NIVEL DE TRANSPORTE.

Misión → GOBERNAR ACCESO MÚLTIPLE A LA RED.

A través de → PUERTOS.

Dos protocolos →
- UDP → no orientado a conexión y no fiable.
- TCP → orientado a conexión y fiable.

Las direcciones IP no son suficientes, asique usan "direcciones de nivel de transporte" llamados puertos.

Los puertos siempre se añaden en la cabecera del nivel de transporte. Tanto como el de origen como de destino.

Puertos menores que 1024 son reservados.

Servidor

Al abrir escucha esperando recibir mensajes de un determinado protocolo de nivel de transporte y en un determinado puerto.

Cliente.

Envía el 1er mensaje a la app servidor utilizando el protocolo de nivel de transporte que usa servidor y enviando los mensajes a la IP y puerto que está escuchando el servidor.

IMP → Para que la interacción servidor-cliente funcione es necesario arrancar primero Servidor. y después Cliente!

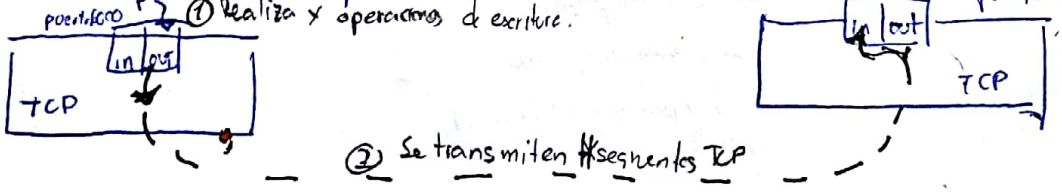
PROTOCOLO UDP

Se encapsulan dentro de los datos de un datagrama IP.

- No fiable y no ordenado.
- Proporciona multiplexación.
- No controla el flujo
- Utiliza puertos.

PROTOCOLO TCP

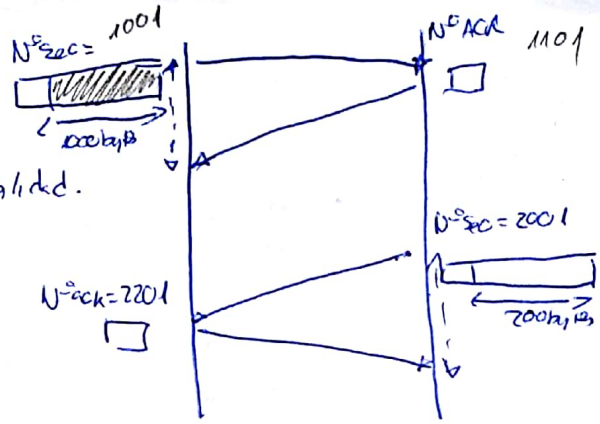
- Orientado a conexión: Fases de establecimiento, intercambio de datos y cierre conexión.
- Fiable - recepta siempre le llegan los datos sin pérdidas, ni duplicados ni desordenados.
- Envía los datos como flujo de bytes.



¡ CONEXIONES Full duplex: ambos lados pueden enviar datos simultáneamente!

SERVICIO ORIENTADO A CONEXIÓN.

① Establecimiento de conexión.
② Intercambio de datos
③ Cierre de conexión



SERVICIO FIABLE

Primer nivel (de abajo a arriba) q.e proporciona fiabilidad.
Arregla las posibles pérdidas y descarte producido

Funcionamiento básico.

- Los segmentos con datos llevan un número de secuencia.
- El receptor debe mandar asentimientos (ACKs)
- Para cada segmento con datos transmitidos se opera un plazo de tiempo a que llegue su ACK.
  Si este plazo espira sin haber recibido el ACK se retransmite el segmento.
- Para asentimientos y retransmisiones se utiliza un protocolo de ventana.
- El receptor reordena segmentos y descarta duplicados

(Como es full duplex cada lado usa su propio nº de seq)

*Formato segmento TCP.

- Puertos → origen y destino del segmento.
- Longcabecera: → Sin opciones (20 bytes)
- Checksum → Obligatorio en TCP

*Flags

- SYN: Establecimiento conexión.
- FIN: Fin conexión.
- ACK: Asentimiento.
- RST: Situación error.
- PSH: Receptor debe entregar los datos a la app.
- URG: Datos urgentes

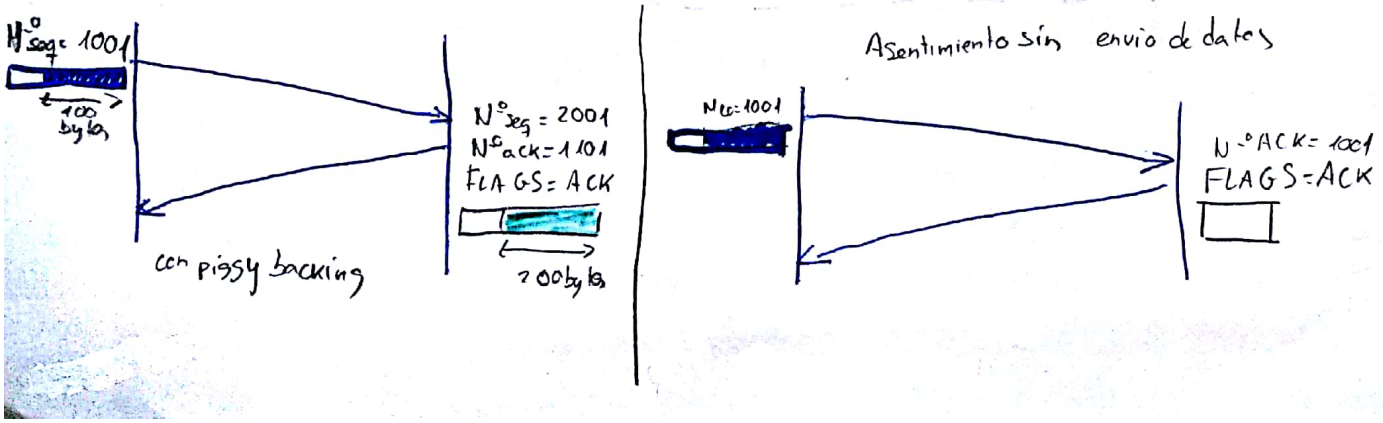
* Nº de secuencia

- Numera bytes, y NO segmentos
- Cada segmento con datos lleva un nº de secuencia, de 32 bits.
- Al iniciar conexión se elige un secuencia num aleatorio para que no se confundan segmentos aún en tránsito. de conexiones anteriores.

* Número de asentimiento

- piggy backing → se envían los datos y el asentimiento de los datos recibidos
- Si el lado que ha recibido datos no tiene nada que enviar, construye un segmento (solo con cabecera TCP) donde envía el nº de asentimiento correspondiente.

¡¡NP! Cada lado de la conexión utiliza sus números de secuencia (partiendo del inicial) y asiente los que está usando el otro extremo.

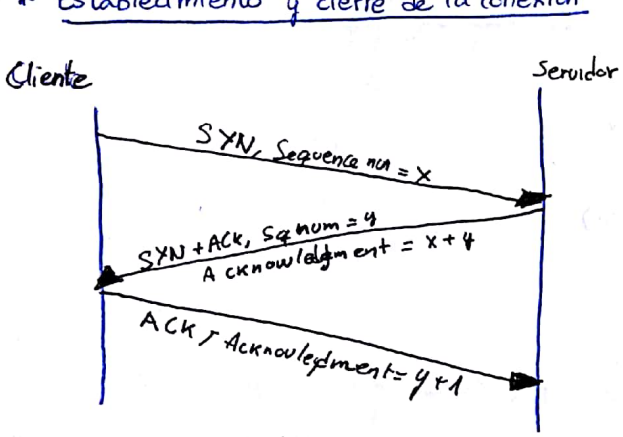


PROTOCOLO TCP

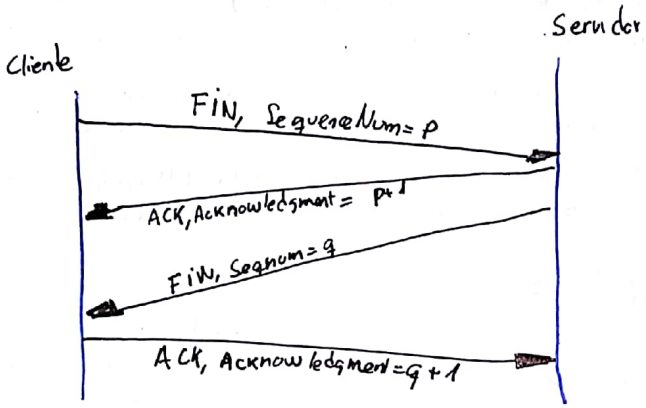
* Ventana anunciada (o ventana de flujo)

- Control de flujo → protocolo de ventana que coordina el envío de segmentos de datos.
- El receptor indica en el *AdvertisedWindow* el nº de bytes (a partir del asentimiento) que está dispuesto a recibir del emisor.
- El emisor puede transmitir estos bytes aunque no reciba asentimientos, una vez transmitidos tendrá que parar hasta recibir nuevos asentimientos del receptor
- Hay dos ventanas de flujo diferentes, una para cada sentido de la comunicación

* Establecimiento y cierre de la conexión



ESTABLECIMIENTO.



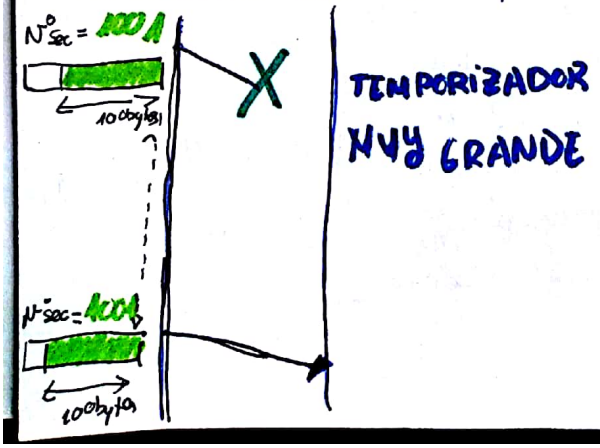
* Sondas de ventana

- Si el receptor no lee los datos recibidos, irá llenando su buffer de recepción e irá anunciando un valor de ventana cada vez menor (incluso llegando a cero)
- Si el emisor recibe AdvertiseWindow = 0, no puede seguir enviando datos. El receptor ha "cerrado la ventana"
- En esta situación el emisor va a enviar periódicamente un segmento con Sequence num igual al último asentido y de longitud 0 bytes para provocar asentimientos del receptor. Estos segmentos son Sondas de ventana.
- Cuando la app en el lado receptor lea los datos que se estén recibiendo en el buffer irá vaciándose y la implementación de TCP en el lado del receptor podrá guardar nuevos datos en dicha buffer y se lo comunicará al emisor rellenando el valor correspondiente en el campo AdvertisedWindow de los asentimientos que estén enviando como respuesta a las sondas de ventana.

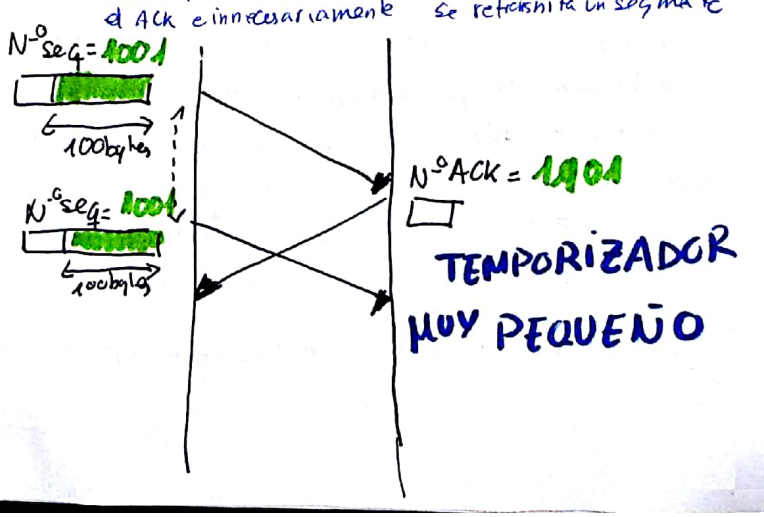
PLAZOS DE RETRANSMISIÓN

Cuando se envía un segmento se arranca un temporizador para esperar su asentimiento.
Transcurrido el plazo marcado en el temporizador (timeout), si no se ha recibido el ACK se retransmite

- Si el plazo es muy grande puede tardarse mucho en retransmitir un segmento que se ha perdido



- Si el plazo es muy pequeño, puede que no de tiempo a que se reciba el ACK e innecesariamente se retransmita un segmento



* Retransmisión adaptativa

▸ Para cada segmento se calcula el tiempo de ronda (Round-Trip-Time, RTT): tiempo entre que se envía el segmento y se recibe el asentimiento. Se va tomando su media en el tiempo.

▸ Estas medidas de RTT se calculan para cada pareja (segmento/ACK)

▸ El timeout se calcula en función de cada medida de RTT y la varianza de dichas medidas. Suele ser aproximadamente igual al doble del RTT.

Se dobla el timeout cuando se retransmite ← Exponential Backoff

■ OPCIONES

* Extensiones de TCP

Se implementan como opciones de la cabecera, que van en el segmento detrás de los campos fijos de la cabecera.

Las opciones más habituales son:

▸ Marcas de tiempo (timestamps)
Almacena la hora del envío en segmentos enviados. El receptor lo copia al enviar un ACK.

▸ Extensión del espacio en números de secuencia.
Timestamp + número de secuencia como identificador de segmento

▸ Escalado de la ventana anunciada (Window Scale)

▸ Tamaño máximo de segmento (MSS: Maximum Segment Size)

- Window Scale

Esta opción aumenta el tamaño de ventana.

▸ En el segmento SYN se incluye esta opción junto con un factor.

▸ Si el receptor está dispuesto a usar esta opción, en su segmento SYN+ACK incluirá también esta opción junto con un factor.

Ventana Anunciada Real = 2^{factor} x Advertised Window

Si el factor es 1, la ventana anunciada real es el doble del valor que viaja en el campo Advertise dWindow de la cabecera TCP

- Maximum Segment Size (MSS)

El MSS es el máximo tamaño de un segmento que no causa fragmentación. (Suponiendo que las cabeceras IP y TCP no tengan opciones. Ej MSS = 1500 − 20 (cabecera ip sin opciones) − 20 (cabecera TCP sin opciones) = 1460 bytes

En Ethernet el máximo tamaño de un datagrama IP es 1500 bytes.

Cada lado de la conexión hace segmentos de tamaño menor o igual a su MSS

Si se usa la opción TCP, en el segmento SYN cada lado incluye su MSS para indicar al otro lado que haga segmentos de tamaño menor o igual a ese valor.

Si ambos lados de la conexión usan esta opción, la consecuencia es que ambos lados usarán segmentos con la parte de datos de tamaño igual al menor de los dos MSS

- PATH MTU Discovery

MSS no garantiza que no vaya a haber fragmentación: ~~En algún salto intermedio~~

Para evitar fragmentación también se usa Path MTU Discovery.

· Al principio de una conexión cada lado elige como tamaño el menor entre su MSS y el MSS del otro extremo.

· A los datagramas IP que contienen los segmentos se les activa el flag DF (Don't fragment).

Si un router del camino no puede reenviar un datagrama sin fragmentarlo, y ese datagrama tiene el flag DF, descarta el datagrama y envía a su origen un ICMP

▸ Es un ICMP de destino inalcanzable por necesidad de fragmentación y está activado el flag DF (Icmp tipo=3 Código=4)

▸ El ICMP incluye el máximo tamaño del datagrama que le permitiría al router no fragmentar.

· El origen al recibir un ICMP de este tipo, disminuye el tamaño del segmento.

DOMAIN NAME SYSTEM (DNS)

Sistema de nombrado de máquinas y correspondencia entre dichos nombres y direcciones IP -> DNS

Es protocolo del nivel de aplicación funciona sobre UDP y TCP

DNS: Base de datos distribuida que se consulta según el modelo cliente/servidor.

Los nombres de las máquinas se agrupan en dominios.

Los dominios se organizan en forma de árbol.

El nombre completo de una máquina (FQDN, Fully Qualified Domain Name) incluye el nombre de la máquina y el nombre del dominio en el que se encuentra.

hielo.gsyc.urjc.es.

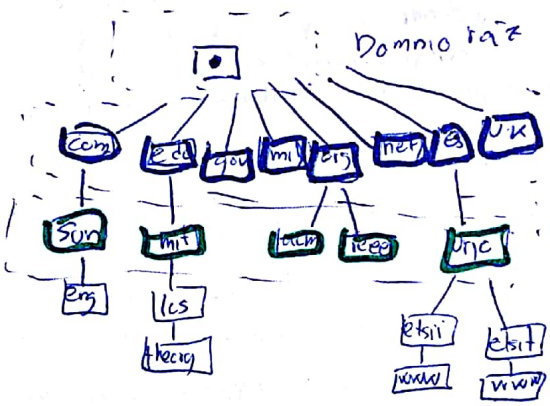
Estrictamente, un FQDN termina siempre en el carácter "." (aunque normalmente se omite excepto en los mapas DNS).

DOMINIOS EN DNS

* Jerarquía de dominios

- Dominio raíz (root domain o dominio ".."): Gestionado por ICANN (Internet Corporation for Assigned Names and Numbers). Los 13 servidores llamados root nameservers
- Dominios de primer nivel (TLDs, Top Level Domains):
  - Dominios genéricos tradicionales (com, edu, gov, mil, org, net, int)
  - Dominios genéricos modernos (aero, info, pro, jobs...)
  - Dominio para la infraestructura DNS (arpa)
  - Dominios por código ISO del país: (uk, ar, de, es, jp)
- Dominios de segundo nivel
- Dominios de tercer nivel
- ...

* Árbol de dominios



* Asignación de dominios

- Los TLDs los asigna la ICANN
- La asignación de nombres de dominio de segundo nivel (subdominios.es, .com.org) está gestionada por organismos denominados "registrars"
- Algunos subdominios están gestionados por varios "registrars" en régimen de competencia.

* Dominio directo y Dominio inverso

DOMINIO DIRECTO → Proporciona para cada nombre una dirección IP.

DOMINIO INVERSO → Proporciona para cada dirección IP un nombre

Respecto al dominio inverso...

- También se conoce como dominio in-addr.arpa.
- Los elementos del dominio inverso son las direcciones de red construidas invirtiendo los números que la componen y terminando en in-addr.arpa. Ej: la red 138.1170.0 es el dominio inverso 117.138.in-addr.arpa.
- Esta inversión de los números de las direcciones IP se realiza para mantener la misma estructura jerárquica de los nombres de dominio:
  - En los nombres de dominio porciones del nombre situadas más a la izquierda representan entidades más específicas, mientras que en las direcciones IP es al revés

■ RESOLUCIÓN DE NOMBRES

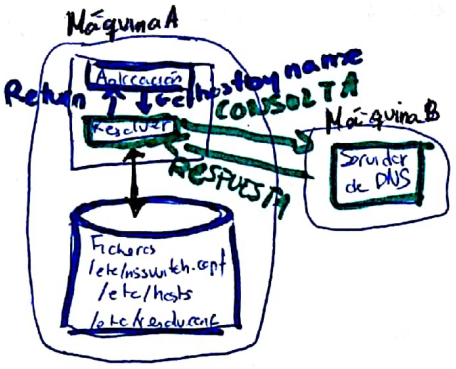
* Consulta de nombre desde las aplicaciones

Cuando una aplicación tiene un nombre de máquina y necesita su IP, consulta al DNS, invocando en su código llamadas a funciones como gethostbyname() o TO_IP()

Las aplicaciones van enlazadas con una librería de consulta al DNS

La consulta es: 1. Consulta el fichero /etc/hosts
2. Si no se resuelve consulta en un servidor DNS, cuya dirección IP está en /etc/resolv.conf.

El fichero /etc/nsswitch.conf determina si se consulta el fichero y/o el DNS y en qué orden.



○ Ejemplos

/etc/hosts

- En cada línea detrás de una dirección IP pueden aparecer uno o más nombres (con o sin dominio) separados por blancos, que quedan asociados a esa dirección
- Los nombres e IPs que aparecen pueden no tener nada que ver con los de DNS

/etc/resolv.conf

- La línea search incluye una lista de dominios separados por blancos
- Las líneas nameserver contienen la IP del servidor de DNS al que consultará la máquina (responderá funciones)
- Si hay + de una línea nameserver, se utilizarán (x orden) si los servidores anteriores no funcionan

NOTA Si un servidor responde que un nombre no existe, no se pregunta a otro.

/etc/nsswitch.conf:

La línea hosts lista lo que se consulta y en qué orden. - files: fichero /etc/hosts
- dns: servidor DNS según /etc/resolv.conf

* Servidor de DNS de un dominio

- La info relacionada con la resolución de nombres de un dominio determinado se guarda en un fichero que se llama mapa del dominio que contiene entre otros: - nombres de las máquinas del dominio con sus IP.
  - nombres de subdominios directos con las IPs de los servidores de DNS que sirven esos subdominios.
- El mapa de un dominio lo edita el administrador de sistemas de ese dominio y se encuentra almacenado en la máquina que funciona como servidor DNS de ese dominio. El mapa de un dominio puede encontrarse en otras máquinas que también sean servidores de DNS. Se dice que todos estos servidores sirven ese dominio.
- Un servidor de DNS que contenga varios ficheros de mapa de dominio servirá todos los dominios correspondientes a dichos ficheros.

* Consulta a un servidor DNS

Cuando un servidor DNS recibe una consulta puede ocurrir que:

1. El servidor de DNS sirva al dominio al que pertenece la consulta. → El servidor podrá leer el mapa de dominio directamente y generar una respuesta.
2. El servidor de DNS no sirva al dominio al que pertenece la consulta. → Dependiendo del modo de consulta:
   - 2.1. Responderá con la dirección IP de otro servidor de DNS que sirva un subdominio al que se refiere la consulta. Esta info la extrae de su mapa de dominios. (servidores del dominio raíz)
   - 2.2. preguntará a otro servidor de DNS para tratar de conseguir la respuesta. → Todos los DNS saben las IPs de los root (nameservers)

* Resolución de nombres en un servidor DNS

- Cuando un servidor S recibe una consulta para resolver un nombre (ejemplo: www.google.com)
  a. S. comprueba si el nombre pertenece a alguno de los dominios que sirve (si lo tiene) Si S lo sirve, busca el nombre en el mapa y devuelve la IP.
  b. Si el nombre no es de ninguno de los dominios que sirve S: 1. S pregunta a un servidor del dominio raíz que le contestará con la IP de un servidor DNS de TLD incluido FQDN (com en este caso)
2. S pregunta al servidor de DNS del TLD (com) que le responde con IP de un servidor de DNS del dominio de 2º nivel (google.com.)
3. Si el FQDN de la pregunta contuviera más dominios, se repite hasta que S obtenga la IP del serv. de dominio en el que reside la máquina x que pregunta.
4. S. pregunta al servidor del último nivel (google.com.) que por servir ese dominio, tendrá en sus mapas la IP pedida y se la daría S.
5. S devuelve a la IP pedida a quien se la pidió.

* Caché en el servidor DNS

Cuando un servidor en una búsqueda aprende un dato que no sabía lo guarda en una caché. Y si lo vuelve a necesitar lo saca de ahí en vez de preguntar.

Los mapas especifican cuánto tiempo (ttl) puede estar en una caché un dato que se saca de dicho mapa.

* Tipos de consultas recibidas por un servidor

- Consultas recursivas.
  - Obligan al servidor a hacer todas las consultas necesarias para encontrar la dir pedida.
  - Son las que hace un resolver
- Consultas iterativas.
  - Hacen que el serv. conteste con lo máx que sepa de esa búsqueda sin preguntar más.
  - Son las que hace DNS a otro DNS.

En cualquier consulta los servidores aprovechan cualquier información que tengan en la caché para responder directamente o para atajar en la cadena de búsqueda

Ej: Si un servidor recibe una consulta recursiva por www.google.com y ya tiene en su caché la IP de un servidor de com, empieza preguntando a él y no a un root nameserver.