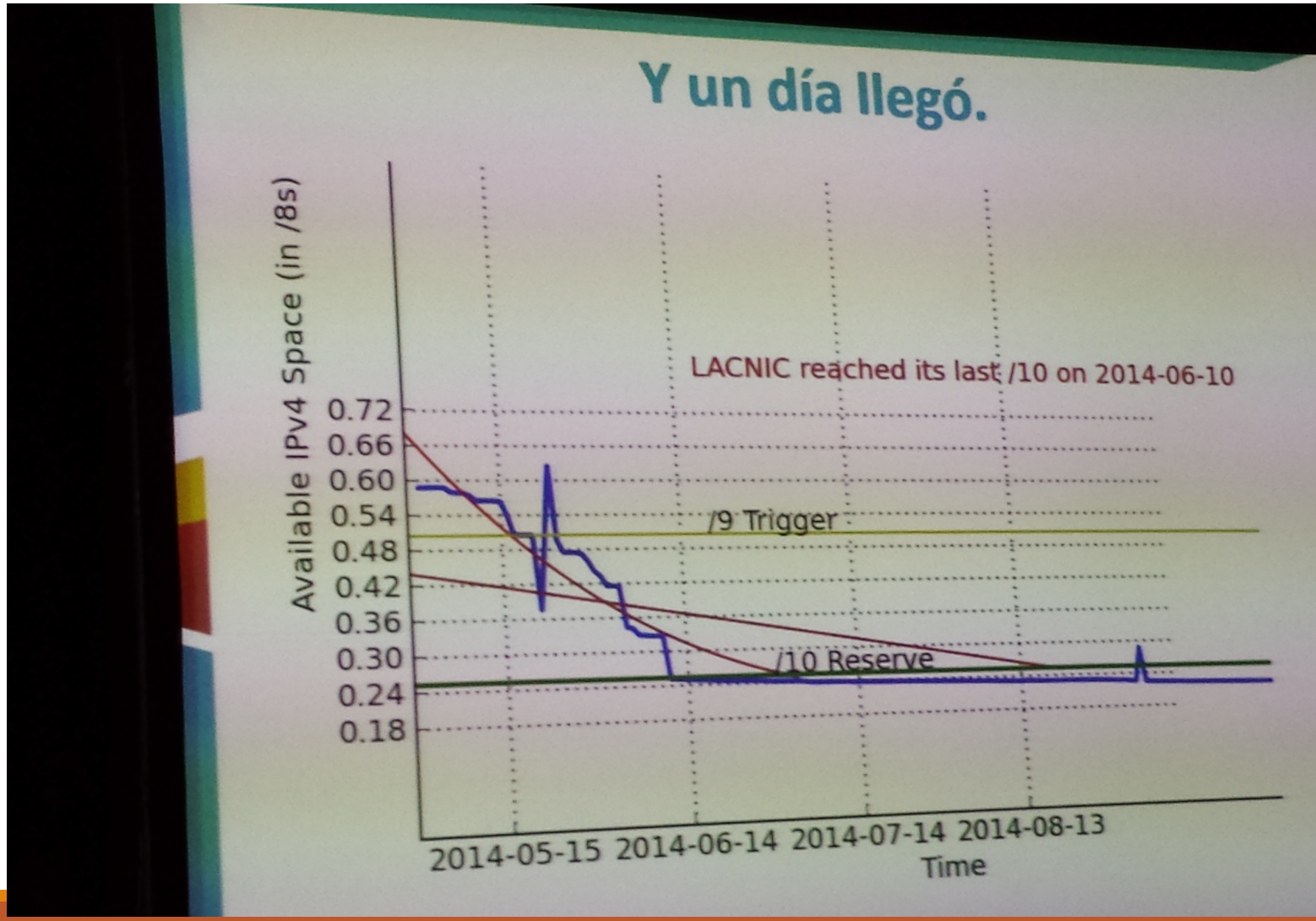
Projected RIR Address Pool Exhaustion Dates:

RIR	Projected Exhaustion Date	Remaining Addresses in RIR Pool (/8s)
APNIC:	**19-Apr-2011** (actual)	0.7939
RIPE NCC:	**14-Sep-2012** (actual)	0.8277
LACNIC:	**16-Sep-2014**	0.7359
ARIN:	**19-Mar-2015**	1.2719
AFRINIC:	**29-Apr-2020**	3.1418



Projection of consumption of Remaining RIR Address Pools

Fin de Ipv4


Y un día llegó.
LACNIC reached its last /10 on 2014-06-10
Available IPv4 Space (in /8s)
0.72
0.66
0.60
0.54
0.48
0.42
0.36
0.30
0.24
0.18
/9 Trigger
/10 Reserve
2014-05-15
2014-06-14
2014-07-14
2014-08-13
Time

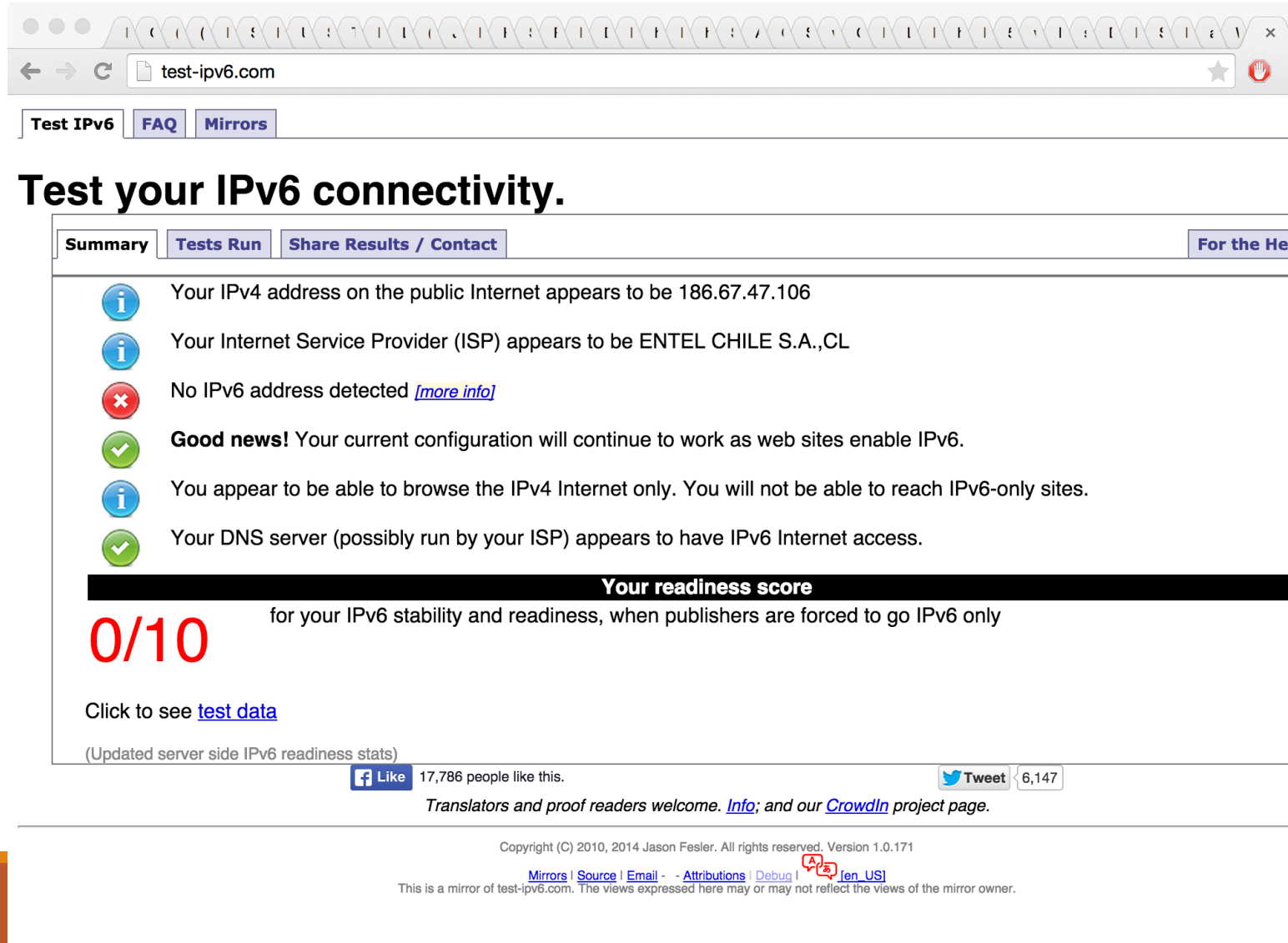
Muerte de Internet

- Ahora sí que murió
- CIDR nos dio tiempo desde 1992 hasta 2015!
- Ya no quedan direcciones IPv4
- ISPs tienen stock para un par de años más
- Es la hora de IPv6 (¡al fin!)

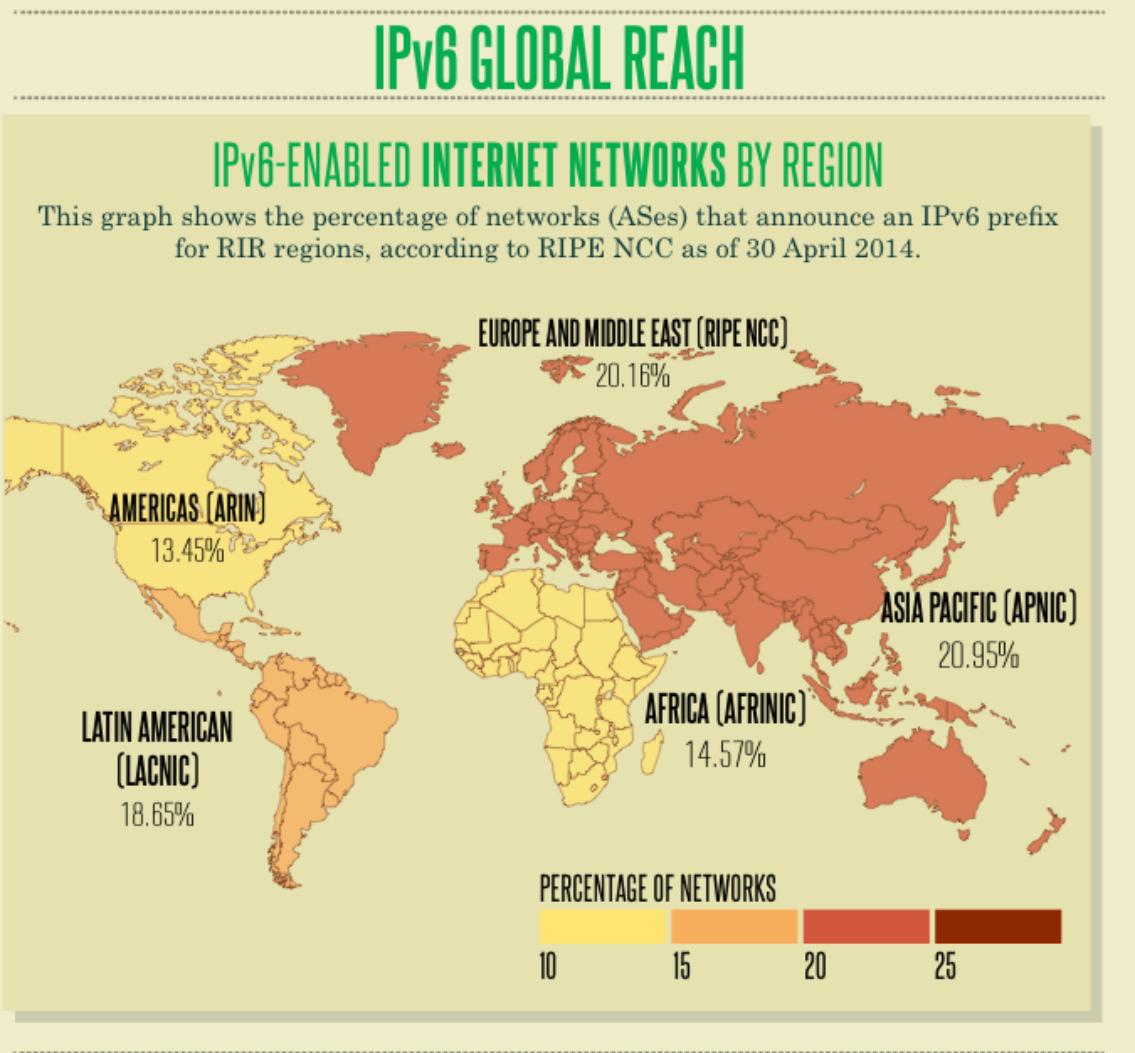
Despliegue en el mundo

- Junio 2012: IPv6 day
- Hoy: va creciendo un poco más rápido de lo esperado
- 2018: se espera que IPv6 sea el protocolo dominante en Internet (1995 dije: 2005!)
- Sabemos vivir en ambos mundos
- Pronto nos tocará: servidores sólo en IPv6 (http://ipv6.google.com)
- Ver: http://test-ipv6.com

Despliegue en el mundo



Despliegue en el mundo



Conclusiones

- Se necesitará mucho conocimiento en redes
- Poca gente entiende la complejidad de esto
- Y será urgente pronto
- Difícil que haya una transición sin sobresaltos
- Hay que estudiar IPv4 e IPv6 en igualdad de condiciones hoy
- Y probar configuraciones posibles
- Los ISPs no dan el servicio fácilmente hoy (pídanlo en sus empresas)

PAQUETES IP

- Datagrama
- Header
- Datagrama IPv4
- Datagrama IPv6
- MTU de la Red
- MTU Path Discovery

DATAGRAMA(1)

- Los datos se empaquetan en un datagrama, que es la unidad utilizada para atravesar las redes en camino.
- La idea básica de un datagrama es equivalente a una carta envuelta en un sobre.
- Los datos del sobre van en el header del paquete y el contenido va como datos.
- Al igual que en la carta, la idea es que al irse ruteando por la red el datagrama queda intacto, sin modificarse ni el header (casi) ni el contenido.

DATAGRAMA(2)

- Cada datagrama es independiente, por lo cual pueden rutearse por caminos distintos.
- Por otro lado, IP provee un servicio de “mejor esfuerzo”, es decir no garantiza la entrega.
- Los paquetes pueden llegar a su destino desordenados, duplicados, alterados o incluso perderse.

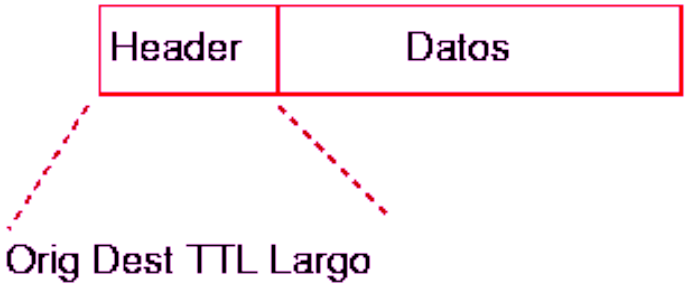
DATAGRAMA(1)

- En resumen los datagramas IP son:
  - paquetes de datos auto contenidos.
  - Independientes.
  - Auto-ruteables.
  - Sin manejo de estado en los routers.
  - Sin conexiones.
- Los datagramas IP se dividen en dos partes
  - Header (Encabezamiento).
  - Datos.

HEADER (1)

- En el Header se encuentra toda la información relevante para rutear el paquete a través de la red, los datos son sólo importantes para la aplicación que los recibe.

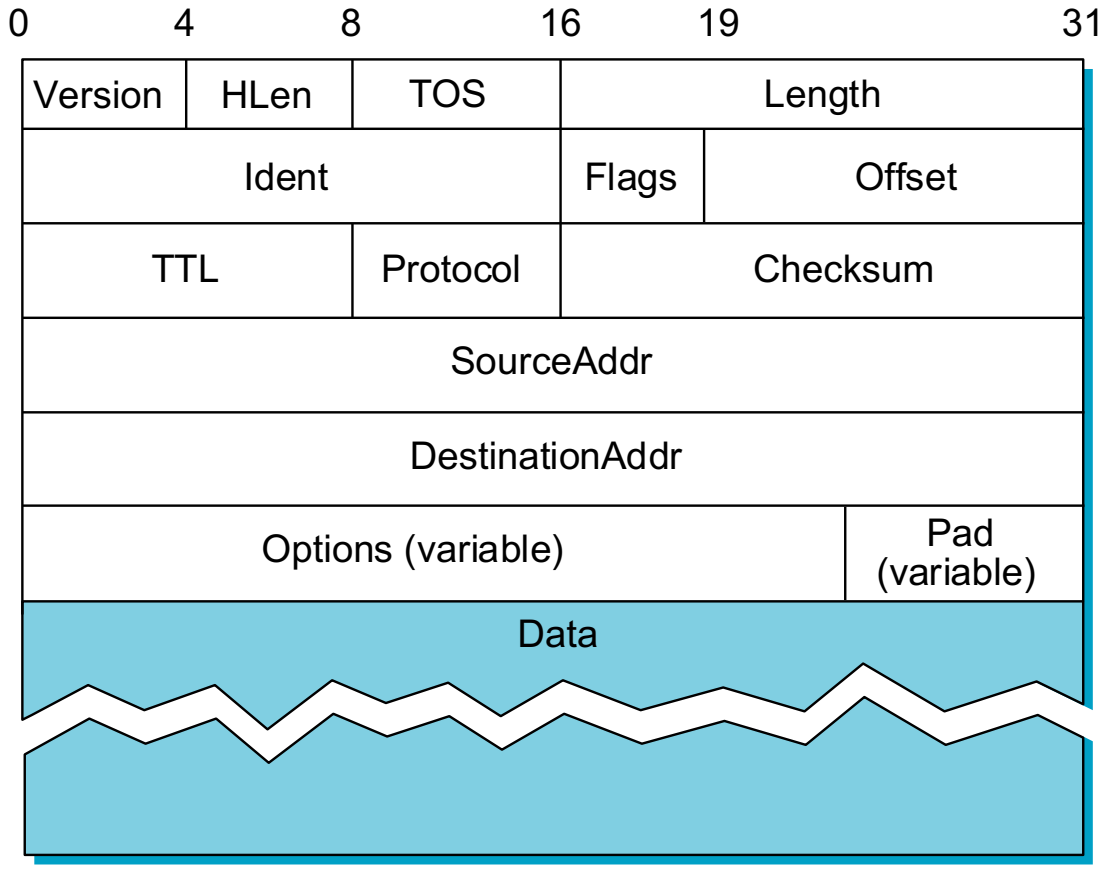
Ejemplo de Datagrama sin chequeo de datos.



HEADER (2)

- El encabezamiento contiene las direcciones IP del origen y del destino, así como el largo y un checksum del header. La parte datos no se valida, por lo que el nivel trasporte tendrá que encargarse de ellos.
- Todos los campos del encabezamiento se representan en forma estándar, conocida como *network order*.
  - En alguna máquinas, deberemos traducir los enteros para llevarlos la representación correcta para esa arquitectura.

DATAGRAMA IPV4 (1)



DATAGRAMA IPV4 (2)

- VERS: Contiene la versión del Protocolo IP. Las versiones más usadas son la 4 y la 6. La 5 es una versión experimental.
- HLEN: El largo del header IP contado en unidades de 32-bit. No incluye el campo de los datos.
- TOS: El tipo de servicio es una indicación de la calidad del servicio que se pide para este datagrama IP.

DATAGRAMA IPV4 (3)

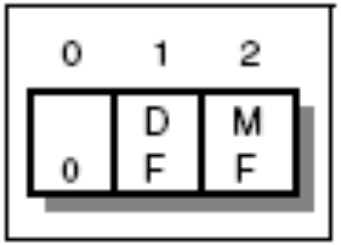
- TOS: Además contiene los siguientes campos:
  - Precedence: Indica la naturaleza y prioridad del datagrama:
    - 000: Routine
    - 001: Priority
    - 010: Immediate
    - 011: Flash
    - 100: Flash override
    - 101: Critical
    - 110: Internetwork control
    - 111: Network control

DATAGRAMA IPV4 (4)

- TOS: Además contiene los siguientes campos:
  - TOS: Especifica el valor del tipo de servicio:
    - 1000: Minimize delay
    - 0100: Maximize throughput
    - 0010: Maximize reliability
    - 0001: Minimize monetary cost
    - 0000: Normal service
  - MBZ: Reservado para uso futuro.

DATAGRAMA IPV4 (5)

- Flags: Este campo contiene flags de control:
  - 0: Reservado, debe ser cero.
  - DF (Do not Fragment): 0 significa que se permite fragmentación; 1 significa que el datagrama no se puede fragmentar.
  - MF (More Fragments): 0 significa que este es el último fragmento del datagrama; 1 significa que más fragmentos siguen al datagrama.



DATAGRAMA IPV4 (6)

- Fragment Offset
  - Es emitido para ayudar a re-ensamblar el datagrama completo. El valor de este campo contiene el número de segmentos de 64-bits contenidos en fragmentos anteriores (los bytes del header no cuentan). Si este es el primer segmento este campo toma el valor de cero.

DATAGRAMA IPV4 (7)

- Time to Live
  - Este campo especifica el tiempo (en segundos) que el datagrama tiene permitido para viajar por la red.
  - Más adelante se profundizará en su funcionamiento.

DATAGRAMA IPV4 (8)

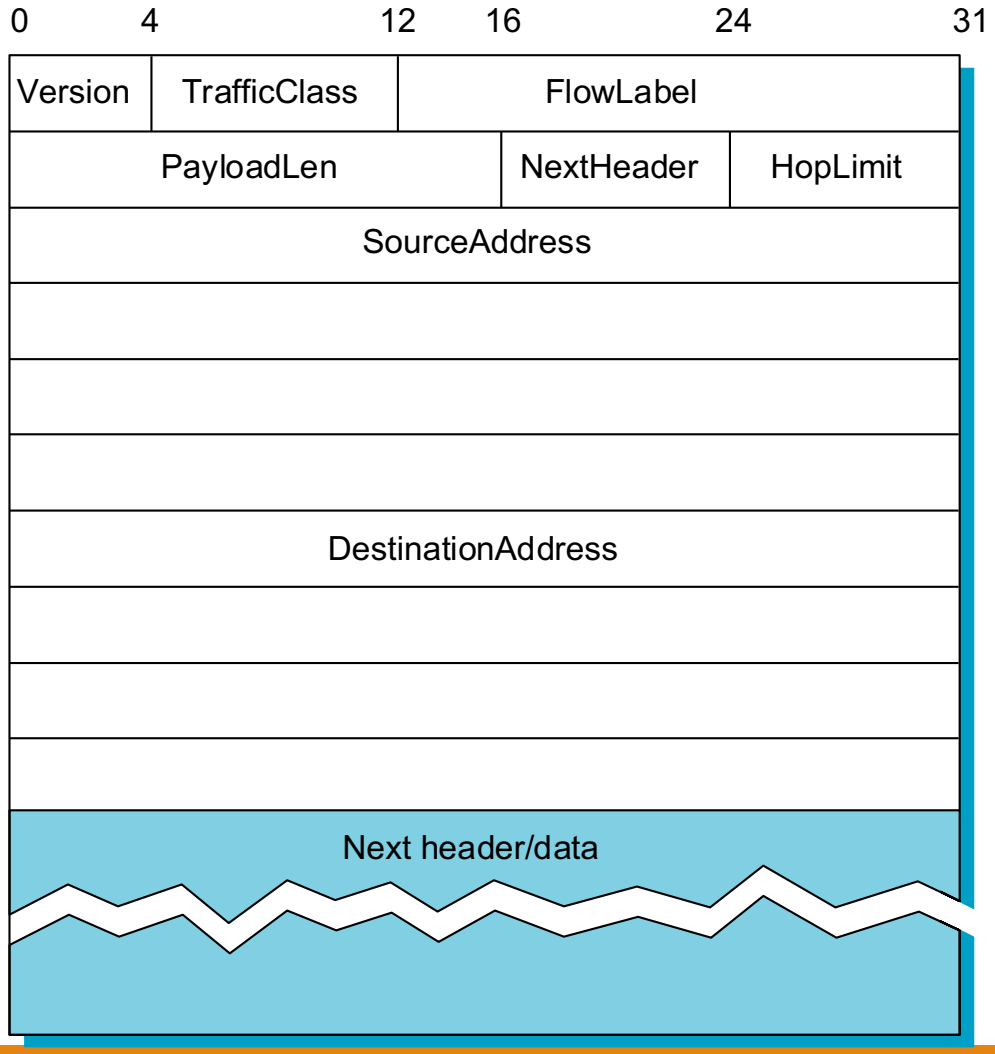
- Header Checksum
  - Este campo contiene el checksum de la información contenida en el encabezado. Si la checksum del header no concuerda con los contenidos de éste, el datagrama es descartado.
- Source IP Address
  - La dirección de 32-bits del host que envió este datagrama.
- Destination IP Address
  - La dirección de 32-bits del host de destino de este datagrama.

DATAGRAMA IPV4 (9)

- Algunas Opciones IP

Option	Description
Security	Specifies how secret the datagram is
Strict source routing	Gives the complete path to be followed
Loose source routing	Gives a list of routers not to be missed
Record route	Makes each router append its IP address
Timestamp	Makes each router append its address and timestamp

DATAGRAMA IPV6 (1)



DATAGRAMA IPV6 (2)

- El campo *Version* ocupa la misma posición.
- *TrafficClass* y *FlowLabel* están relacionados con QoS. *TrafficClass* es equivalente al header *ToS* de IPv4 y *FlowLabel* es un número que identifica una secuencia de paquetes que van un origen a un destino y que deben ser “tratados” de cierta manera.
- *PayloadLen* indica el largo del paquete descontando el largo del header.

DATAGRAMA IPV6 (3)

- El campo *NextHeader* indica la existencia de opciones IP, por ejemplo la fragmentación es una opción más. Si no hay opciones, el campo indica el protocolo de nivel superior (como el campo *Protocol*).
- *HopLimit* equivale al campo TTL de IPv4.
- El header IPv6 es de tamaño fijo (40 bytes).

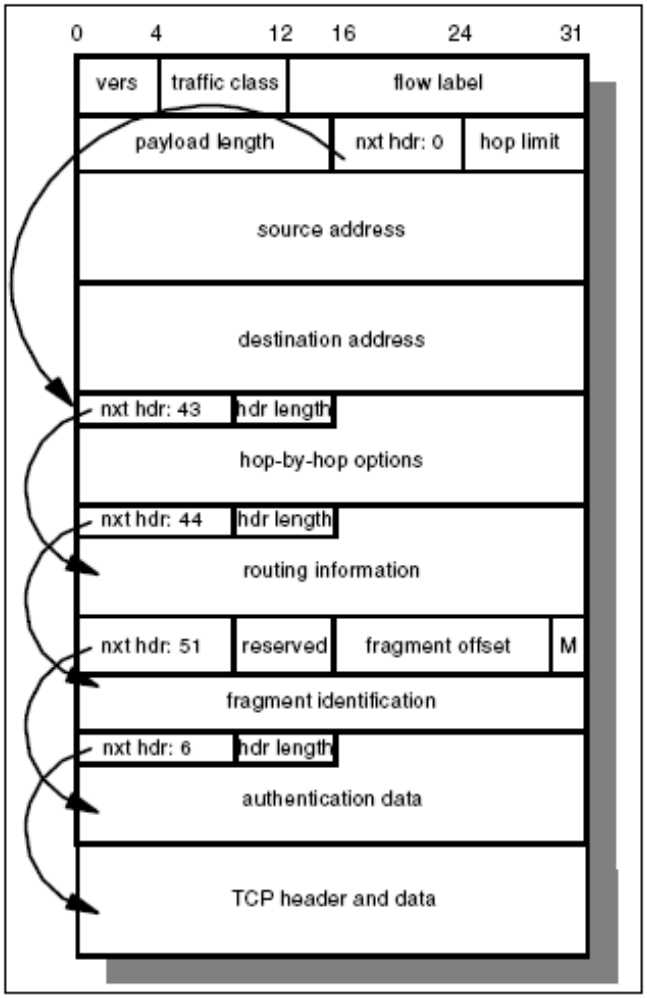


Figure 220. IPv6 packet containing multiple extension headers

JUBOGRAMAS IPV6

- El campo *NextHeader* indica que este es un Jumbograma y el payload length == 0
- El largo va en 32 bits en el header opcional
- Requiere que TCP y UDP cambien un poco
- Supercomputing applications

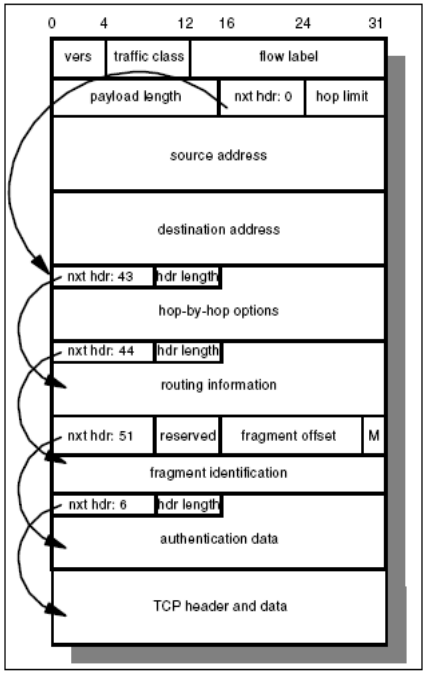


Figure 220. IPv6 packet containing multiple extension headers

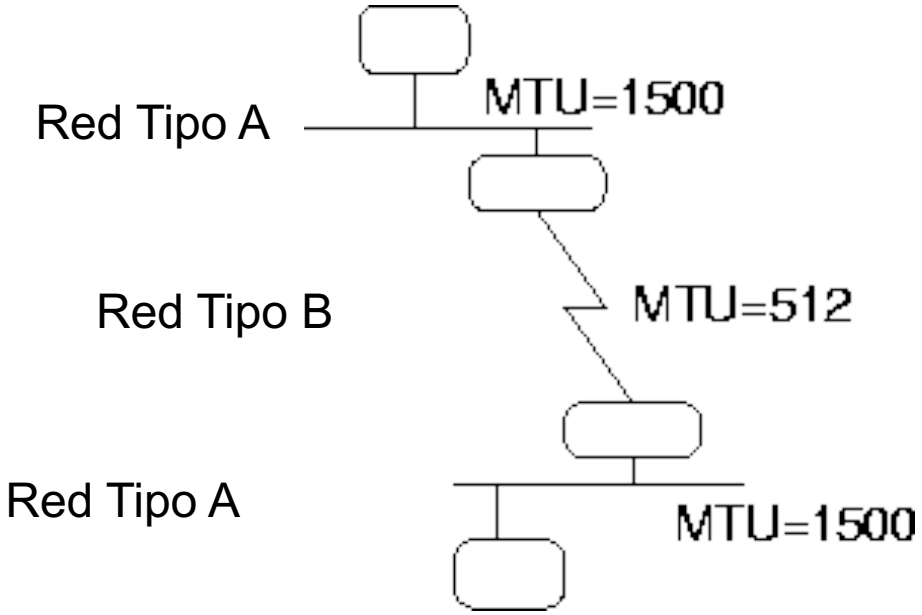
	Additional header length	option code (datagram size)	size a 4 Byte number
Next header	0	194	4
Jumbo payload length			

MTU DE LA RED (1)

- Antes de enviar un datagrama es necesario determinar su tamaño.
- Obviamente quiero que sea lo más grande posible, pero debe caber en un frame físico.
- Las redes por las que transitará pueden tener tamaños de frames (MTU: Maximum Transfer Unit) distintos.
- Problema para los protocolos de transporte tipo TCP y UDP

MTU DE LA RED (2)

- Si un paquete llega a un router y es muy grande para seguir su camino debo ***fragmentarlo*** en unidades más pequeñas. (detalles luego)



MANEJO DE ERRORES

- Manejo de Errores
- El ICMP (IPv4)
- ICMPv6 (Ipv6) y Multicast

MANEJO DE ERRORES

- Al detectarse un error relacionado con un datagrama, se envía un mensaje de error a la dirección IP de origen.
- Este mensaje va en un datagrama dirigido al layer IP propiamente tal, no a una aplicación de nivel superior.
- Por ello, se encapsula en un datagrama IP con valor protocolo (en el header) de ICMP. El datagrama original (que causó el error) va como dato
- Típicos paquetes de error son porque el TTL llegó a cero, porque no existen rutas a esa red, tiempo esperando fragmentos excedido, etc.

ICMP (1)

- Internet Control Message Protocol
  - Protocolo integral de IP, utilizado para reportar errores.
  - Utiliza a IP como un protocolo de capa “inferior”.
  - ICMP no se puede usar para reportar errores de mensajes ICMP.

ICMP (2)

- Internet Control Message Protocol
  - En el caso de los fragmentos, sólo se generan mensajes ICMP para el primer fragmento.
  - Nunca se genera ICMP para datagramas con direcciones de destino broadcast o multicast, o direcciones de origen que no sean únicas.

ICMP (3)

- Los mensajes ICMP tienen un tipo y un código asociado (que depende del tipo). Algunos mensajes además contienen datos adicionales.
  - 0: Echo reply
  - 3: Destination unreachable
  - 4: Source quench
  - 5: Redirect
  - 8: Echo
  - 9: Router advertisement

ICMP (4)

- Los mensajes ICMP tienen un tipo y un código asociado (que depende del tipo). Algunos mensajes además contienen datos adicionales.
  - 10: Router solicitation
  - 11: Time exceeded
  - 12: Parameter problem
  - 13: Timestamp request
  - 14: Timestamp reply
  - 30: Traceroute
  - 32: Mobile host redirect

ICMP (5)

- Para los mensajes de tipo==3, tenemos los siguientes códigos:
  - 0: Network unreachable
  - 1: Host unreachable
  - 2: Protocol unreachable
  - 3: Port unreachable
  - 4: Fragmentation needed but the Do Not Fragment bit was set
  - 5: Source route failed
  - 6: Destination network unknown

ICMP (6)

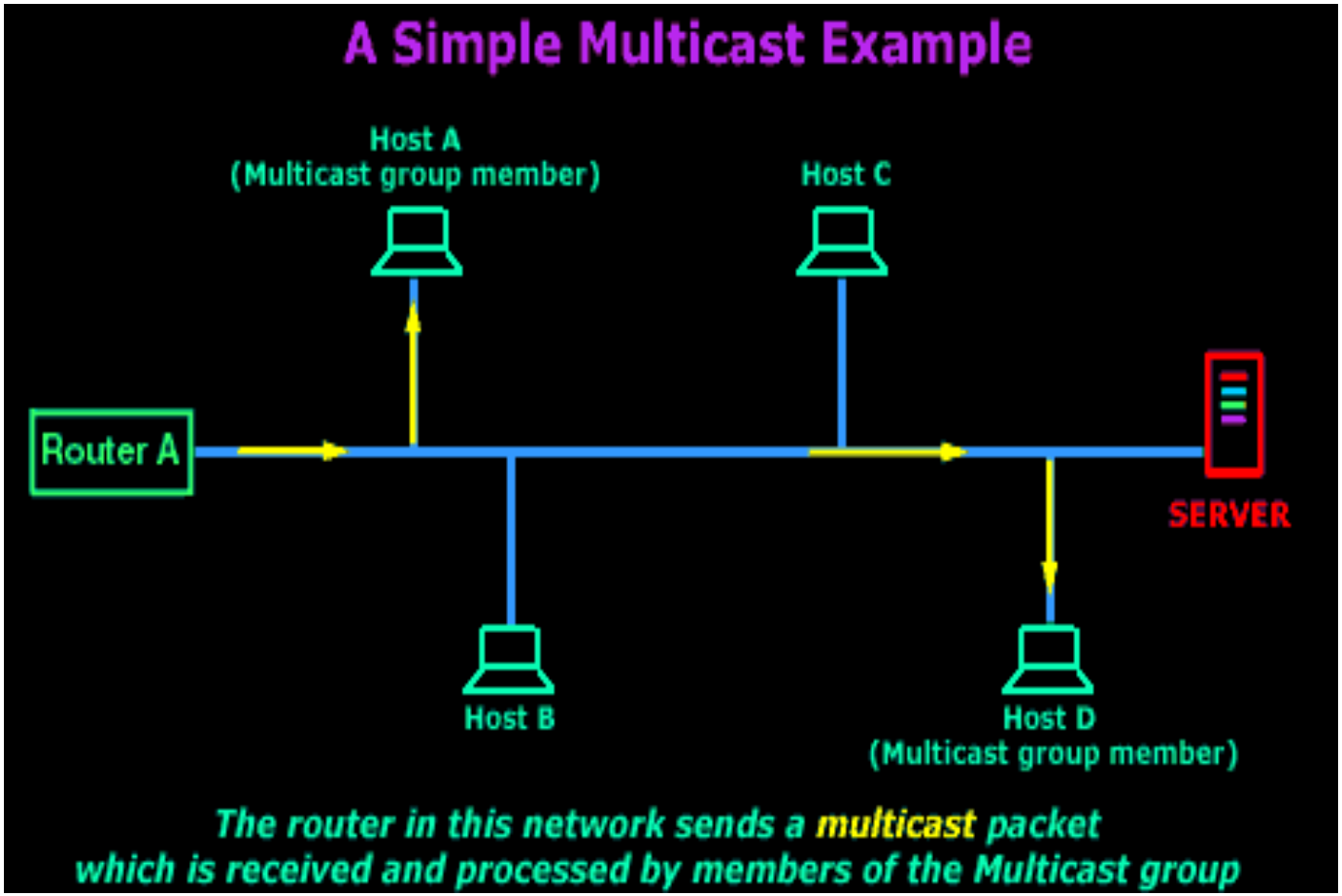
- Para los mensajes de tipo==3, tenemos los siguientes códigos:
  - 7: Destination host unknown
  - 9: Destination network administratively prohibited
  - 10: Destination host administratively prohibited
  - 11: Network unreachable for this type of service
  - 12: Host unreachable for this type of service
  - 13: Communication administratively prohibited by filtering

ICMPv6 (1)

- Multicast:
  - Uno a muchos (pero no todos, != broadcast)
  - Los computadores se “suscriben” a grupos
  - Se usa intensivamente en la red local
  - Pero se busca masificar para difusión (tipo canal de TV), y entonces es ruteable
  - IPv4: usa direcciones 224.0.0.0-239.255.255.255 (16 bloques /8 o clases A; llamadas clase D)
  - IPv6: usa ff00::/8
  - La conexión uno a uno ahora se llama “unicast”

ICMPv6 (2)

Multicast en Red Local



ICMPv6 (3)

- Multicast en ethernet:
  - En IPv4 se usan los 23 últimos bits
  - Se usa 01:00:5E:00:00:00 – 01:00:5E:7F:FF:FF
  - En IPv6 se usan los últimos 32 bits
  - Se usa 33:33:xx:xx:xx:xx
  - Genera colisiones, pero no muchas
  - Las tarjetas aceptan que el kernel les pida escuchar direcciones específicas de este rango

ICMPv6 (3)

- IPv6 usa multicast intensivamente (es obligatorio)
  - Principalmente en red local
  - ff02:: multicast local
  - ff02::1 todos los computadores (broadcast)
  - ff02::2 todos los routers en la red
  - ff02::1:2 Todos los servidores DHCPv6
  - ff02::1:ffxx:xxxx nodo solicitado (~ ARP)

ICMPv6 (4)

IPv6 usa ICMPv6 para traducción de direcciones (en vez de ARP)

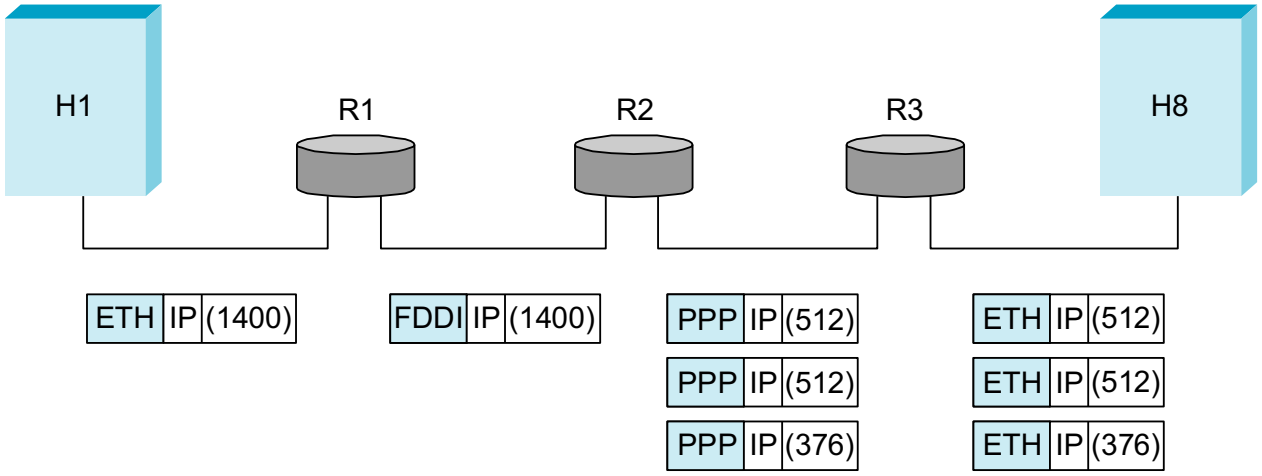
- Busco destino: 2037::01:800:20**0e:8c6c**
- Envío pregunta a: ff02::1:ff**0e:8c6c**
- Se envía a MAC: 33:33:**ff:0e:8c:6c**
- Toda tarjeta con dirección IPv6 pide recibir ese multicast (copiando sus últimos3 bytes)
- La probabilidad de “colisión” es baja y no molesta mucho (simplemente no respondo si no soy yo)

RUTEO Y FRAGMENTACIÓN

- Fragmentación y MTU Path Discovery
- Ruteo Básico
- TTL

FRAGMENTACIÓN (1)

- Como se vio anteriormente los paquetes pueden atravesar distintos tipos de redes.
- Cada una de éstas redes tiene un MTU determinado.
- Si un paquete llega a un router y es muy grande para seguir su camino debo fragmentarlo en unidades más pequeñas o rechazarlo

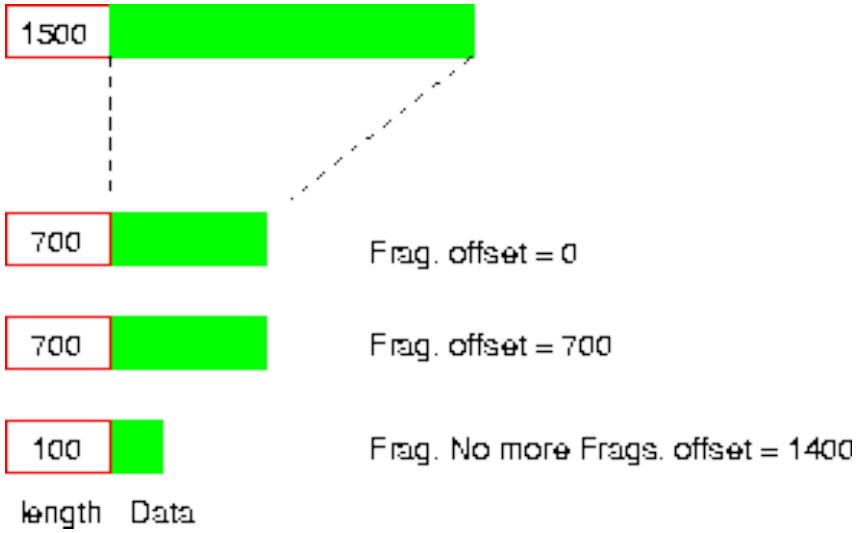


FRAGMENTACIÓN (2)

- Al fragmentar divido el datagrama en varios datagramas con (casi) el mismo header y los trozos de los datos en cada uno.
- El largo de cada datagrama es el que corresponde a cada fragmento.
- Cada uno de estos fragmentos será ruteado luego como un datagrama independiente.
- El problema es que si el nivel transporte envía un datagrama, el receptor debe recibir también uno (y no varios más pequeños).

FRAGMENTACIÓN (3)

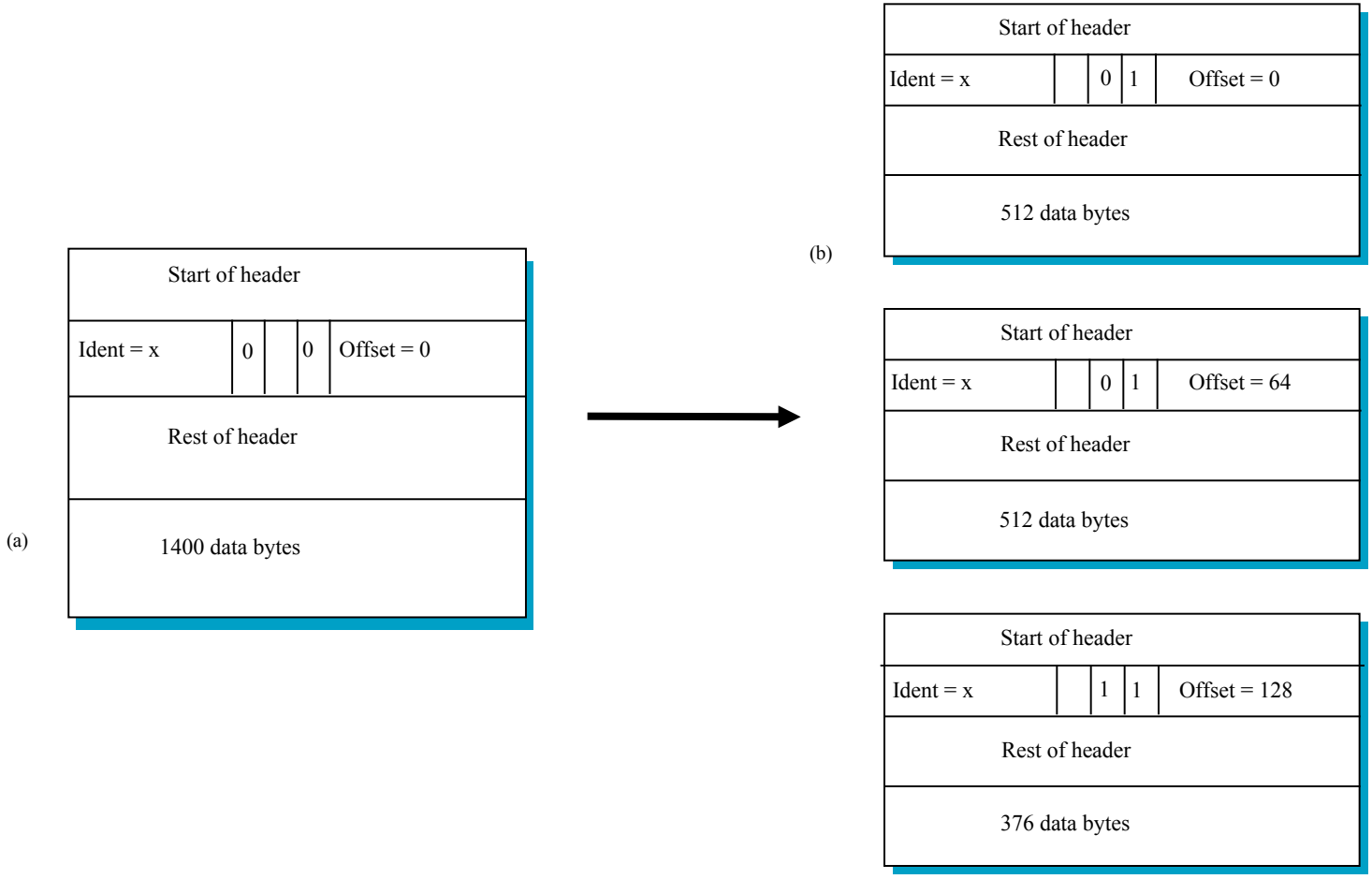
- Para esto, el receptor final “pega” los fragmentos para reconstruir el datagrama original.
- En esto se usan los otros campos del encabezado: el identificador es un valor único por cada datagrama enviado desde un mismo host.



FRAGMENTACIÓN (4)

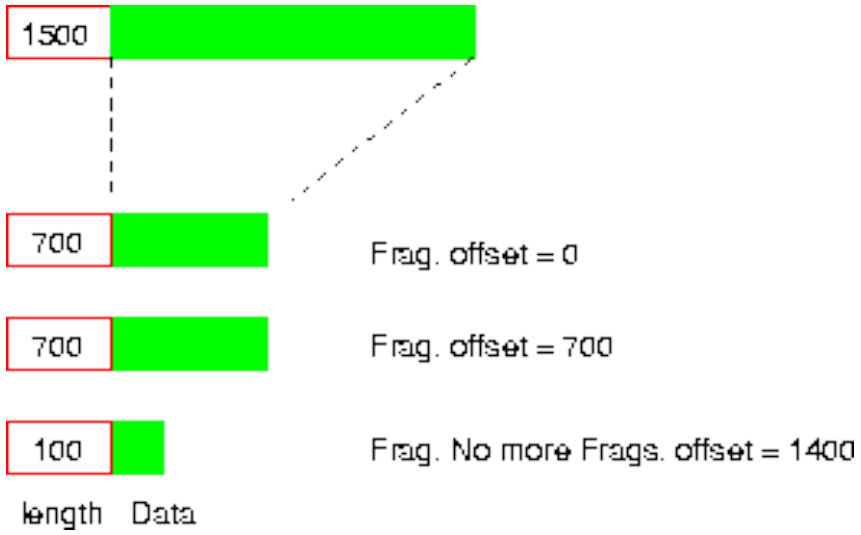
- *Recordar: IPv4 tiene espacio en el header siempre para esto, IPv6 tiene un header opcional que sólo va en los paquetes fragmentados*
- Los fragmentos llevan todos el identificador del datagrama original, permitiendo reconocerlos.
- El offset del fragmento indica la posición dentro del datagrama original donde van los datos de este fragmento.
  - Para ahorrarse espacio en el header, el offset va anotado contando de a 8 bytes, por lo que debe multiplicarse por ocho para obtener el verdadero valor.

FRAGMENTACIÓN (5)



FRAGMENTACIÓN (6)

- Finalmente, en el campo *Flags* se anota que son fragmentos y no un datagrama completo.
- Otro bit existe (*No more fragments*) para el último fragmento, de modo de saber cuándo terminar.



FRAGMENTACIÓN (7)

- Al recibir el primer fragmento de un datagrama, se activa un timer de modo que al transcurrir demasiado tiempo esperando armarlo lo descarto generando un error.
- Esto descarta el datagrama completo, aunque se hayan recibido varios fragmentos en el intertanto.

FRAGMENTACIÓN (8)

- Típicamente la implementación consiste en una lista enlazada de fragmentos para cada datagrama (identificado por su campo identificador), donde se van agregando los fragmentos a medida que llegan.
- Esto se hace así puesto que desconocemos el largo total del datagrama, hasta que no recibimos el último fragmento.
- Al fragmentar, debo re-calcular el chksum del hdr (IPv4)
- EN IPv6 los routers no PUEDEN fragmentar. Solo el origen.

MTU Path Discovery

- Uso en protocolo de transporte:
- Envío sólo paquetes con el bit de No Fragmentar
- Si recibo ICMP “Need to Fragment”
  - Adapto el MSS al nuevo tamaño (viene en el ICMP, o intento con uno más pequeño)
  - Si timeout: intento con uno más pequeño

- Siempre mantengo el bit de No Fragmentar
- Me adapto en caso de cambio de rutas
- Pero MTU Path nunca crece
- En IPv6 eliminamos fragmentación en ruta por esto mismo

RUTEO BÁSICO (1)

- Cada router y cada host mantiene una tabla de rutas, puesto que incluso un host conectado a una sola red debe saber cómo llegar a los distintos destinos.
- Todo el ruteo de un datagrama se hace paso a paso (*hop-by-hop*), decidiendo cada vez a qué router de la red local debo entregárselo para acercarme al destino final.

RUTEO BÁSICO (2)

- El ruteo se hace igual para los datagramas generados internamente por una aplicación como para uno recibido desde la red.
- Un router está conectado directamente a una o más redes, cuyos prefijos de red conocemos. En esas redes pueden haber otros routers que nos permiten ir más lejos.
- El algoritmo de ruteo que toda implementación de IP debe realizar, se basa en una tabla de rutas local

RUTEO BÁSICO (3)

- Esa tabla consiste de una entrada para una red y el router que debo usar para ir hacia ella.
- El router va representado por una dirección IP de una red a la cual yo estoy directamente conectado.
- En esa tabla también figuran todas las redes a las que estoy conectado las que se marcan con un tipo especial (DIR).

RUTEO BÁSICO (4)

- Aquí se puede ver un Host conectado a la red con más de un router y su tabla de rutas.

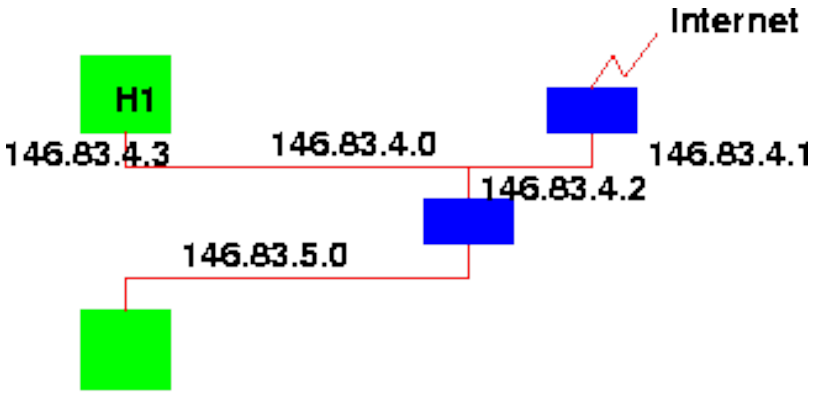


Tabla en H1 :

Net	Gateway	Type
146.83.4.0	146.83.4.3	DIR
146.83.5.0	146.83.4.2	GW
default	146.83.4.1	GW

RUTEO BÁSICO (5)

- Para evitar el tener la tabla de todas las redes de Internet en todas las máquinas conectadas, se usa una ruta default que nos indica nuestro router usual para todas las redes que no conozco.
- Esta ruta se representa con la red 0.0.0.0 y se usa en todos los routers para mostrar la ruta hacia el resto de Internet.

RUTEO BÁSICO (7)

- Ejemplo de tabla de rutas mínima:
  - Prefix = 0/0 => default
  - Gateway = 0.0.0.0 => directly connected
  - Gateway IP tiene que estar en un prefijo al que estoy directamente conectado

NetPrefix	Mask	Gateway (IP)	Interface
192.168.1.0/24	255.255.255.0	0.0.0.0	eth0
127.0.0.0/8	255.0.0.0	0.0.0.0	lo0
0.0.0.0/0	0.0.0.0	192.168.1.1	eth0

RUTEO BÁSICO (6)

- Obviamente, se requiere que algunos Routers de la red (los principales) no tengan ruta default, y efectivamente manejen la tabla completa de las redes conectadas (se llaman Routers default-less).
- Desde cualquier punto de Internet, la cadena de rutas default deben llevarnos a un Router default-less para que el algoritmo funcione.

RUTEO BÁSICO (7)

- Algoritmo de búsqueda en la tabla:
- Filas ordenadas por largo del prefijo de red (máscaras con más 1's primero)
- Entrada default calza siempre (final)

```
search(IP, table)
{
for(i=0; i < table.rows; i++) // ordenadas por largo del prefijo
        if((IP & table[i].Mask) == table[i].NetPrefix)
            return(table[i]);

    return NULL;
}
```

RUTEO BÁSICO (7)

- El algoritmo básico de ruteo es el siguiente:

```
RouteIP(dgram, table)
{
    Route = search(dgram.destIP, table);
    if(Route == NULL) {
        error(dgram, "Net Unreachable");
        return;
    }

    if( Route.type == DIR )
        sendphys(dgram, dgram.destIP, Route.interface);
    else if ( Route.type == GW )
        sendphys(dgram, Route.gateway, Route.interface);
}
```

RUTEO BÁSICO (8)

- El algoritmo anterior depende de la correctitud de las tablas de rutas utilizadas.
- Como estas tablas pueden contener errores, se incluyen algunos mecanismos básicos en IP para evitar daños demasiado graves.
- Por ello, los datagramas IP incluyen el campo TTL, de modo de impedir que un ciclo en las rutas no genere datagramas permanentemente girando en la red, consumiendo ancho de banda sin llegar a ningún lado.

TTL (1)

- Cada paquete IP tiene un campo llamado TTL (Time to Live), HopLimit en IPv6
  - Este campo especificaba el tiempo (en segundos) que el datagrama tiene permitido para viajar por la red. Teóricamente, cada router que procesa el datagrama debe restarle su tiempo de procesamiento a este campo.
  - En la práctica un router procesa un datagrama en menos de un segundo. Por lo tanto el router resta 1 al valor de este campo. Entonces el TTL se convierte en una métrica de “saltos” en vez de ser una métrica de tiempo.

TTL (2)

- Cada paquete tiene un campo llamado TTL (Time to Live) o HopLimit
  - Cuando el valor llega a cero, se asume que el datagrama lleva viajando en un ciclo y es descartado. El valor inicial debe ser asignado por el protocolo de más alto nivel que crea el datagrama.
  - Al destruir un datagrama por TTL, se genera un ICMP “Time Exceeded”

RUTEO BÁSICO (9)

- En IPv6 se le llama HopLimit.
- En IPv4 implica recalcular el checksum en cada router
- Al llegar este contador a cero, el datagrama debe destruirse y no seguir ruteándolo.
- Sin embargo, es bueno generar un mensaje de error para el origen, de modo de advertirle que sus datagramas se están perdiendo.
- Esto va en un datagrama ICMP (Time Exceeded) al origen.

RUTEO BÁSICO (10)

- Sin embargo, es posible que (si hay un ciclo en una dirección) haya un ciclo también en la dirección del origen.
- Si esto ocurre, el datagrama ICMP también verá su TTL llegar a cero, y deberá ser destruido.
- Obviamente, si genero otro ICMP en este caso, ¡ocurrirá lo mismo!

RUTEO BÁSICO (11)

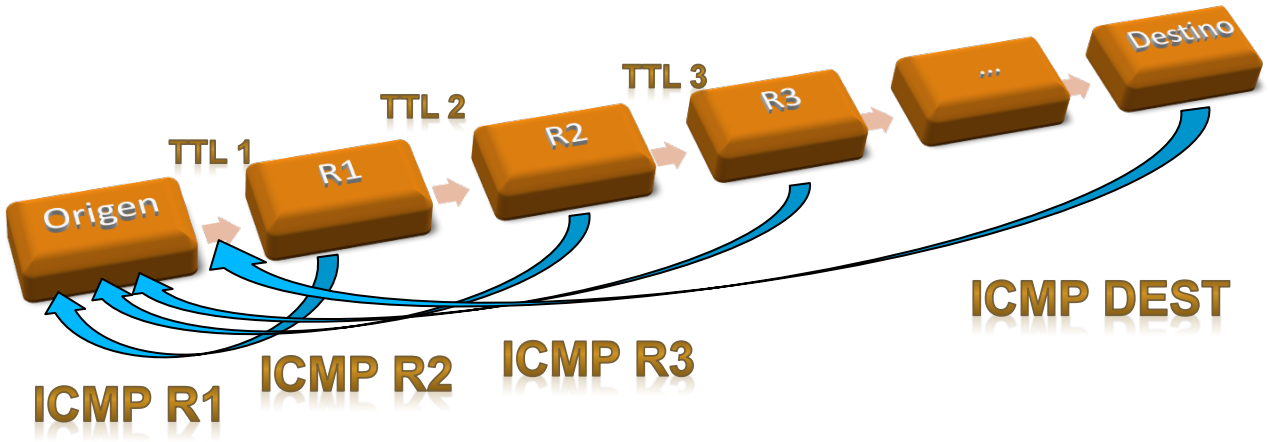
- Por ello, se define en IP que nunca se genera un mensaje ICMP para reportar errores producidos por paquetes ICMP de reportes de errores.
- En el caso de recibir un paquete que debe rutearse por la misma interfaz de red por la que llegó, un Router rutea bien el paquete, pero también genera un ICMP redirect hacia el host de origen, si el origen está en la misma red.

RUTEO BÁSICO (12)

- Traceroute:
  - Queremos ver la lista de routers que atravesamos para ir a “destino”
  - Enviamos paquetes UDP a un puerto desconocido con IP “destino”
  - Primero con TTL=0 y miramos la IP de origen del ICMP “time exceeded”
  - Luego TTL=1, TTL=2, …

RUTEO BÁSICO (13)

Traceroute



RUTEO BÁSICO (14)

```
traceroute to dnc.org.nz (202.78.242.181), 30 hops max, 40 byte
packets
 1  nat.nic.cl (200.7.6.1)  0.248 ms
 2  ciscomle.nic.cl (200.16.114.11)  0.375 ms
 3  200.27.103.25 (200.27.103.25)  0.933 ms
 4  190.208.9.9 (190.208.9.9)  5.904 ms
 5  190.208.9.86 (190.208.9.86)  2.906 ms
 6  ae2-202.nyc20.ip4.gtt.net (173.241.129.209)  157.925 ms
 7  xe-2-3-2.nyc38.ip4.gtt.net (141.136.105.18)  157.822 ms
 8  206.111.13.221.ptr.us.xo.net (206.111.13.221)  151.871 ms
 9  207.88.14.185.ptr.us.xo.net (207.88.14.185)  204.332 ms
10 te-11-0-0.rar3.sanjose-ca.us.xo.net (207.88.12.69)  202.814 ms
11 207.88.13.234.ptr.us.xo.net (207.88.13.234)  201.901 ms
12 ip67-92-171-26.z171-92-67.customer.algx.net (67.92.171.26) 203 ms
13  ten-0-2-0-3.cor01.sjc01.ca.VOCUS.net (114.31.199.242) 358.034 ms
14  ten-0-2-0-3.cor01.syd04.nsw.VOCUS.net.au (114.31.199.28)  355 ms
15  ten-0-2-0-2.cor03.syd03.nsw.VOCUS.net.au (175.45.72.224)  355 ms
16 ten-0-1-0-1.cor01.alb01.akl.VOCUS.net.nz (114.31.199.117) 351 ms
17 ten-1-0-0.bdr01.alb01.akl.VOCUS.net.nz (114.31.202.39)  350 ms
18 as9503.cust.bdr01.alb01.akl.VOCUS.net.nz (175.45.93.118)  352 ms
19 TenGigabitEthernet0-3-0-5020309.akkin-rt2.fx.net.nz
   (202.53.187.197)  351.807 ms
20  * * *
21  * * *
```

HEADER (1)

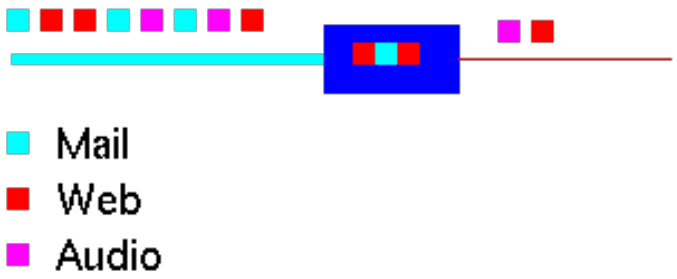
- El header IPv4 puede contener opciones que lo hagan más largo, por ello lleva un campo con el largo del header. IPv6 en cambio tiene un Header de largo fijo.
- Teóricamente, un router no debe cambiar nada en el encabezamiento, de modo de mantener el sobre y el contenido intactos hasta el destino final.

HEADER (2)

- El TTL es la excepción a la regla, y esto complica todo, puesto que el checksum del header debe recalcularse y cambiarse en cada router. Esto hace casi imposible hacer un ruteo eficiente de paquetes IPv4.
- La fragmentación es otro ejemplo
- En IPv6 se ha rediseñado completamente el header de modo de hacerlo de tamaño fijo, sin checksum y las opciones típicamente sólo son analizadas en el destino final.
- La fragmentación ahora es opción end-to-end

Colas, ruteo, delay

- ¿Qué genera el delay variable y la pérdida en Internet?
- Encolamiento en routers:



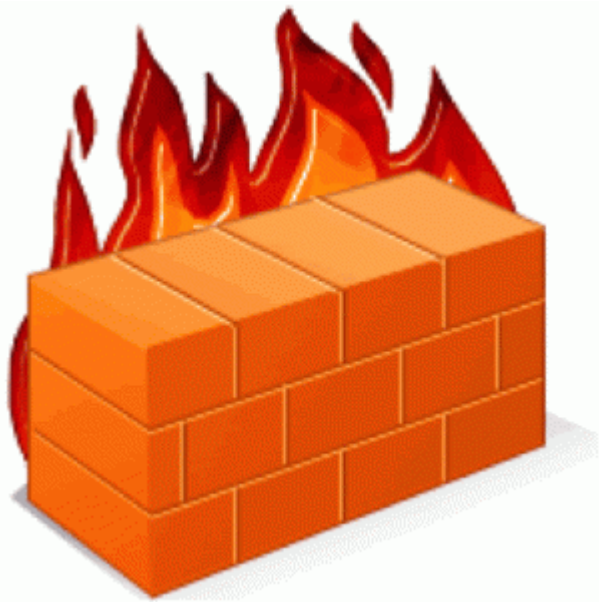
- Congestión introduce delays variables
- Y, finalmente, pérdidas
- ¿Memoria Infinita?
- ¿Todos los paquetes son iguales?

Colas, ruteo, delay

- Cambiar FIFO: colas de prioridad
- Marcar paquetes por prioridad (Audio>Web>Mail)
- Saltar las colas
- Y, finalmente, perder paquetes de la cola para poner este
- Red: trata de perder paquetes al azar antes que se acabe la memoria para señalar a TCP que disminuya su ventana
- Ninguna propuesta funciona en todos los casos
- Aplicaciones Video/audio deben ser adaptativas, tiempo real, altas varianzas en calidad

SEGURIDAD

- Firewalls
- Filtro de Paquetes
- Firewalls tipo Proxy



FIREWALLS (1)

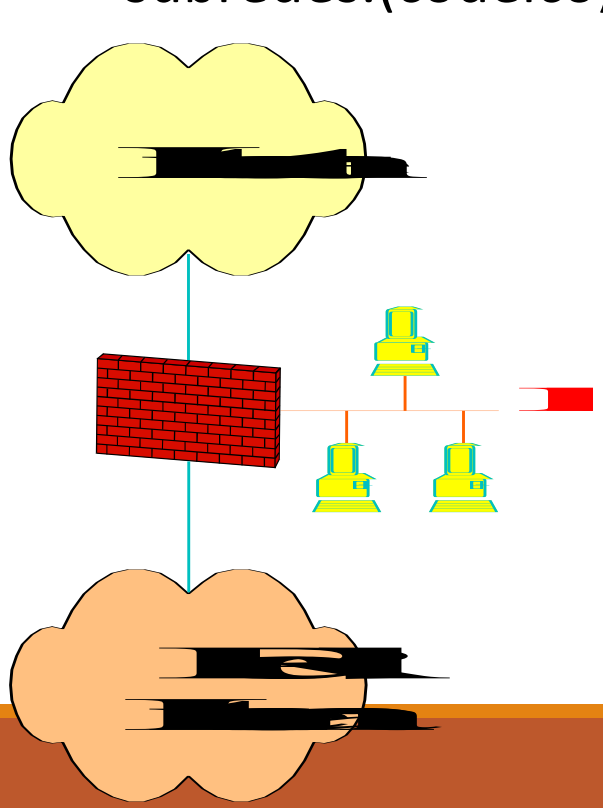
- ¿Qué es un Firewall?
  - Un equipo que se usa para proteger y restringir el acceso desde y hacia computadores en una red.
  - Tiene al menos dos interfaces de red.

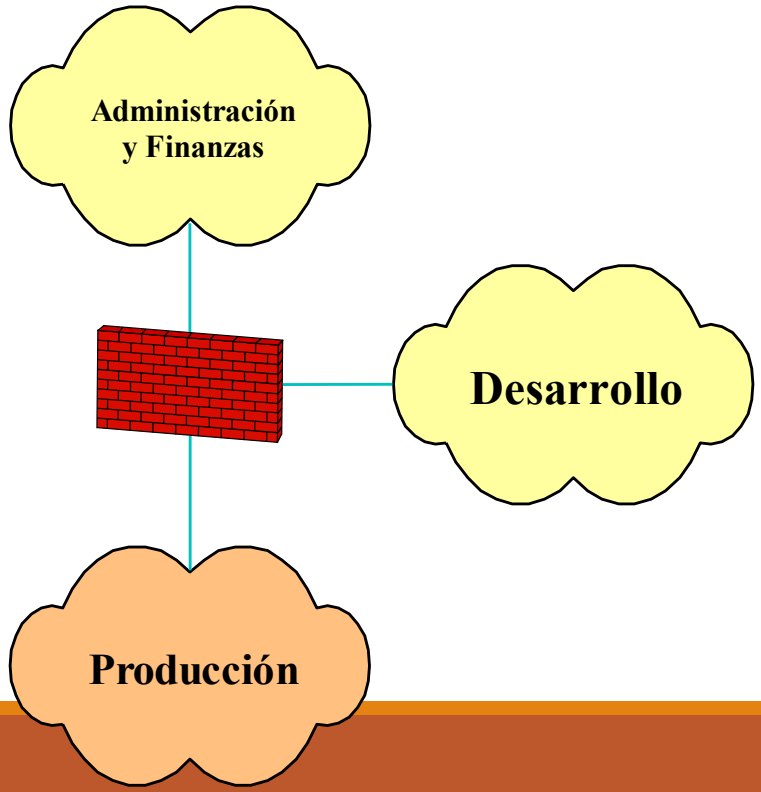
FIREWALLS (2)

- Se necesitan porque:
  - El nivel de seguridad en los computadores es deficiente. Es imposible mantener todos los computadores con todos los parches de seguridad al día.
  - Podemos restringir tráfico por origen, destino y servicio (aplicación).
  - Sirven para restringir y controlar el tráfico entrante y el saliente.

FIREWALLS (3)

- ¿Dónde se Instala un Firewall?
  - A la entrada de una o más redes que deben protegerse.
  - Puede ser a la salida de Internet o entre subredes.(codelco)





FIREWALLS (4)

- ¿Qué puede hacer un Firewall?
  - Definir reglas que dependan del día/hora.
    - Permitir que bajen películas después de las 20:00 L-V.
  - Autenticación de usuarios de varias formas:
    - User/password, servidor RADIUS, certificado digital, tarjeta token.
    - Autenticación transparente de usuarios.
    - Disponible para cualquier servicio IP.

FIREWALLS (5)

- ¿Qué puede hacer un Firewall?
  - Traducción de direcciones IP (NAT)
    - Permite usar direcciones IP inválidas en la red interna y usar direcciones válidas al salir a Internet (RFC 1918).
    - Redes Privadas hoy son enormes
    - Direcciones internas 1-1 o n-1 con direcciones válidas.
      - Cuando la traducción de direcciones incluye traducción de puertos, se conoce como PAT o masquerading.

Redes Privadas y seguridad

- Han permitido que Internet crezca casi sin usar direcciones públicas
- Pero, ¿aportan en seguridad?
- ¿Son necesarias en IPv6?
- -> Complican las fusiones de empresas con redes privadas
- -> Les encantan a los administradores de redes

FIREWALLS (5)

- NAT e IPv6
  - NAT is evil
    - En Ipv6 ya no es necesario tener redes privadas
    - ¿Aporta en seguridad?
    - ¿Daña la funcionalidad?

FIREWALLS (6)

- ¿Qué puede hacer un Firewall?
  - Soporta H.323, FTP y otros protocolos donde la dirección IP del cliente se envía al servidor.
  - Inspección de contenido
    - Antivirus, bloqueo de applets, Java, ActiveX, …
  - Interfaz gráfica o vía browser
    - Administradores read-only vs. read/write.
    - Administración de múltiples firewalls centralizadamente.

FIREWALLS (7)

- ¿Qué puede hacer un Firewall?
  - Alta disponibilidad
    - Permite configurar dos firewalls en paralelo en modo activo-pasivo que se mantienen sincronizados.
    - En algunos fabricantes en modo activo-activo (balanceo de carga).
  - Genera log para auditoría con software de terceros.
  - Administración de tráfico.
  - Integración de gateway VPN (IPSec)

FIREWALLS (8)

- Implementación
  - Hardware especializado
    - Sistema operativo propietario
    - Sin disco duro ni memoria virtual
    - No corre otros procesos ni servicios
    - Ejemplos: Checkpoint Firewall-1, Cisco PIX.
  - Software comercial sobre un servidor
    - Sistema operativo comercial (Windows, Unix) o Linux.
    - Ejemplos: ZoneAlarm, Norton Internet Security, Kerio Personal Firewall. Iptables, ipchains o ipforward en Linux. Ipfw en *BSD.

FIREWALLS (9)

- Implementación
  - Combinación hardware + software
    - Bundle de hardware con sw de firewall cargado en flash.
    - Corre algún sistema operativo jibarizado (Linux, BSD)
    - Switch de alto rendimiento con un firewall externo.

FILTRO DE PAQUETES(1)

- Permite especificar cuáles paquetes pueden pasar y cuáles no pueden pasar por el filtro. Tienen un conjunto de reglas (conocidas como ACL, Access Control List) que definen:
  - Dirección fuente, dirección destino, servicio, acción.
  - Dirección fuente o destino puede ser una dirección individual, una subred o “todos” (Generalmente se anota en la forma dirección/mascara).
  - El servicio se identifica por el puerto de origen o destino y el protocolo asociado.
  - Acción puede ser “permitir” o “denegar”. En algunos casos, la acción de denegación se traduce en un rechazo y en otras en ignorancia.
- Cuando la acción es “permitir”, el filtro de paquetes actúa como un router.

FILTRO DE PAQUETES(2)

- Una regla por defecto al final que deniega todo:

IPorigen	PuertoOrigen	IPDestino	PuertoDestino	Protocolo	Acción	Comentario
*	*	Mail-Server	25	TCP	Permitir	SMTP
*	*	Web-Server	80	TCP	Permitir	HTTP
*	*	Web-Server	443	TCP	Permitir	HTTPS
*	*	DNS-Server	53	TCP, UDP	Permitir	DNS
Red Interna	*	*	80	TCP	Permitir	HTTP
Red Interna	*	*	443	TCP	Permitir	HTTPS
Red Interna	*	*	21	TCP	Permitir	FTP
Mail-Server	*	*	25	TCP	Permitir	SMTP
DNS-Server	*	*	53	TCP, UDP	Permitir	DNS
*	*	*	*	*	denegar	

- En algunos casos, esta regla por defecto permite todo.
- Hay un regla explícita que permite el retorno de los paquetes en respuesta a una conexión permitida. Generalmente es especificada como related o established.

FILTRO DE PAQUETES(3)

- Paquetes relacionados
  - En el caso de TCP, generalmente se aceptan aquellas conexiones originadas desde el “interior” de la red.
  - Se identifican mediante la detección de un segmento SYN.
  - Se puede permitir conexiones desde el exterior, pero sólo si se hace explícitamente.
  - En el caso de UDP, se relacionan mediante la tupla (D_{origen}, P_{origen}, $D_{destino}$, $P_{destino}$)

FILTRO DE PAQUETES(4)

- Procesamiento de Paquetes
  - Cuando un paquete llega al filtro:
    - Se revisa la lista de reglas en secuencia hasta encontrar una que haga match con el paquete.
    - Si se encuentra, se toma la acción correspondiente.
    - Si no se encuentra:
      - Si es un paquete de una sesión TCP establecida (respuesta de un servidor a un cliente), dejarlo pasar.
      - Si no, tomar la acción por defecto.

FILTRO DE PAQUETES(5)

- Procesamiento de Paquetes
  - Como se busca el primer match con una regla, el orden de las reglas importa.
    - Por ello se recomienda colocar las reglas que habilitan servicios primero y luego una que niegue todo lo que no se acepta explícitamente.
  - La conexión TCP es directa entre el cliente y el servidor. El filtro actúa como router.

FILTRO DE PAQUETES(6)

- Tipos de Filtros de Paquetes
  - Stateless
    - Cada paquete de considera independiente y aislado de cualquier otro.
      - No es capaz de detectar conexiones.
      - Fueron la primera generación de firewalls
  - Stateful
    - Mantienen información de control de los paquetes, asociándolos a un flujo o conexión.
    - Los más avanzados permiten reglas mas complejas, como que un flujo no exceda cierto throughput (traffic shaper) o cierta tasa de conexiones por unidad de tiempo.

FILTRO DE PAQUETES(7)

- El Problema del FTP

  - Cuando el cliente FTP quiere bajar o subir un archivo, le manda un comando al servidor FTP pidiéndole que le mande el archivo a un puerto de destino especificado en el comando:

  - El cliente FTP espera en un puerto aleatorio y la conexión que vendrá del servidor FTP. El servidor inicia la conexión.

  - ¿Cómo un filtro de paquetes puede permitir esta conexión?

FILTRO DE PAQUETES(7)

- El Problema del FTP
  - Existe una opción en el FTP llamada “FTP pasivo” que si el cliente y el servidor soportan, hace que el cliente siempre abra la conexión.
  - Otra solución es permitir todas las conexiones desde afuera a clientes internos a puertos > 1024 (no privilegiados).
    - Se corre el peligro de permitir conexiones de afuera que no deberían permitirse.
  - Un problema similar tienen los filtros de paquetes con X Windows (el servidor abre conexiones al cliente).

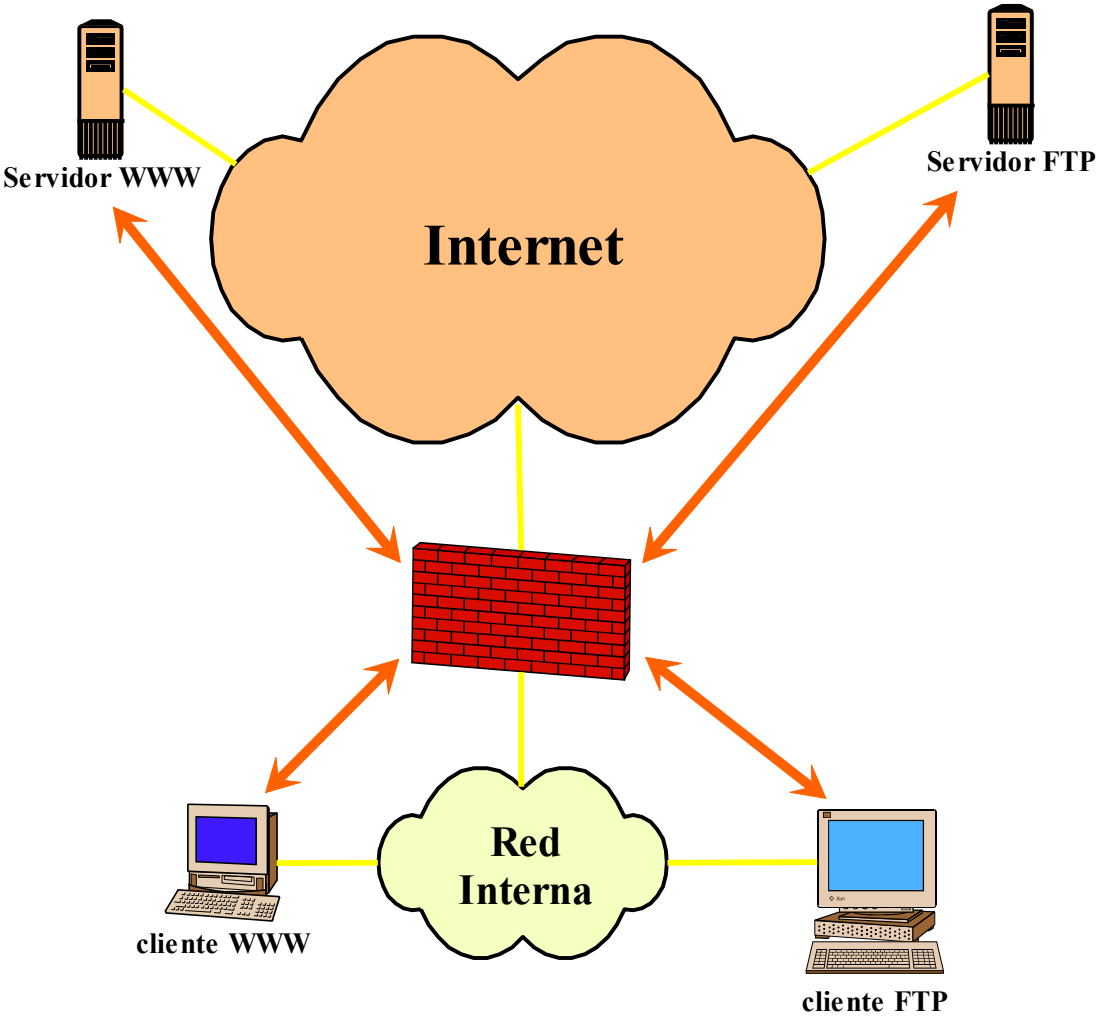
FILTRO DE PAQUETES(8)

- Ventajas del filtro de paquetes:
  - Barato (gratis si ya se tiene un router)
  - Soporta cualquier protocolo (servicio TCP)
- Desventajas:
  - No soporta bien FTP, X Windows y otros protocolos donde el servidor abre una conexión al cliente.
  - El filtro de paquetes usa ciclos de la CPU del router. Podría llegar a sobrecargar al router.
  - El rendimiento depende del número de reglas. La mayoría de los paquetes deberían tener un match con las primeras reglas.
  - No puede revisar contenido de los paquetes ni restringir comandos de los protocolos que pasan por él.

FIREWALL TIPO PROXY (1)

- Es un computador con dos o más interfaces de red que separa físicamente la red interna de la red externa (Internet).
  - No se permiten conexiones directas entre ambas redes.
  - En el firewall existen procesos que actúan como proxy para las conexiones permitidas.
  - Cuando un cliente quiere abrir una sesión TCP contra un servidor, se debe conectar al proceso proxy, quien abre otra conexión al destino final y actúa como intermediario entre las dos puntas. Este tipo de proxy se conoce como SOCKS.
  - Esto permitiría revisar el contenido de los paquetes…
  - Ejemplo: servidor proxy HTTP.

FIREWALL TIPO PROXY (2)



FIREWALL TIPO PROXY (3)

- Ventajas
  - Permite control más fino sobre el contenido de las conexiones.
  - En una conexión FTP o HTTP o SMTP, podría restringir los comandos que se pueden ejecutar o las URLs.
  - El servidor proxy SMTP podría correr un antivirus.
  - Al no existir una conexión directa entre el equipo en la red interna y el equipo en Internet, algunos ataques no se pueden materializar.

FIREWALL TIPO PROXY (4)

- Ventajas
  - El servidor proxy para FTP puede capturar la dirección IP del cliente y el puerto que se envió al servidor FTP para permitir la conexión desde el servidor al cliente durante una cierta ventana de tiempo, lo mismo con otros protocolos que requieren que el servidor se conecte al cliente.
  - Como el cliente se conecta al firewall primero, éste puede pedir una autenticación previa a conectarse a Internet.

FIREWALL TIPO PROXY (5)

- Desventajas
  - Requiere cambiar las aplicaciones clientes para que soporten el uso de un proxy (no todas lo soportan).
  - Se requiere un proceso proxy por cada conexión TCP establecida, lo que puede congestionar al firewall y requiere más CPU y más memoria que un filtro de paquetes.

FIREWALL TIPO PROXY (6)

- Desventajas
  - Aumenta la latencia al tener que estar recibiendo paquetes, procesarlos y mandarlos por otra interface de red.
  - Sólo pueden usarse protocolos (aplicaciones) para los cuales existe un servidor proxy en el firewall.
    - Si aparece una aplicación nueva, no será soportada hasta que el proveedor del firewall desarrolle un servidor proxy para ella o provea una API para desarrollar un proxy tipo “puente” rápidamente.

FIREWALL TIPO PROXY (7)

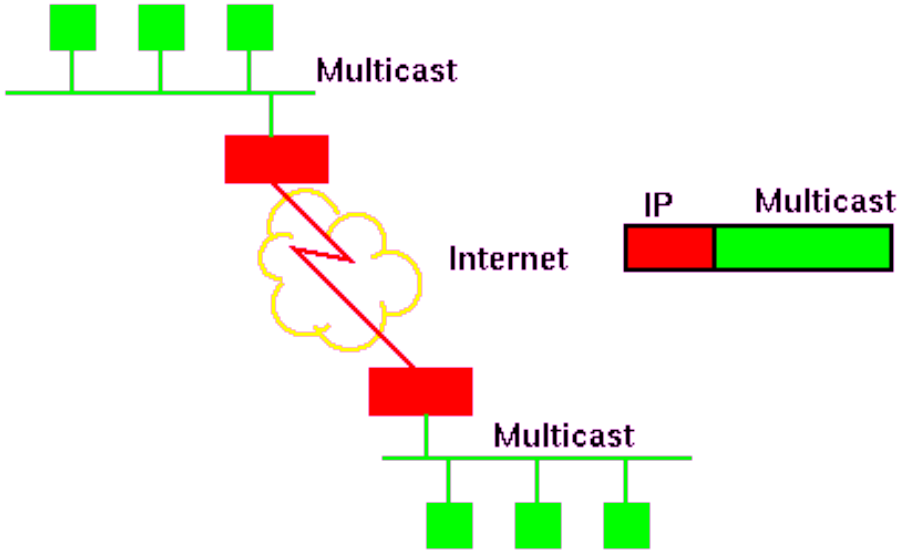
- Firewalls Mixtos
  - Funcionan como filtro de paquetes para la mayoría de las aplicaciones pero tienen servidores proxy para algunas.
  - Firewall-1 tiene proxy para HTTP, FTP, SMTP, TELNET, rlogin, ping y puede interceptar cualquier servicio para que el usuario se autentique primero.
  - Así, cuando un servicio no requiere un proxy, el firewall actúa como filtro de paquetes. Cuando requiere proxy, levanta un servidor proxy para ese servicio.

Red Privada y acceso

- Mi casa / empresa
  - ¿Cómo hace skype?
  - ¿Cómo instalo un servidor?
  - ¿cómo acceso mis cámaras IP desde fuera?

VPN (1)

- Túneles IP

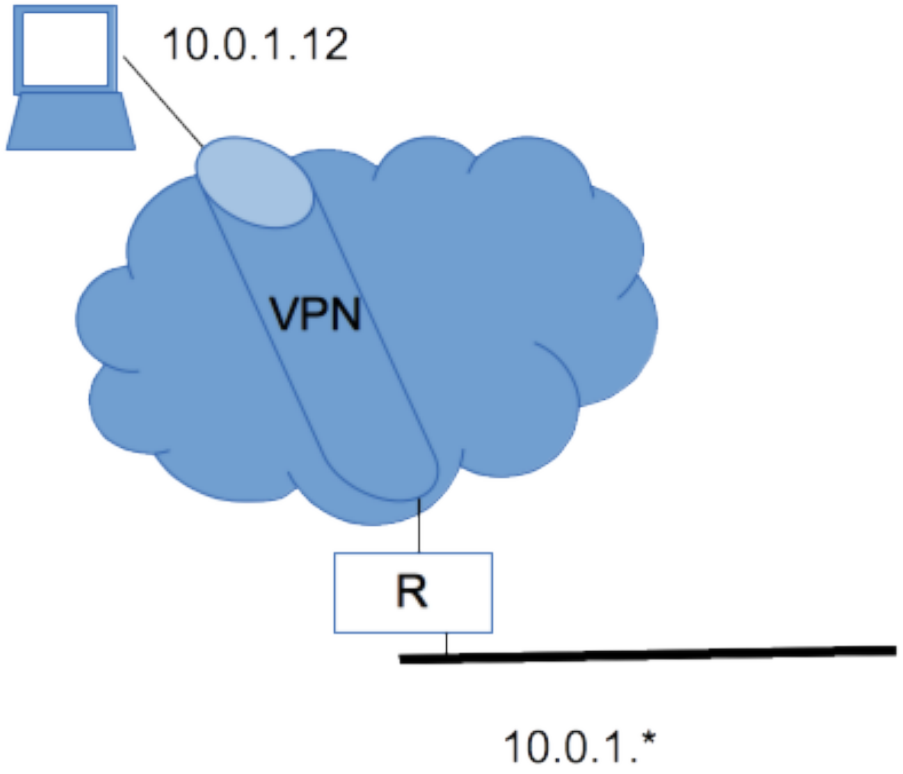


VPN (2)

- Encapsula IP sobre IP
- Funciona como un enlace punto a punto sobre Internet
- La VPN usa un túnel con el router de acceso a la red privada para poder usar una IP de la red interna
- Además, el túnel se encripta y se firma para garantizar privacidad y seguridad
- Para establecer el túnel requiero autenticarme fuertemente

VPN (3)

- VPN



Mobile IP

- Túneles IPv4