# Comunicación de Datos 2

## Notas, Definiciones, Protocolos

El protocolo de transporte **UDP** se utiliza para transmitir datagramas de forma rápida en redes IP y funciona como una alternativa sencilla y sin retardos del protocolo TCP. Se usa principalmente para: consultas DNS, conexiones VPN y para el streaming de audio y vídeo.

# Diferencias IPv4 - IPv6

## Problemas en IPv4

Problemas del direccionamiento classful

Prefijos de longitud fija (classful), provoca un uso muy 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)

Solución Temporal:

* Prefijos variables (classless) junto con CIDR y NAT
* CIDR propone un uso racional de las direcciones IPv4
* Cambio en la estructura de las direcciones Ipv4 (classful a classless)
* Reasignación de direcciones
* Asignación topológica de bloques de direcciones ex clase C
* NAT además permitió y permite compartir una dirección IP pública entre varios equipo
  + NAT44: Realizado en el router de entrada a la intranet del usuario, obteniendo una IP pública
  + NAT444 (CGNAT): Realizado por el ISP. El usuario final no adquiere una IP pública

## Características favorables de IPv6

Espacio de direcciones muy ampliado

Header IP más simple: Longitud fija y Opciones fuera del header

Incorpora seguridad (IPsec) y noción de flujos

Fragmentación sólo en el origen, no en routers intermedios

Autoconfiguración (SLAAC)

Simplicidad y flexibilidad en la administración

Mejora en movilidad

Uso de direcciones multicasting (no más broadcast)

Jumbogramas

¡Compatibilidad con IPv4!

**Datagramas** **IPv6**

Están compuestos de un número variable de headers

El primero es el header IPv6 : es obligatorio, reemplaza al datagram IPv4 (sin opciones). Luego existirán un número variable de headers intermedios. Último header (no es necesario que esté presente), protocolo encapsulado en IP (ej TCP/UDP) Un router procesa el header IPv6 y opcionalmente, en orden, los demás headers

Ventajas: Mayor eficiencia y Mayor funcionalidad



**Header IPv6**

Versión (4 bits): 6

Traffic Class (1 byte):

Flow Label (20 bits): identificador de flujo para el par origen-destino

Payload length (2 bytes): longitud de los datos

Nxt Header (1 byte): identificador de extensión al encabezamiento o protocolo encapsulado

Hop limit (1 byte): alcance del datagram

Direcciones (origen y destino) (16 bytes)

Extensiones: longitud variable, formato propio

**Diferencias entre datagramas IPv4/IPv6**

IPv4

* Longitud variable: entre 20 y 60 bytes,
* Direcciones: de 4 bytes

IPv6 (header)

* Longitud (fija): 40 bytes,
* Direcciones: 32 bytes,
* Soporte para headers adicionales,
* Incremento por cada extension header: múltiplo de 8 bytes,

**Otras Mejoras IPv6 respecto de IPv4:**

Longitud fija del header a analizar por los routers, y Soporte para headers adicionales.

No se calcula checksum y No se fragmenta en los routers intermedios.

Identificación de flujo más eficiente

**Extensión de headers IPv6**

Orden de los Headers

El Hop by Hop DEBE ir al comienzo (luego del tipo (v6))

Se recomienda que para los demás se respete el siguiente orden en los Tipos de Headers:

Hop by Hop header

Destination Options header

Routing header (0 y 2)

Fragment header

Authentication

Encapsulating Security Payload

Destination Options header

**Las Direcciones IPv6** :

* Espacio de direcciones significativamente mayor que IPv4, IPv6 posee 128 bits de longitud (6,65 \* 10 ^ 23 direcciones por cada m2 de la tierra)
* Presentan Facilidad de administración en la intranet
* Organización eficiente en la Internet
* Eliminación de NAT, ya no hace falta un router representante
* Una interfaz de red puede tener un número arbitrario de direcciones, de diferentes alcances
* Autoconfiguración: en el arranque, el equipo genera automáticamente una dirección de alcance link
* Se posibilitan nuevas aplicaciones sin inconvenientes anteriores(p.ej. Internet of Things)
* Son Representadas en notación hexadecimal con 8 grupos de 16 bits separados por “:”
* Uso de prefijo igual que en IPv4, se dividen en una parte red y host (indicada con /)
  + 2001:1234:0000:0000:0000:0001:0000:ab54 /64
  + Red: 2001:1234:0000:0000 --- Host:0000:0001:0000:ab54
* Son válidos los valores “todos ceros” y “todos unos”, en la parte de red o de host… esta limitación se eliminó
* Debido a su longitud, se representan “compactadas”, eliminando los ceros

según ciertas reglas

**Reglas de representación de direcciones IPv6**

* Una Dirección Completa: 2001:0db8:85a3:0000:0000:8a2e:0370:7334
* Usando Supresión de ceros: 2001:db8:85a3:0:0:8a2e:370:7334
  + se suprimen ceros al comienzo de cada bloque (no al final)
  + cada bloque debe tener al menos un dígito hexadecimal

Usando Compresión de ceros:

* 2001:db8:85a3::8a2e:370:7334 (dos grupos fueron comprimidos)
  + grupos de ceros contiguos se reemplazan por “::”
    - solo es posible comprimir grupos completos y no partes
      * solo se puede utilizar una única vez en umayor grupona dirección, al mayor grupo.
      * Si hay dos grupos iguales, debe aplicarse al que está más a la izquierda
* Representación de direcciones mixtas: solo los últimos 32 bits (dos bloques de 16 bits, igual que una dirección ipv4) en notación decimal con puntos, solo cuando esto se pueda deducir del prefijo.

Ejemplo: IPv6 del tipo IPv4-mapeada ::ffff:c000:280/96 => ::ffff:192.0.2.128

* Representación de ports: [2001:db8::1]:80

## Tipo de Direcciones IPv6

#### Unicast y Subclasificacion

Estas direcciones se asignan únicamente a un equipo… sin embargo su alcance varía y está dado por los bits de “scope” e Identifican múltiples interfaces de conexión a la red. Un datagram enviado a una dirección unicast es entregado a esta interfaz.

Identifican una única interfaz de conexión a la red

**Tipos de direcciones unicast: Globales , Link , y Site**

**Direcciones de alcance global:**

Identificadores únicos de alcance global, son reconocidas en toda la Internet (lo que antes eran las ip publicas)

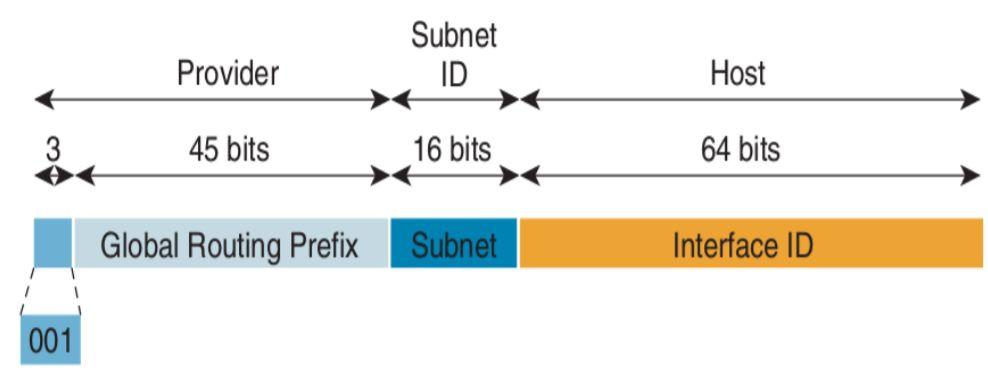
Asignados explícitamente a una interfaz

Se requiere solicitarlas a un RIR (Registro Regional de Internet)

Utilizadas para comunicar equipos en la Internet.

Para la Representación de Direcciones globales unicast, se utiliza el prefijo 2000::/3

Espacio dividido en bloques para: Asignación a proveedores o Propósitos especiales



Una dirección IPv6 loopback (o de "bucle local" en español) es una dirección especial que se utiliza para dirigir el tráfico de red a la misma máquina que lo genera. En otras palabras, esta dirección se utiliza para hacer referencia al propio equipo en el que se encuentra, de manera similar a como la dirección IPv4 127.0.0.1 se utiliza para el mismo propósito.

En IPv6, la dirección loopback se representa como ::1 y se utiliza para probar la configuración de la red en un sistema y para comprobar que los servicios de red en la máquina funcionan correctamente. Los paquetes de red enviados a la dirección IPv6 loopback no salen de la máquina local y se entregan inmediatamente al proceso que los generó.

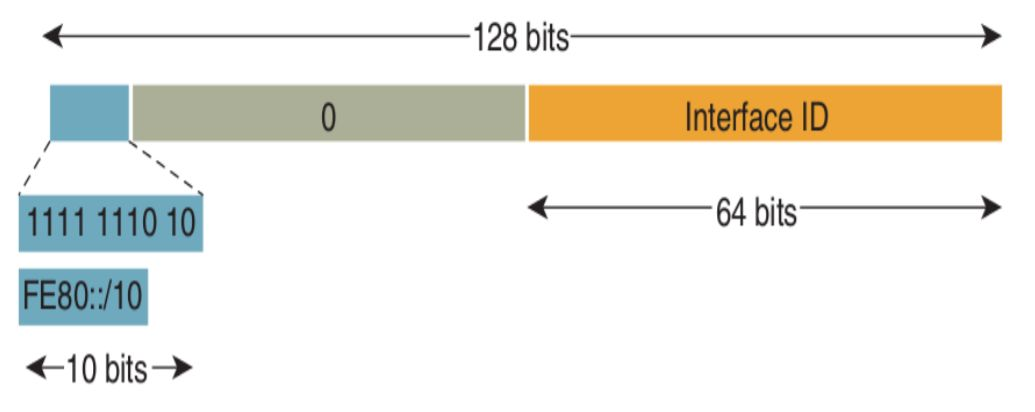
La dirección loopback es una dirección especial que no se asigna a ningún dispositivo de red y se reserva exclusivamente para uso local.

**Direcciones de alcance en el link (Link local)**

Válidas sólo en la red física, son privadas (p.ej. Ethernet)

Generadas automáticamente, y se repite el prefijo de alcance en todas las redes.

Utilizadas para procesos de autoconfiguración, e interfaces punto a punto debido a su sencillez. Y se representan de la siguiente manera:



**Direcciones Unique de alcance en el site o locales (Unique Local Address – ULA-)**

Similares a las direcciones privadas IPv4

Usadas para comunicar equipos dentro del site local, sin el riesgo que tener que usar direcciones globales

No permitidas en la Internet global. Pueden ser únicas para cada site.

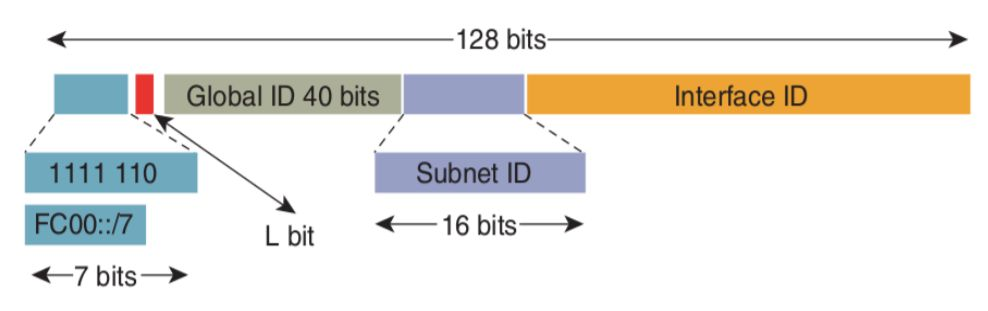
Son manejadas independientemente por cada empresa.

Para el global id se utiliza un método aleatorio, es raro que dos empresas compartan el mismo… por más que suceda, no hay problema porque son locales a la organización (son locales)

Representación de Unique Local Address (ULA)

FC00::/8 Reservado para uso futuro

FD00::/8 ULA



**También existen los tipo Multicast y Anycast**

**NO se definen direcciones broadcast o difusión !**

#### **Direcciones Multicast IPv6**

Un datagram enviado a una dirección multicast es entregado a todas las interfaces asociadas a esta dirección.

Se envían los paquetes a una red con una dirección m1 de esté tipo pero solo lo analizan/procesan aquellos que tienen Multicast m1. Se trata de un broadcast v4 controlado

Ejemplo de direcciones multicast (1)

Solicited Node Address (SNA)

Son utilizadas para realizar el Neighbor Discovery

Se debe agregar una SNA por cada una de las direcciones

Mejora la performance de ARP utilizado en IPv4

Se generan o representan con el prefijo ff02::0001:ff00:0000/104 y se completa con los últimos 24 bits de la dirección IPv6

Otras direcciones multicast conocidas, es decir casos particulares :

FF02::1 -> todos los nodos de la red local

FF02::2 -> todos los routers de la red local

FF05::2 -> todos los routers de la red site

#### **Direcciones Anycast IPv6**

Son como las direcciones unicast, pero compartidas (varios equipos pueden compartirla)

El alcance está dado por sus primeros bits

Identifican múltiples interfaces de conexión a la red

Un datagram enviado a una dirección anycast es entregado a una sola de las interfaces

asociadas a la dirección: la primera que se encuentre en el camino

Son de gran de utilidad cuando un servicio(envio de paquetes) se realiza en más de un host/servidor

Sintácticamente iguales a las direcciones unicast

Requieren soporte del sistema de ruteo.

Por el momento, solo utilizadas como direcciones de destino y asignadas a routers

Uso restringido de direcciones anycast globales (no es escalable)

Ejemplo, "dirección anycast del router de la subred"



### Identificador de la interfaz con EUI-64

Esté campo debe ser único para cada equipo en una misma subred, para cada interfaz

Es un campo olbligatorio siempre que el prefijo sea 001 a 111.

Como se genera ?

Cada equipo tiene una MAC con 6 bytes = 6 \* 8 (48) bits, que vamos a utilizar.

Los 24 bits de la izquierda identifican al fabricante (definido por IEEE)

Los de la derecha identifican al producto, y los da el fabricante… Si bien no son direcciones totalmente unicas, vamos a suponer que lo son

Entre esos dos grupos de 24 bits se completa con 11111111 11111110 tal que se llega a los 64 necesarios. Y por último se cambia el septimo bit de la direccion total por 1 (bit de Unicidad de IEEE, es siempre 0)

Vemos que ahora la dirección de la interfaz de un equipo será igual en cualquier red

**Distintos métodos de autoasignación**

Tecnicas de filtrado en ipv6:

RTBH: Remote Triggered Black Hole

La idea es pasar desde mi trigger router, contenedor y administrador del blackhole

**Asignación de direcciones Globales:**

De esto se encarga la IANA donde cada RIR tendrá /23 y los ISP /32, para las corporaciones se asigna generalmente prefijo /42 o /64 y a los usuarior residenciales no hay una asignación acordada pero se suele utilizar /60 o /64. Con ciertas consideraciones:

Se debe hacer un uso racional del espacio cuando empecemos a asignar las mismas… si bien NAT ya no se usa, de todos modos los tamaños de las subredes a definir deben ser compatibles con las ULA.

**Asignación de direcciones IPV6:**

**puede ser tanto automática y manual**

**y en routers y en hosts**

Las asignaciones automaticas o autoconfiguradas son las statefull y las multicast que se dan automaticamente al conectarse a una red.

Las direcciones globales y ULA se la asignan solos los host con el proceso radvd. radvd utiliza a su vez [neighbor discovery](https://en.wikipedia.org/wiki/Neighbor_Discovery_Protocol).

**Neighbor discovery:** es un protocolo basado en ICMPv6.

Permite descubrir a mis vecinos conectados y autoconfigurar las direcciones (para diferentes alcances), configurar los parámetros, descubrir routers, determinar el estado de diferentes hosts, etc…

Maneja 5 mensajes: Neighbor Solicitation(NS)(f80::id\_interfaz) y Advertisment(NA), y los correspondientes para routers (RS) y (RA), por último está Redirect

Método de Router Discovery:

**Los routers por otro lado:**

Utilizan Router Discovery con el objetivo de:

* Ubicar routers adyacentes (en el mismo link)
* Conocer cuáles pueden actuar como default routers
* Conocer prefijos de direcciones de red locales al link
* Adquirir información relativa a autoconfiguración de direcciones
* conocer el MTU del canal, el tipo de autoconfiguración, y tiempos de validez de los parámetros anunciados

//Método de Router Discovery:

Periódicamente los routers en el link anuncian su presencia (unsolicited advertisment)

Disponibilidad de actuar como default router

Prefijos de red

Otros parámetros de configuración de hosts

Un router puede emitir un advertisment en cualquier momento, en circunstancias

especiales (unsolicited advertisments)

Un router puede emitir un (solicited) advertisment a pedido de un host

Chequeos de consistencia entre advertisments de los diferentes routers

Un host en el link utiliza la información de los advertisments para mantener su información

local:

Lista de default routers

Lista de prefijos

MTU, etc

Acepta información de distintos routers (prevalece el último valor, si llega a haber inconsistencias)

Un host que necesite configurarse puede enviar un mensaje Router Solicitation

Todos los routers en el link responderán con advertisements unicast

**Router Discovery - Linux IPv6 Router Advertisement Daemon (radvd)**

Permite configurar a los hosts de manera stateless

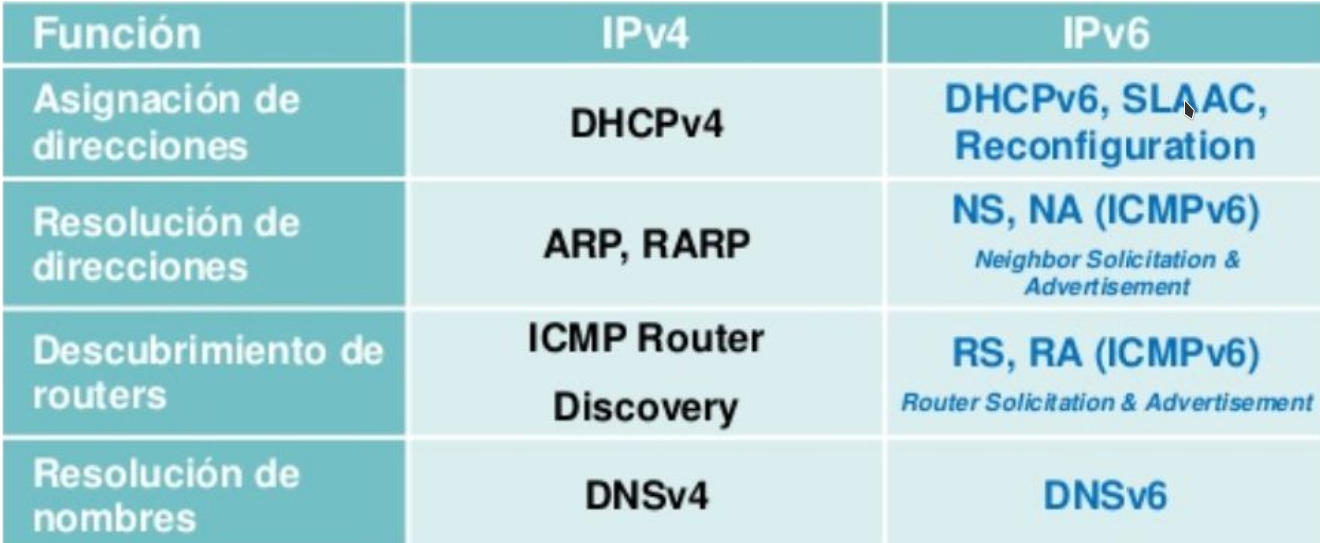
Configuración en /etc/radvd/radvd.config

Utiliza Router Advertisements y Router Solicitations

Puede configurarse una variedad de parámetros

el comando “radvdump” permite monitorear los anuncios emitidos y parámetros enviados por cada interfaz.

**DUDA: Por que las redes a veces se asignan sólo la dirección global (es obligatoria ?) y algunas redes también agregan la ULA ?**



## 

## 

## Coexistencia y transición IPv4 - IPv6

Dado que en el mundo se utilizaba IPv4, surge la necesidad de realizar mecanismos de transición hacia IPv6.

**Túneles:** transporte de datagramas v6 en redes v4, o viceversa

**Traslación:** conversión de direcciones v4 a v6 y de v6 a v4 (se traducen las direcciones)

**Doble stack:** Implementación de ambos protocolos (ipv4 e ipv6), en el mismo equipo

**Doble Stack**

Es un método de transición suave

Consiste en implementar dos “pilas de protocolo” en algunos dispositivos. Tal que funcionan con ambos protocolos. Ejemplo:

Web Server “servidor” (solo v6) Cuando arranca, pide un socket, port 80

El SO doble-stack asigna dos direcciones: 12.1.1.1 port 80 (v4) y 2001:1234::1 port 80 (v6)

**DUDA: doble stack puede implementarse en cualquier host o ruter ?**

Ej de las filminas:

Cliente browser (solo v4) quiere comunicarse con “servidor” que solo maneja ipv6.

Pide al DNS un registro A (mapping de nombre en dirección v4)

DNS devuelve: 12.1.1.1

Cliente envia datagram a 12.1.1.1 port 80, mediante su router intermedio

El router doble stack: como el server es v6 solamente, cambia v4 a v6 mapped: ::ffff:12.1.1.1

Web server “servidor” recibe la consulta de ::ffff:12.1.1.1.

Contesta a esa dirección v6

SO doble stack cambia la dirección v6 mapped por la v4 (12.1.1.1) y envía datagram

Browser recibe respuesta desde 12.1.1.1 port 80

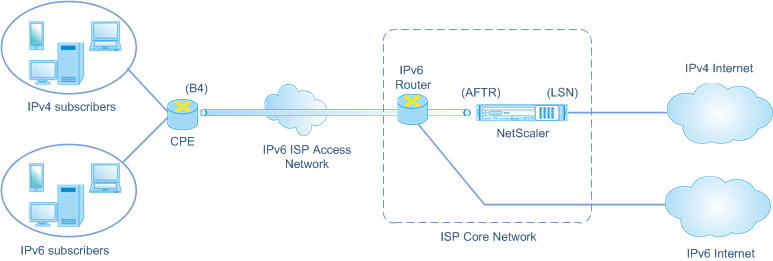
**Doble Stack Lite (DS Lite)**

Permite que el ISP desarrolle redes de acceso solo IPv6

Utiliza un Túnel IPv4 en IPv6 y tambien NAT, para hacer acceder a las redes v4 al ISP con una red v6 de por medio.

Cuando un paquete v4 llega al ISP, debe desencapsularlo y enviarlo por el AFTR conectado

Es independiente del resto de la internet, abstrae al ISP.



**Túneles**

Método ampliamente utilizado

Existen varios tipos de túneles para comunicar dos equipos (autom. y manuales)

Son para paquetes ipv6 y tienen que pasar por una ipv4.

**tunel de tipo IPv6 to IPv4 (automático)**

Para este tipo de túneles, consiste en la asignación de direcciones ipv6 que tengan embebida una dirección ipv4 publica (aclaración: estos paquetes tienen un prefijo especial: 2002::16, y luego la parte que corresponde a ipv4, y completo con )

**tunel de tipo Teredo (automático)**

En este caso no es necesario que los hosts tengan un ip publica, y el host puede estar detrás de un NAT. En este caso el router/server teredo necesita saber cuál es el NAT que se está utilizando en ipv4

**Teredo**

Proporciona conectividad IPv6 a host que soportan IPv6 pero que se encuentran conectados a internet mediante una red IPv4.

Opera usando un protocolo de túneles independiente de la plataforma diseñado para proporcionar conectividad IPv6 encapsulando los datagramas IPv6 dentro de datagramas UDP IPv4.

Estos datagramas pueden ser encaminados en Internet IPv4 y a través de dispositivos NAT. Otros nodos Teredo, también llamados Teredo relays, tienen acceso a la red IPv6, reciben los paquetes, los desencapsulan y los encaminan.

**Traducción y Traductores IPv4/IPv6**

**intro:** Debido al agotamiento inminente de las direcciones IPv4, los ISP han comenzado a realizar la transición a la infraestructura IPv6. Sin embargo, durante la transición, los ISP deben seguir soportando IPv4 junto con IPv6, porque la mayor parte de Internet público todavía utiliza IPv4.

NAT64 a gran escala es una solución de transición IPv6 para ISP con infraestructura IPv6 para conectar a sus suscriptores solo IPv6 a Internet IPv4. DNS64 es una solución para permitir la detección de dominios solo IPv4 por parte de clientes solo IPv6. DNS64 se utiliza con NAT64 a gran escala para permitir una comunicación perfecta entre los clientes solo IPv6 y los servidores solo IPv4.

Poco recomendable porque pierdo las ventajas de ipv6, pero efectiva

Permiten comunicación entre equipos que sólo soportan IPv6 con equipos que sólo soportan IPv4.

**Método Stateless IP/ICMP Translation (SIIT)**

Traduce entre los formatos de cabecera IPv6 y IPv4

Utiliza el prefijo de red ::ffff:0:0/96 (IPv4 mapeadas)

**Método NAT64**: NAT64 es un mecanismo que permite a hosts IPv6 comunicarse con servidores IPv4. El servidor NAT64 dispone de al menos una dirección IPv4 y un segmento de red IPv6 de 32-bits

Conversión de datagramas IPv4/IPv6

Almacena mappings propios de NAT

[Ejemplo en video](https://www.youtube.com/watch?v=EhCzKyojkNs&ab_channel=SunnyClassroom)

**DNS64**

Genera registros AAAA a partir de direcciones IPv4

Ejemplo de Traducción IPv4/IPv6: NAT64

P: prefijo de traslación usado en la intranet v6 (usa well known): 64:ff9b::/96

C6: dirección (IPv6) del cliente C

S4: dirección IPv4 del server S

N4: dirección IPv4 pública del NAT64

Configuración de ruteo en la intranet v6: si dirección comienza con P, ir a NAT64.

1- C solicita al DNS64 la dirección v6 asociada al servidor de nombre S.

2,4- DNS64 pide a un servidor DNS, los registros A y AAAA correspondientes a S

3,5- DNS64 recibe solo registro A (S es solo v4), con la dir v4 S4

6- DNS64 genera a partir de S4 , una dirección v6. La dirección generada es una "IPv4 converted IPv6 addresses" que se obtiene agregando el prefijo (P-S4)

7- DNS64 envía un registro AAAA con dire cción P-S4 a C

8- C envía datagram con dir origen=C6, dir destino= P-S4. El datagram es ruteado a NAT64, ya que tiene el prefijo P

9- NAT64 hace lo siguiente:

a-traduce campos de datagram v6 a datagram v4

b-obtiene la dirección de destino del v4 a generar, la saca de la de destino v6 (dirS)

c-La dirección de origen del datagram v4, es su v4: N4

d-Crea un mapping NAT, asociando el destino P-S4 y el origen C6 (para esto usa ports además de direcciones)

10-NAT64 envía el datagram generado a S4

11-S contesta con datagram v4, dir origen=S4, dir destino=N4

12- Cuando llega la respuesta de S a NAT64:

a- Traduce campos de datagram v4 y genera datagram v6

b- Reemplaza la dirección de origen v4 (S4) por P-S4

c- Reemplaza la dire. de destino v4 (N4) por la que tiene en la tabla de mapping (C6)

13- NAT64 envía el datagram en la red v6, dirigido a C

## 

## 

## Introducción al Nivel de Red

El nivel de red o capa de red, según la normalización [OSI](https://es.wikipedia.org/wiki/Modelo_OSI),​ es un nivel o capa que proporciona [conectividad](https://es.wikipedia.org/wiki/Conectividad) y selección de ruta entre dos [sistemas de hosts](https://es.wikipedia.org/w/index.php?title=Sistema_host&action=edit&redlink=1) que pueden estar ubicados en redes geográficamente distintas. Es el tercer nivel del [modelo OSI](https://es.wikipedia.org/wiki/Modelo_OSI) y su misión es conseguir que los datos lleguen desde el origen al destino aunque no tengan conexión directa. Ofrece servicios al nivel superior (nivel de transporte) y se apoya en el nivel de enlace, utiliza sus funciones.

Para la consecución de su tarea, puede asignar direcciones de red únicas, interconectar subredes distintas, encaminar paquetes, utilizar un control de congestión y control de errores.

Servicios no orientados a la conexión (CLNS): Cada paquete debe llevar la dirección destino, y con cada uno, los nodos de la red deciden el camino que se debe seguir. Existen muchas técnicas para realizar esta decisión, como por ejemplo comparar el retardo que sufriría en ese momento el paquete que se pretende transmitir según el enlace que se escoja dependiendo del tipo de red.

Servicios orientados a la conexión (CONS): Sólo el primer paquete de cada mensaje tiene que llevar la dirección destino. Con este paquete se establece la ruta que deberán seguir todos los paquetes pertenecientes a esta conexión. Cuando llega un paquete que no es el primero se identifica a que conexión pertenece y se envía por el enlace de salida adecuado, según la información que se generó con el primer paquete y que permanece almacenada en cada conmutador o nodo.

**Servicio provisto**

* Transporte de paquetes (datagramas) entre procesos de nivel transporte
* Independiente de la topología e Independiente de la tecnología
* Direccionamiento uniforme independiente de por donde haya que pasar

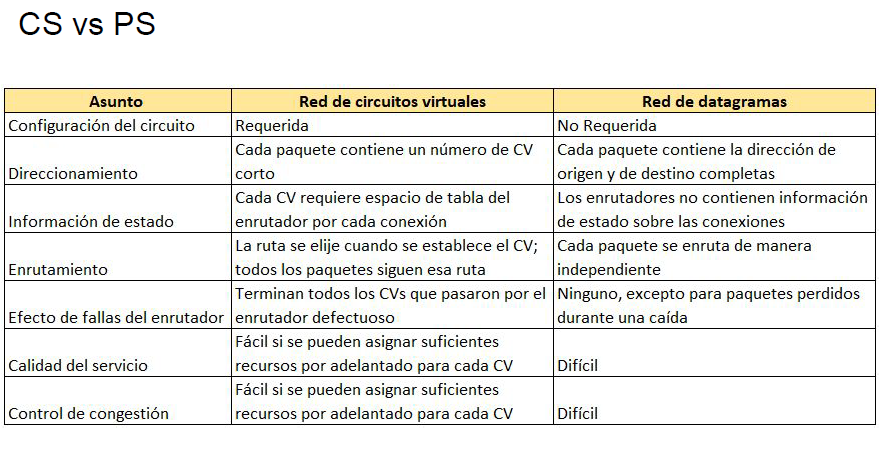
**Funciones del nivel**

* **Reenvío de paquetes**: determinación del próximo equipo a quien entregar un paquete
* **Ruteo de paquetes**: determinación del camino que deben seguir los paquetes para llegar a destino
* Control de congestión: evitar que la red colapse por tráfico excesivo
* Calidad de servicio: proveer características de envio (ancho de banda throughput, errores, etc. de acuerdo a lo que solicitan las aplicaciones
* Seguridad: autenticación de interlocutores, cifrado de la información
* Cómo cobrar a los equipos por el uso que hacen de la red
* Adaptación del tamaño del paquete soportado al MTU de las redes
* Establecer, controlar y terminar conexiones a nivel de red (ofrecidas a N4). Solo en el caso de prestar servicio orientado a conexión.

Tipos de Organizaciones

* Organización basada en circuitos virtuales
  + Circuitos virtuales: En una red de circuitos virtuales dos equipos que quieran comunicarse tienen que empezar por establecer una conexión. Durante este establecimiento de conexión, todos los routers que haya por el camino elegido reservarán recursos para ese circuito virtual específico. Todos los paquetes de esa conexión ya tendrán un comportamiento predefinido en los routers.
* Organización basada en datagramas
  + Datagramas: Cada paquete se encamina independientemente, sin que el origen y el destino tengan que pasar por un establecimiento de comunicación previo. Son servicios sin configuración previa o sin conexión, no es confiable ni orientado a la conexión.

Está orientado a la conexión



Reenvío y 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
* 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

Reenvió: 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

Ruteo: Consiste en determinar, desde un nodo dado, como llegar al destino. Origen y destino pueden estar en distintas subredes. Realizado manualmente o a través de aplicación (protocolos de ruteo)

Granularidad

* Datagrama
* Circuitos virtuales

Tipos

* Source routing
* Hop by hop routing

Componentes de la función de ruteo

Protocolos de ruteo

* Difunden la información necesaria para ruteo

Base de datos de información de ruteo (RIB)

* Información en cada router acerca de las rutas a los distintos destinos, con cierta(s) métrica(s)

Algoritmos de ruteo

* Usan la información de la base de datos para determinar (cálculo en background) las rutas a seguir para cada destino. El reenvío de los paquetes (en tiempo real) se hará en base a estas tablas (FIB).

Clasificación de los algoritmos de ruteo

Según dónde se toma la decisión

* Centralizados
* Descentralizados

Según cómo se actualizan las tablas de ruteo

* Estáticos
* Dinámicos

Según la sensibilidad a la carga

* Adaptativos
* No Adaptativos

## 

# Nivel de Aplicación

Arquitecturas generales de aplicación:

* p2p: como torrent o gnutella para compartir archivs, no hay servidores, dificiles de administrar y muy fácil de escalar.
* cliente servidor: los clientes no se conectan, son faciles de administrar dificiles de escalar. Cada servidor tiene una ip permanente mientras que los clientes son dinámicos. ej: correo electronico
* Arquitecturas hibridas: hay un registro en un servidor, y luego la conexion pasa a ser de a pares. Ej: google meet, whatsapp, etc

**Comunicación de procesos:**

Siempre tendremos un cliente y un servidor. El cliente es quien la inicia y el servidor responde.

Un ejemplo de cliente servidor: mi navegador web queriendo conectarse con el servidor de google.

Un ejemplo de p2p: el cliente es el que busca el archivo y servidor quien se lo entrega

En estos contextos de nivel de aplicaciones se dice que los mensajes que se envian los procesos, viajan por un socket. Estos socket son puntos de comunicación entre las dos partes que utilizan un medio de transporte “encapsulado” (abstraido).

Los desarrolladores vemos a los sockets como APIs que nos ofrece cada lenguaje, a fin de enviar mensajes a través de la web.

Para que un proceso reciba un mensaje, se debe poder identificar: no solo con la dirección del equipo, sino con el id de proceso ya que un equipo puede correr muchos procesos.

ADEMÁS para comunicarme con un proceso también necesito un número de puerto que en general esta establecido por defecto (dicho número de puerto es positivo de 16bits) (por ejemplo HTTP usa el puerto 80, y mail usa 25)

**Protocolos de la capa de aplicación:**

definen que tipos de mensaje voy a tener: su formato sintaxis semántica, y reglas de flujo

cualquier de estos protocolos pueden ser publicos o privados (la mayoría son publicos).

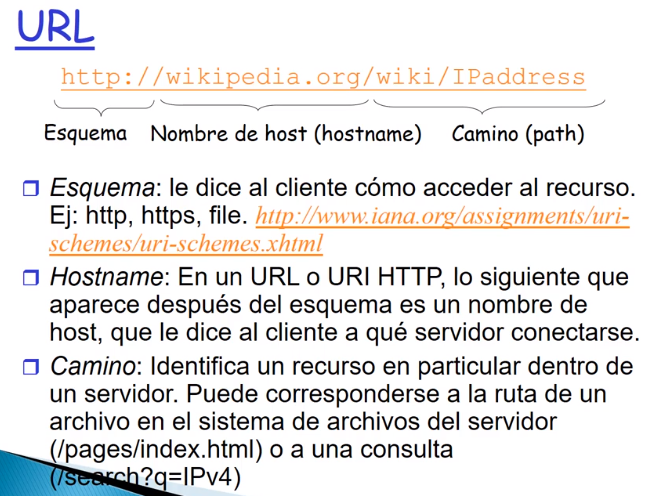
Luego nuestra aplicación también debe elegir que servicio(protocolo) de transporte a utilizar

Esto dependera de que querramos hacer con nuestra aplicación, por ejemplo si buscamos confiabilidad o rapidez, o gran taza de datos.

Un caso muy simple y usado es el de Web y HTTP que utiliza TCP.

Hay que saber que un protocolo es una herramienta que tiene una aplicación web, la web tiene por su parte estandars de documentos como lo es HTML, controlando además otro tipo de objetos de las paginas.

Más en detalle sobre HTTP: es un modelo cliente-servidor **simple** para aplicaciones.

Como podemos interpretar el cliente es nuestro browser, nuevamente.

Usa TCP, y cuando seá solicitan los componentes se habia preestablecido la conexion entre las partes. Una vez todo realizado se cierra la conexion y NO SE GUARDA UN ESTADO de la conexion en el servidor. La desventaja que nos trae esto es que se puede llegar a estados inconsistentes y dessincronizados cuando alguna de las partes pierde la conexion.

## Conexiones HTTP

Las conexiones entre el cliente http y el servidor pueden darse de dos tipos: **no-persistentes y persistentes** en el tiempo. Cuando el desarrollador web sabe que tipo de conexion va a darse, debe establecer qué tipos de solicitudes va a realizar teniendo en cuenta que **hay por debajo un protocolo TCP**. La diferencia está en utilizar una conexión http (tcp) para cada objeto o una única para todos.

**Conexiones no-persistentes:** se establece una conexión con el servidor por objeto. Es decir, si tuviéramos 10 objetos que queremos visualizar desde el browser, entonces se establecerá una conexión por el servidor con los requerimientos y todo 10 veces.

Así aparece el RTT: es el tiempo ocupado desde el envio de un paquete pequeño desde cliente al servidor y hasta su regreso (se cuenta hasta que un objeto/archivo empieza a llegar). Es decir: es el tiempo de ida y vuelta

**Conexiones persistentes:** las no-persistentes requiere de al menos de dos RTT por objeto.

En http 1.1 aparece esta variante persistente, donde el servidor deja las conexiones abiertas con el cliente para mantener el flujo sin tanto RTT. Si bien la conexión no se establece devuelta, si hay que hacer una petición por objeto, reduciendo RTT a 1 por c/objeto.

A su vez, dentro de las persistentes se puede tener **con pipelining o sin pipeline**.

Sin pipelining se envia un requerimiento nuevo cuando el anterior termina (son secuenciales), con el pipelining se permite la descarga en paralelo de los objetos

Cualquier error que aparezca con número 400..499 suele ser un error del servidor

## Mensajes HTTP

Dos tipos de mensajes HTTP: requerimiento y respuesta

Mensaje de requerimiento HTTP: ASCII (es decir, formato legible)

GET /somedir/page.html HTTP/1.1 //linea de requerimientos, con métodos

Host: [www.someschool.edu](http://www.someschool.edu) //nuevamente el host

User-agent: Mozilla/4.0

Connection: close //no es persistente

Accept-language:fr //el idioma

//linea vacia

Mensajes HTTP de respuesta:

HTTP/1.1 200 OK //linea de status del protocolo

Connection close //desde aca es encabezado

Date: Thu, 06 Aug 1998 12:00:15 GMT

Server: Apache/1.3.0 (Unix)

Last-Modified: Mon, 22 Jun 1998 …...

Content-Length: 6821 [bites]

Content-Type: text/html

data //esto es el HTML, imagen, o cualquier objeto solicitado

**Códigos de respuesta del servidor al cliente**

1xx - Informativo: La solicitud está en curso o hay otro paso que dar.

Ej: 101 Switching Protocols

Indica el protocolo al que está cambiando el servidor según lo solicitado por un cliente que envió el mensaje, incluyendo en el encabezado de request: Upgrade.

2xx - Éxito: La solicitud tuvo éxito. El servidor está enviando los datos que solicitó el cliente.

Ej: 200 OK

request exitoso, objeto requerido es incluido luego en mensaje

3xx - Redireccionamiento

El servidor le está diciendo al cliente un URI diferente al que debe redirigirse. Los encabezados generalmente contendrán un encabezado de ubicación con el URI actualizado. Diferentes códigos le dicen al cliente si un redireccionamiento es permanente o temporal.

Ej: 301 Moved Permanently

Se movió el objeto requerido, nueva ubicación es especificada luego en el mensaje (Location:)

4xx — Error “del Cliente” en el servidor

El servidor no entendió la solicitud del cliente, o no puede o no

quiere procesarla. Los diferentes códigos le dicen al cliente si fue

un URI incorrecto, un problema de permisos u otro tipo de error.

Ejs: 400 Bad Request y 404 Not Found respectivamente

Requerimiento no entendido por el servidor

Documento no encontrado en servidor

5xx — Error del Servidor

Algo salió mal del lado del Servidor.

Ej: 505 HTTP Version Not Supported

**Web caches (también servidores proxy)**

Objetivo: satisfacer el requerimiento del cliente sin involucrar al servidor destino.

Usuario configura el browser: Acceso Web vía cache Browser envía todos los requerimientos HTTP al cache Si objeto está en cache: cache retorna objeto Si no, cache requiere los objetos desde el servidor Web, los almacena y retorna el objeto al cliente.

Caché vs Proxys

La idea del **cache** es almacenar “localmente” datos ya solicitados y así poder acceder a éstos más rápidamente en el futuro.

Un problema que debe atender el cache es la

obsolescencia que puede tener los datos locales.

El caché puede usar tiempos de expiración, o consultar

a la fuente por vigencia del dato local.

Un **proxy** es un servicio que consiste en realizar una solicitud a pedido de otro.

¿Les ha pasado que para algunas cosas ustedes desean pedir a otro enviar un mensaje por ustedes?

Por ejemplo podemos usar proxy para acceder a servicios externos de una intranet, para que desde fuera no se sepa qué computadores hay dentro. El origen es siempre el mismo.

Cache actúa como cliente y servidor. Típicamente el cache es instalado por ISP (universidad, compañía, ISP residencial)

Por qué Web caching?

Reduce tiempo de respuesta a las peticiones del cliente.

Reduce tráfico en el enlace de acceso al ISP.

Internet densa con caches permite a proveedores de contenido “chicos” (no $$) entregar contenido en forma efectiva.

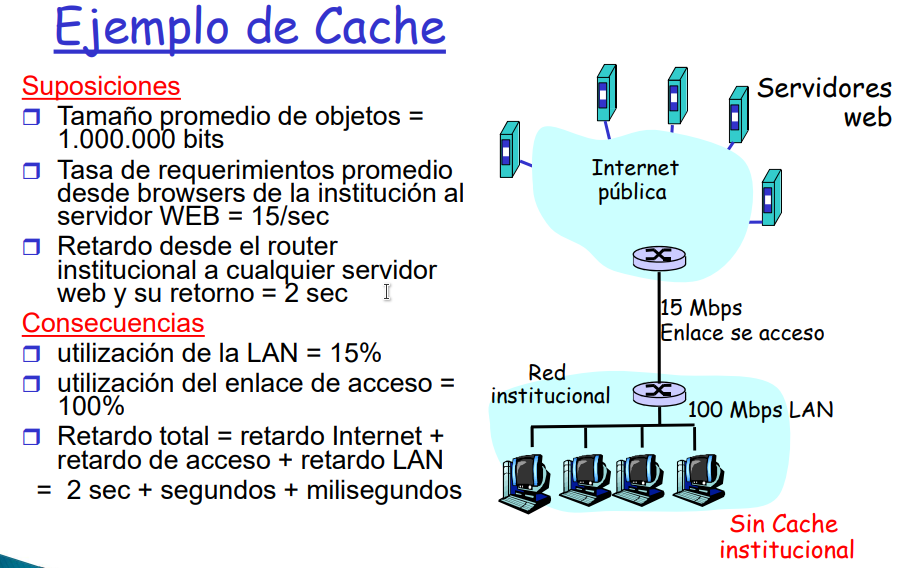
Caché y HTML

se utiliza un Get Condicional

Objetivo: verificar que el cache tiene la versión actualizada de un objeto

Cache: especifica la fecha de la copia en el requerimiento HTTP If-modified-since:

Servidor: responde sin el objeto si la copia de la cache es la última. : HTTP/1.0 304 Not Modified



## DNS

el domain name system es un protocolo de la capa de aplicación con una base de datos distribuida en el planeta y en los diversos ISPs **no centralizado**, tanto para evitar grandes daños por fallos, mantenimiento, grandes retardos de acceso por distancias, e inviabilidad del trafico. Es utilizado entre los browsers antes de hacer la solicitud http pide la dirección que corresponda.

También puede proveer alias a un nombre real o canónico de una dirección.

Mismo para los servidores de correo

La idea es dejar la complejidad en los puntos más lejanos de la internet.

hay 13 servidores root DNS en todo el mundo, y varias veces son salteads

los Top Level Domain server son responsables de cualquier .com,org,net,edu… y cada pais el suto

los autoritativos son propios de las organizaciones y se encargar de mantener el hostname/servidor -- ip de esta, cuando quiera hacerla publica

los servidores de nombre local no tienen que estar estrictamente en la jerarquia, pero si es que lo estan funcionan como caché (es decir, proxy) y trata de responder por si solo a la consulta de cualquier host. Si no puede, pasa la pregunta para arriba.

el puerto por defecto de DNS es el 53

#### Tipos de consultas DNS

iterativas: voy preguntando a cada intermediario por la dirección del siguiente a consultarle

recursivas: paso la tarea de consultar a mi sucesor correspondiente. estos tienen que guardar más información y fuerzan más a la red/servidores en momentos demandantes

necesita 2\*( 1 + cdadServidoresIntermedios) mensajes

La caché en los servidores siempre es guardada, por cada consulta o mappeo realizado, esta desaparece despues de un tiempo para no guardar basura ante cambios. Normalmente se guardan las direcciones de los TLD para ahorrar la consulta a los root.

Formato de los registros guardados: RR:(name, value, type, ttl)

type me define lo que hay en name y value. puede ser:

A: clásicos, tipo diccionario a dirección ipv4

AAAA: igual, pero para ipv6

NS: name sera un dominio, y value: una dirección de un servidor TLD o autoritativos

CNAME: permite definir alias a un determinado dominio en “name” y el real iria en “value”

MX: para los correos, el value es el nombre canonico de un servidor de correo, y name el alias

#### Comandos y Práctica en DNS:

comando dig: busca la ip dado un domino

dig <dominio> [<type>] | dig <@servidor dns> <dominio> | dig <dom> +trace

el +trace pide que nos den toda la ruta de esa direccion , por todos los servidores.

### 

## Correo Electrónico

es de las primeras aplicaciones en toda la internet

En esta aplicación se tiene :

Clientes como gmail, outlook, thunderbird, etc.

Y servidores de correo que se comunican con el protocolo SMTP(Simple Mail Transfer Protocol). Cada servidor tiene una cola de “pendientes a enviar” y diferentes casillas de usuario

#### SMTP (RFC 5321)

es protocolo que utiliza TCP (orientado a la conexión) y conexiones persistentes

utiliza el puerto 25 en el servidor.

La conexion de realiza servidor a servidor sin intermedios

TRES PARTES:

handsshaking(uno TCP y otro SMTP) transferencia cierre

Interaccion de Comandos y Respuestas:

comando de tecto ascci de 7 bits

respuesta: código de status

todos los mensajes terminan con <Enter> “.” <Enter>

Comparación con HTTP:

HTTP: pull (saca contenido desde servidor)

SMTP: push (pone contenido en servidor)

Ambos tienen interacción comando/respuesta en ASCII, y tienen códigos de estatus

HTTP: cada objeto es encapsulado en su propio mensaje

SMTP: múltiples objetos son enviados en un mensaje multiparte (seccionadox)

#### Protocolos de Acceso al Servidor de Correo

SMTP: permite envío y almacenamiento de correo en servidor del destinatario

Protocolo de acceso a correo: permite extraer correo desde el servidor

POP: Post Office Protocol [RFC 1939] • autenticación (agent <-->server) y bajada

IMAP: Internet Mail Access Protocol [RFC 1730] • Más características que POP (IMAP es más complejo) • Permite manipulación de los mensajes almacenados en el servidor

HTTP: Hotmail , Yahoo! Mail, etc.

#### POP3

Es un protocolo extremadamente simple, de acceso que le permite a un usuario extraer correos desde el servidor

**1 Fase de autorización en el puerto 110**

Comandos del cliente: user: declara username pass: password

Respuestas del servidor: +OK -ERR

**2 Fase transaccional, cliente:xd puto**

list: lista números de mensajes

retr: extrae mensajes por su número

dele: borra

quit

3 Fase de borrado: el servidor borra los mensajes que el cliente de correo solicitó

- Ejemplo previo usa modo “bajar y borrar”.

Bob no puede releer el correo si cambia el cliente “bajada y conserva”:

obtiene copia de los mensajes en diferentes clientes.

POP3 no mantiene el estado de una sesión a otra (“stateless”)

#### IMAP

Puede mantener los mensajes en el servidor. Permite que el usuario organice sus correos en carpetas. IMAP mantiene el estado del usuario de una sesión a otra:

Nombre de carpetas

mapeo entre Ids (identificadores) de

mensajes y nombres de

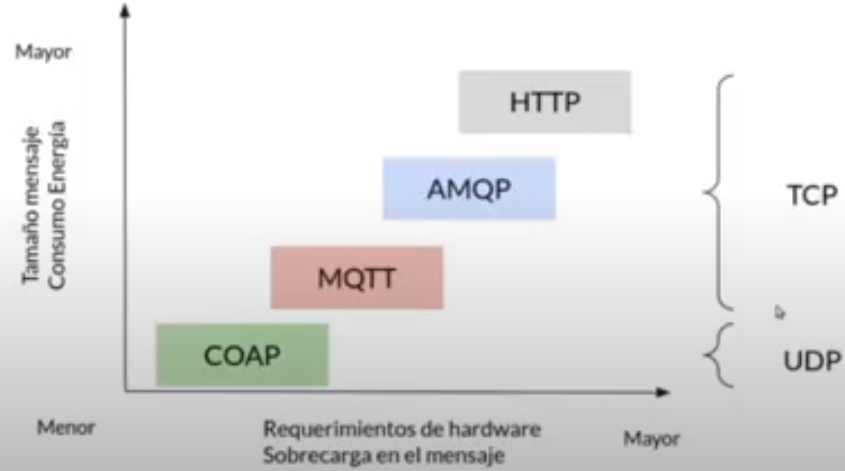
carpetas.

# 

# IoT y MQTT

iot es el concepto de la conexión de dispositivos a la internet, también de sistemas dispositivos vehículos conectados y trasmitiendo información.

Hay cuatro protocolos dependiendo del tipo de aplicación que desarrollamos



Vamos a profundizar en MQTT:

Se basa en el patrón Broker, donde un nodo publica una información, y el broker se encarga de distribuirla a los suscriptores.

Un nodo puede estar suscrito o publicar a uno o más nodos.

Suscripción multinivel con #: Me suscribo a todos los hijos

Suscripción al mismo nivel con +

ejemplos:

1 casa/plantaBaja/Cocina/temperatura

2 casa/+/+/temperatura

3 casa/#

1 Suscripción al sensor de temperatura

2 Suscripción a los dos sensores de temperatura

3 Suscripción a todos los sensores

● Protocolo desarrollado para redes de bajo ancho de banda, alta latencia y poco confiable.

● Soporta Calidad de servicio

●Disponibilidad (se han implementado de dos maneras) Forward paquetes a otra broker Soporta armar un cluster

●Implementa websocket para utilizar en las diferentes webs

● El encabezado es pequeño: 2 bits, y permite agregar funcionalidad

● Seguridad posee autenticación y TLS

Brinda autenticación usuario y password Implementa SSL ó TLS

● Basado en publish and subscribe

Es decir, cada nodo o dispositivo se suscribe o pública en un “topico” determinado (un topico es un conjunto no vacío de la jerarquia establecida en arbol, por ejemlo: CASA como raíz de una jerarquía, y las habitaciones sus hijos )

● Último testamento

El cliente puede definir un mensaje Last Will and Testament (LWT) y un tópico para publicarlo Cuando dicho cliente para un tiempo sin responder, el servidor publica ese mensaje en ese tópico

TIPOS DE MENSAJES MQTT

CONNECT Enviado por el cliente al servidor para establecer conexión

CONNACK El servidor acepta la conexión

PUBLISH Mensaje que representa una publicación

PUBACK Utilizando en QoS 1, en respuesta al mensaje PUBLISH

PUBREC Primer mensaje en QoS 2 flujos de mensajes

PUBREL Segunda parte QoS 2 flujos de mensajes

PUBCOMP Última parte QoS 2 flujos de mensajes

SUBSCRIBE Pedido de suscripción a un tópico

SUBACK Aceptación de suscripción al mensaje

UNSUBSCRIBE Usado para desuscribir a un topico

UNSUBACK Aceptación de desuscripción

PINGREQ Hertbeat: mensjae de solicitud

PINGRESP Heartbeat: mensaje de respuesta

DISCONNECT Mensaje de desconección

# 

# Otros

**Ruteo Link State:** este protocolo de ruteo e basa en que un router o encaminador comunica a los restantes nodos de la red, identifica cuáles son sus vecinos y a qué distancia está de ellos. Con la información que un nodo de la red recibe de todos los demás, puede construir un "mapa" de la red y sobre él calcular los caminos óptimos.

**REGISTROS DNS**

DNS: es una base de datos distribuida que almacena registros de recursos (resource records, RR)

Formato RR: (name, value, type, ttl)

Type=A

name es un hostname (nombre real o canónico)

value es una dirección IP

Type=NS

name es un dominio (e.g. foo.com) value es el nombre de host de un servidor DNS autoritativo que sabe cómo obtener las direcciones IP de este dominio.

Type=CNAME

name es un alias para algún nombre real (indicado en type A)

www.ibm.com es realmente servereast.backup2.ibm.com

value es el nombre real (canónico)

Type=MX

value es el nombre canónico de un servidor de correo que tiene un alias asociado con name

Backward Learning es un protocolo para que un router pueda completar su tabla de ruteo. Las rutas se completan en función de los paquetes que pasan por él, provenientes de otros nodos. Puede incorporar información acerca de costos de llegar a un nodo. (hops)

En ruteo avanzado de Linux, si un datagram no es consumido por ninguna regla

de ruteo, se envía en función de la tabla de ruteo default.

Las últimas dos reglas por las que están afectados los datagramas en ruteo son

consultar la tabla main y la tabla default, por lo que si no fue accionada por la tabla main,

esta es verdadera.