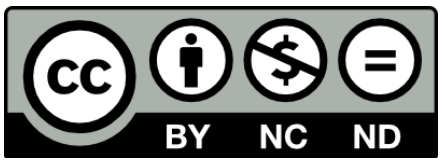
Fundamentos de Redes

Los Del DGIIM, losdeldgiim.github.io

Doble Grado en Ingeniería Informática y Matemáticas
Universidad de Granada



Fundamentos de Redes

Los Del DGIIM, `losdeldgiim.github.io`

Irina Kuzyshyn Basarab
José Juan Urrutia Milán

Granada, 2023-2024

Índice general

1. Introducción a los fundamentos de redes **9**

1.1. Sistemas de comunicación y redes 9

1.1.1. Motivación para usar redes 10

1.1.2. Topologías de redes 11

1.1.3. Clasificación de redes 12

1.1.4. Nomenclatura típica en figuras (Iconos) 12

1.2. Diseño y estandarización de redes 13

1.2.1. Modelo OSI vs TCP/IP 14

1.3. Terminología, conceptos y servicios 15

1.3.1. Retardos en la comunicación 17

1.3.2. Tipos de servicios 18

1.4. Internet: topología y direccionamiento 18

1.4.1. Organización topológica 19

1.4.2. Red Iris 19

1.4.3. Direccionamiento por capas 19

2. Capa de red **21**

2.1. Funcionalidades 21

2.2. Conmutación 22

2.2.1. Conmutación de circuitos 22

2.2.2. Conmutación de paquetes 23

2.2.3. Conmutación con circuitos virtuales 23

2.3. El protocolo IP 24

2.3.1. Direccionamiento 25

2.3.2. Network Address Translation (NAT) 28

2.3.3. Encaminamiento 32

2.3.4. Routing Information Protocol (RIP) 34

2.3.5. Open Shortest Path First (OSPF) 35

2.3.6. Cabecera IP 35

2.3.7. Fragmentación 36

2.4. Asociación con la capa de enlace: el protocolo ARP 37

2.5. El protocolo ICMP 38

2.6. Autoconfiguración de la capa de red (Dynamic Host Configuration Protocol (DHCP)) 39

3. Seguridad en redes **41**
3.1. Introducción 41
3.2. Cifrado 42
3.2.1. Cifrado simétrico 43
3.2.2. Cifrado asimétrico 43
3.3. Autenticación 44
3.3.1. Reto-respuesta 44
3.3.2. Intercambio de Diffie-Hellman 45
3.4. Funciones Hash 45
3.5. Firma digital y certificados digitales 46
3.5.1. Firma con clave secreta o Big Brother (BB) 46
3.5.2. Firma digital con clave asimétrica o Doble cifrado 47
3.6. Protocolos seguros 47
3.6.1. PGP (Pretty Good Privacy) 48
3.6.2. TLS, SSL 48
3.6.3. IPSec 49

4. Relaciones de Problemas **51**
4.1. Introducción 51
4.2. Capa de red 58

1. Introducción a los fundamentos de redes

Objetivos

- Conocer y comprender los principios básicos de las comunicaciones.
- Entender el diseño funiconal en capas de las redes y los conceptos y terminología fundamentales involucrados.
- Comprender desde un punto de vista teórico-conceptual el modelo de referencia OSI y su correspondencia con el modelo de capas usado en Internet.

Introducción

La arquitectura lógica de Internet está diseñada por capas. Veremos el modelo TCP/IP (aunque mencionaremos el modelo OSI):

Aplicación que hace uso de la red
Transporte **(TCP/UDP)**
Red **(IP)**
Enlace
Física

Tabla 1.1: Modelo de capas del protocolo TCP/IP.

Las dos últimas capas están implementadas en Hardware y las tres primeras en Software, también llamado Network Operating System (NOS). En la asignatura veremos las capas altas, las implementadas en software.

1.1. Sistemas de comunicación y redes

Definición 1.1 (Sistema de comunicación)**.** Es una infraestructura (hw + sw) que permite el intercambio de información. Un sistema típico es el siguiente:

- Tenemos una fuente y un transmisor en un mismo equipo (que es el que va a mandar la información). La fuente genera la información y el transmisor adapta la información al medio.
- Después tenemos el canal de comunicación, el cual produce errores: ruidos, interferencias, diafonías (cuando hay muchos cables en paralelo juntos, puede suceder que la información de un cable se meta en otro)...

- Al final tenemos un receptor y el destino (en un mismo equipo). El receptor adapta la información para el destino y éste espera los datos a recibir.

1.1.1. Motivación para usar redes

Para entender su uso hablaremos de la primera red de comunicaciones, que era una red de telefonía móvil. Cada usuario contaba con su línea de teléfono, que conectaba con una central de conmutación local, luego regional y luego nacional, la cual debía conectarse con la central local del destino. Se usaba conmutación de circuitos.

- Inicialmente se creaba un camino físico juntando cables, llamado circuito.
- Era ineficiente porque no se está hablando todo el tiempo, y los tiempos de silencio el circuito se desaprovecha.
- Era un problema de seguridad el mal funcionamiento de una central, pues dejaba a miles de teléfonos sin servicio.

Si ahora pensamos en ordenadores (o equipos más generales, móviles, PCs, portátiles, móviles...) en vez de móviles, y cambiamos las centrales de conmutación por routers, contamos con muchísimos caminos para conectar dos ordenadores, haciendo más segura la red.

En la actualidad ya no tenemos un camino físico, sino que son los routers quienes deciden por dónde enviar los paquetes y en qué momento. Estos tienen colas, lo que supone algo de retardo, pero tienen la ventaja de que se usa mucho mejor el canal y hay más seguridad, pues hay más de un camino.

Definición 1.2 (Red)**.** Sistema de comunicación con sistemas finales o terminales autónomos (con capacidad para procesar información) que facilita el intercambio eficaz y transparente de información. Concretamente tenemos:

- **Hosts:** sistemas finales o terminales autónomos. Son los que transmiten y reciben datos.
- **Subred:** infraestructura para el transporte de información, formada por líneas de transmisión y nodos o elementos de conmutación: routers y switches.

De una red esperamos:

- Autonomía.
- Interconexión.
- Intercambio de información con eficacia y transparencia.

En cuanto a medios de transmisión, originalmente se usaban cables de pares (pensados para transmitir 4 kHz, la media de la voz humana), luego cables coaxiales, que mejoraron mucho; y fibra óptica que puede transmitir sin interferencias, por lo que es el mejor medio guiado existente. Los cables trenzados son para distancias más cortas, Ethernet por ejemplo.

1.1.2. Topologías de redes

Dada una subred, su topología es el patrón de interconexión entre sus nodos. Las más relevantes las veremos a continuación.

En bus: Todos los nodos tienen acceso a un mismo medio, conocido como bus. Es la más sencilla, pero como el medio es común, todos intentan acceder y se producen colisiones.

En anillo: un círculo en el que tenemos los distintos nodos. Es similar al bus pues el medio es compartido. Una versión habitual es **token ring**, testigo de anillo, en el que se usa un testigo que se van pasando, de forma que así se evitan colisiones.

En estrella: todos están conectados a un centro principal, típicamente un switch.

A diferencia del bus, en este caso cada cable es independiente del resto. Si un PC pone algo en una toma el resto no lo escuchan. Cada línea tienen una cola para guardar a dónde enviar los paquetes y el switch tiene un procesador que coge los paquetes de dichas colas y los envían a las salidas. Es una topología mucho más segura por el hecho de no compartir el medio.

En árbol: típica en redes empresariales. Se suele estructurar en tres niveles:

- Primer nivel: red troncal.
- Segundo nivel: red de división.
- Tercer nivel: red de acceso.

Los equipos de primer y segundo nivel suelen ser switches.

Como potencial riesgo, pueden aparecer ciclos en el árbol. Ethernet no tiene ningún mecanismo para evitar que un paquete se mueva en círculo, lo que echaría la red abajo. El protocolo Spanning Tree Protocol (STP) elimina en cualquier topología los enlaces redundantes que formen bucles.

Mallada: todos los nodos están conectados entre sí por medios independientes.

Es muy fiable, ya que si se cae un enlace tienes más caminos para llegar a tu destino. No obstante, no es escalable, ya que si metemos un $n-$ésimo nodo hay que meter $n-1$ enlaces. Para redes pequeñas es de gran utilidad.

Dentro de una empresa, la red troncal puede seguir esta topología para evitar caídas importantes.

Híbrida: se usa una mezcla de todas. Es la más utilizada.

En cuanto a las topologías que comparten el medio, para evitar el ya mencionado problema de las colisiones, se usan los dos protocolos que veremos a continuación.

Definición 1.3 (CSMA/CD)**.** El protocolo Carrier Sense Multiple Access / Collision Detection (CSMA/CD) se encarga de detectar colisiones en topologías que comparten el medio, y dar error en caso de que se produzcan. Estas se detectan comprobando si lo que hay en el bus es lo que se acaba de poner.

Es el protocolo que usa Ethernet.

Definición 1.4 (CSMA/CA)**.** El protocolo Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA) se encarga de evitar colisiones en topologías que comparten el medio. Para ello, primero escucha el medio y, si no hay ningún mensaje, envía el mensaje. Si no recibe confirmación, hay colisión.

Es el protocolo que usa Wi-Fi.

1.1.3. Clasificación de redes

Según tamaño y extensión:

- Personal Area Network (PAN): Red de área personal. Incluye todo lo que puede tener una persona: relojes, portátiles, cascos...
- Local Area Network (LAN): Red de área local. Abarca unas decenas de metros,suele ser un mismo edificio.
- Metropolitan Area Network (MAN): Red de área metropolitana. Se usa para conectar un campus o una ciudad.
- Wide Area Network (WAN).: Red de área extensa. Son redes desponibles en todo el país, como las redes telefónicas.

Según tecnología de transmisión:

- Difusión: lo que pone un nodo en el medio le llega a todos. Ejemplo de esto es un HUB.
- Punto a punto: Cada nodo solo está unido a otro. Ejemplo de esto es un switch.

Según el tipo de tranferencia de datos:

- Simple: solo transmite o recibe. Por ejemplo los TDT (para que una televisión analógica reciba señal digital).
- Half-duplex: transmite y recibe, pero no simultáneamente. Por ejemplo el Wi-Fi, aunque como cambia muy rápido no nos damos cuenta.
- Full-duplex: transmite y recibe simultáneamente. Por ejemplo, Ethernet.

1.1.4. Nomenclatura típica en figuras (Iconos)

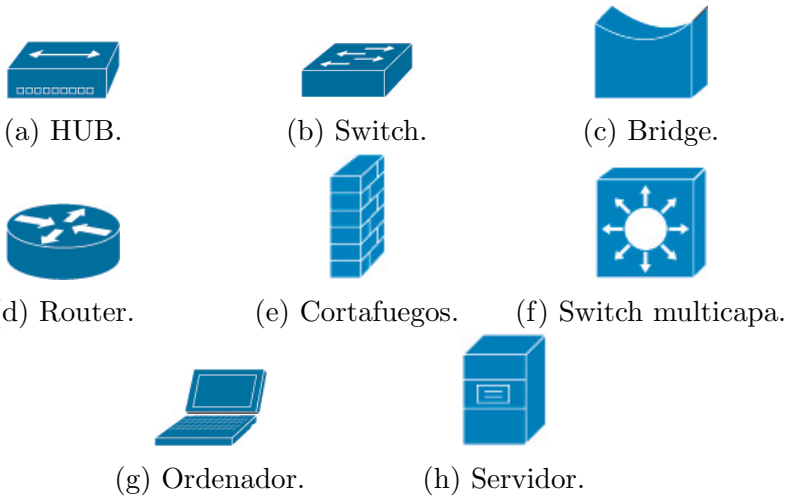
HUB: es un concentrador: permite centralizar los nodos de una red de computadoras. Se implementa mediante un bus.

Switch: tiene muchas bocas y conecta dispositivos dentro de la misma red LAN. Funciona en el nivel de enlace (nivel 2).

Bridge: funciona como un switch, pero uniendo tecnologías distintas. También funciona en el nivel de enlace.

Router: tiene pocas bocas, y se usa para conectar distintas redes.

Cortafuegos: bloquea el acceso no autorizado a una red, permitiendo tan solo el autorizado.



(a) HUB. (b) Switch. (c) Bridge.

(d) Router. (e) Cortafuegos. (f) Switch multicapa.

(g) Ordenador. (h) Servidor.

Figura 1.1: Iconos de los distintos elementos de una red.

NAT: dispotivo en el que se ejecuta el protocolo Network Address Translation (NAT), que permite que una red privada pueda acceder a Internet. Se explicará más adelante.

Switch multicapa[1]**:** todas las bocas de un switch pertenecen a la misma LAN, pero esto a veces no nos interesa. Podemos hacer distintas redes virtuales (VLAN) dividiendo un mismo switch en varias redes, permitiéndonos esto conectar dos switches distintos dentro de la misma VLAN. Esto no nos permite movernos por distintas redes en el mismo switch, ya habría que pasar por un router al necesitar movernos a nivel de red para cambiar de red.

Esta es la funcionalidad que sí se puede hacer con un switch multicapa, no necesitar pasar por un router para cambiar de red.

Todos estos elementos los veremos representados en distintos esquemas para mostrarnos las topologías de las redes. Estos vendrán identificados por los símbolos de la Figura 1.1.

1.2. Diseño y estandarización de redes

La idea principal que se sigue al diseñar redes es solucionar los problemas en capas. Se estandarizan Modelos de Referencia (no son implementaciones, solo una referencia), en los que se definen las distintas capas y las funcionalidades de cada una. Los principios que se siguen son que las funcionalidades distintas tienen que estar en distintas capas y minimizar el flujo de información entre las capas.

A continuación detallamos las capas de los modelos de referencia más importantes, junto a las funcionalidades de cada una y los problemas que han de solventar.

Capa física: se encarga de transmitir los datos. Hay distintos tipos de codificaciones para enviar bits de información.

Capa de enlace: se encarga de los mecanismos de acceso al medio. Si hay un medio común, antes de transmitir datos tiene que asegurarse de que ningún equipo está transmitiendo. Suele seguir dos protocolos:

- Media Access Control (MAC), control de acceso al medio.
- Logical Link Control (LLC), control de acceso lógico para las primeras retransmisiones. Si algún paquete llega mal, retransmite varias veces.

Capa de red: una vez llegado a este punto, se asume que no han habido colisiones en la comunicación. Esta capa se encarga principalmente de:

- El direccionamiento: saber dar una dirección y tener un identificador dentro de la red.
- El encaminamiento: saber cómo llegar al destino.

Capa de transporte: se encarga de recuperar los paquetes que en la capa de enlace no se ha podido. Es la capa encargada de la fiabilidad.

- Corrige errores.
- Gestiona la congestión de la red.
- Control de flujos: si hay un receptor más lento que el emisor, debe decirle al emisor que disminuya la velocidad de emisión, para adecuarse a la del receptor.

Además se encarga de la multiplexación de datos: mediante puertos (los veremos más adelante) le indica al SO a qué aplicación corresponde cada paquete.

Capa de aplicación: los clientes y los servidores deben buscar alguna forma de comunicarse.

Los dos modelos de referencia más importantes son el OSI y el TCP/IP, que describiremos a continuación.

1.2.1. Modelo OSI vs TCP/IP

Aplicación
Presentación
Sesión
Transporte
Red
Enlace
Física

Tabla 1.2: Modelo OSI.

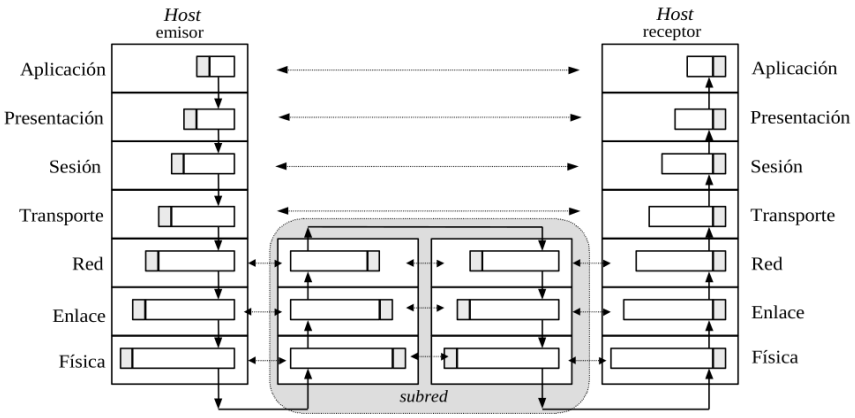


Figura 1.2: Comunicación real frente a comunicación virtual.

Aplicación
Transporte
Red
Red subyacente

Tabla 1.3: Modelo TCP/IP.

El modelo Open System Interconnection (OSI) fue propuesto por la ISO y el TCP/IP por el IETF. Las tres primeras capas del modelo OSI se corresponden con la capa de aplicación, y las dos última con la red subyacente, que es la parte física. Esta última en el modelo TCP/IP depende un poco de la tecnología está implementada de una forma u otra, pero la comunicación con la capa de red no puede variar, pues la capa de red si está estandarizada.

- Las capas físicas solo se encargan de hacer la primera conexión.
- La capa de red, salto a salto se encarga de llegar al destino, usando routers y sus tablas de encaminamiento.
- Una vez hecho el encaminamiento, las capas superiores son solo de los extremos, son los computadores de los extremos los que se comunican.
- Por tanto, los computadores tienen las 5 capas (en el caso de TCP/IP) y los routers solo las 3 más bajas.

1.3. Terminología, conceptos y servicios

En la Figura 1.2 vemos el camino que siguen los datos que le manda un emisor a un receptor. A la hora de emitir, cada capa recibe los datos de la capa superior, conocidos como Unidad de Datos de Servicio (Service Data Unit (SDU)), y le añade una cabecera, formando la Unidad de Datos de Paquete (Protocol Data Unit (PDU)). Decimos que los datos de la capa superior se han *encapsulado* en la capa inferior. Por tanto, cada capa envía el PDU a la capa inferior convirtiéndose en el SDU de la

capa inferior. La capa física, al no tener capa inferior, manda señales eléctricas en vez de bits.

Por otra parte, en la recepción, cada capa recibe de la inferior el PDU, al cual le quita la cabecera quedándose con el SDU. En función de la cabecera, se estudia qué ha de acerse en cada capa y, posteriormente, se envía el SDU a la capa superior.

En función de la capa en la que nos encontremos, el SDU recibe un nombre distinto:

- Capa de enlace: trama.
- Capa de red: datagrama.
- Capa de transporte: depende del protocolo:
  - TCP: segmento.
  - UDP: datagrama.
- Capa de aplicación: mensaje.

La información que se envía desde un host a otro ha de pasar por todas las capas para llegar al destino, lo que se conoce como *comunicación real* o vertical, y viene representada en la Figura 1.2 por las flechas continuas. No obstante, y tan solo en sentido abstracto, decimos que nos capas del mismo tipo en distintos equipos hablan entre sí, lo que se conoce como *comunicación virtual* u horizontal, y viene representada en la Figura 1.2 por las flechas discontinuas. Esta comunicación virtual, como ya hemos mencionado, no es directa, sino que se realiza a través de recursos que nos proporcionan otras capas adyacentes.

Además de las definiciones que acabamos de dar, otros términos relevantes son los siguientes:

Entidad de nivel n: entidad que se encuentra en la capa $n-$ésima.

Entidades pares: entidades de la misma capa, que se comunican horizontalmente entre sí.

Protocolo: reglas que describen cómo han de comunicarse las entidades pares. En ellos, se especifican los paquetes que se mandan, etc.

Interfaz: a diferencia del protocolo, que se refiere a cómo se comunican las entidades pares, la interfaz se refiere a cómo se comunican las entidades de capas adyacentes.

Service Access Point (SAP): punto de acceso al servicio. Es un punto específico en la interfaz entre dos capas donde se proporciona un servicio.

Servicio: conjunto de funciones que una capa proporciona a la capa superior.

Capa proveedora/usuaria del servicio: en la comunicación vertical, la capa que proporciona el servicio es la proveedora, y la que lo usa es la usuaria.

Pila de protocolos: conjunto de protocolos que se usan en cada capa.

Arquitectura de red: modelo de referencia, junto a la pila de protocolos.

Otro concepto importante es el de *retardo*, que describireos a continuación.

1.3.1. Retardos en la comunicación

Supongamos que queremos transmitir un paquete entre dos equipos (terminales), por medio de otro, un router. Veamos los retardos que tenemos en la comunicación.

1. En primer lugar, hemos de considerar el **tiempo de transmisión**, que es el tiempo que se tarda en poner el paquete en el medio.

   Si el tamaño del paquete es de L Bytes y la velocidad de transmisión es de v_t bps, el tiempo de transmisión t_t es de:

   $$t_t = \frac{L \cdot 8}{v_t} \text{ s}$$

   La velocidad de transmisión depende exclusivamente de la tarjeta de red.

2. En segundo lugar, tenemos el **tiempo de propagación**, que es el tiempo desde que se escribe el primer bit en el medio hasta que llega este primer bit al siguiente equipo.

   Este tiempo depende de la distancia entre los equipos y de la velocidad de transmisión, que depende del medio. En el caso de una transmisión inalámbrica, la velocidad de transmisión es la velocidad de la luz, mientras que en una transmisión por cable suele ser de $^2/_3$ de la velocidad de la luz.

   Si la distancia entre los equipos es de d m y la velocidad de transmisión es de v_p $^{\text{m}}/_{\text{s}}$, el tiempo de propagación t_p es de:

   $$t_p = \frac{d}{v_p} \text{ s}$$

3. En tercer lugar, tenemos el tiempo que se encuentra en el equipo intermedio, en nuestro caso el router. Cuando el paquete llega al equipo intermedio, éste lo mete en una cola hasta que pueda procesarlo. El tiempo que el paquete está en la cola depende se conoce como **tiempo de cola**, y depende de la situación del equipo. Además, el equipo ha de procesar el paquete, lo que le lleva un tiempo conocido como **tiempo de procesamiento**, que suele ser del orden de milisegundos. Por último, y para poder continuar con la comunicación, el equipo ha de obtener acceso al medio, lo que le lleva un tiempo conocido como **tiempo de acceso al medio**.

   Por tanto, el tiempo que el paquete está en el equipo intermedio es la suma de estos tres tiempos.

4. Por último, la comunicación deberá continuar, por lo que se obtienen nuevos retardos de transmisión, propagación, etc. No obstante, en estos casos, estos no serán los mismos, pues la distancia entre equipos, tarjetas de red, etc., serán distintos.

1.3.2. Tipos de servicios

Relacionado con el nivel de transporte, hay dos clasificaciones importantes:

Según la conexión: En función de si, antes de enviar un paquete, se comprueba que el otro equipo esté encendido o no, tenemos dos tipos de servicios:

- Service Oriented to Connection (SOC): sí se comprueba.
- Service No Oriented to Connection (SNOC): no se comprueba.

Según la fiabilidad: En función de si se asegura que todo funcione bien o no, (por ejemplo, que todos los bits de un archivo estén bien), tenemos dos tipos de servicios:

- Fiable: se asegura de que todo funcione bien. Si algo falla, la conexión se termina.
- No fiable: no comprueba que todo funcione bien. La finalidad de estos protocolos es ser rápidos.

Para tener un protocolo fiable, contamos con los siguientes mecanismos:

- Control de conexión. Ser fiable implica ser orientado a conexión.
- Control de errores.
- Control de congestión. Hablamos de congestión cuando las colas de los routers se llenan y empiezan a descartar paquetes.
- Control de flujo.
- Entrega ordenada. Si se envían muchos paquetes, estos deben llegar en orden.

Algunos protocolos que estudiaremos más adelante:

- TCP es un servicio fiable, y por tanto orientado a conexión.
- UDP es un servicio no orientado a conexión, y por tanto no fiable.

1.4. Internet: topología y direccionamiento

Internet tiene dos aspectos importantes:

- Los protocolos de comunicación.
- Cómo se organiza Internet, el direccionamiento.

Todo esto se describe en las conocidas Request for Comments (RFC).

1.4.1. Organización topológica

Los operadores se establecen según la siguiente jerarquía:

- **Tier 3**: son los más cercanos a los usuarios. Ofrecen servicios de conectividad a empresas y particulares, y se conocen como Internet Service Provider (ISP). Algunos conocidos en España son Movistar, Vodafone, Orange...
- **Tier 2**: son de ámbito más regional. Necesitan pasar por una Tier 1 para llegar a toda Internet y ofrecen servicios de conectividad a operadores de Tier 3.
- **Tier 1**: son los que componen la estructura troncal de Internet. Están todos comunicados entre sí y están como mínimo en dos continentes.

Hay dos tipos de relaciones entre operadores:

- **Tránsito:** conexiones entre distintos tier. Por ejemplo un tier 3 paga a un tier superior para enviar datos.
- **Peering:** conexiones entre el mismo tier.

Antiguamente, para que un ISP de un país hablase con otro del mismo país había que ir hasta EEUU por falta de recursos. Más tarde se pusieron puntos neutros en cada país para comunicar operadores dentro de un mismo país.

1.4.2. Red Iris

La Red Iris la red española para investigación. Todas las universidades públicas y centros de investigación están conectados a ella.

La red se divide según autonomía (en Andalucía, se denomina Red RICA) y por otro lado tiene conexiones externas con la red científica europea.

1.4.3. Direccionamiento por capas

Según la capa en la que nos encontremos, hemos de realizar el direccionamiento de una forma u otra.

- Capa de Enlace: La dirección depende de la tarjeta de red, y necesitamos saber la dirección MAC del siguiente punto. Las direcciones MAC son de la forma AA:BB:CC:DD:EE:FF, y son teóricamente únicas en todo el mundo. No obstante, en la realidad se ponen aleatorias para evitar seguimientos (puesto que si sabemos la dirección MAC de una tarjeta podemos hacer seguimiento de paquetes).
- Red: Se usan direcciones IP de la forma A.B.C.D. Las públicas son únicas en todo el mundo, las privadas no.
- Transporte: direccionamiento a través de puertos, que identifican a qué proceso va un determinado paquete.
- Aplicación: Nombres de dominio mediante DNS.

2. Capa de red

Objetivos

- Comprender las funcionalidades y servicios de la capa de red.
    - Concepto de conmutación de paquetes y datagramas.
    - Direccionamiento en Internet.
    - Encaminamiento salto a salto.
    - Asociación con la capa de enlace a través del protocolo ARP.
    - Señalización de errores mediante el protocolo ICMP.

Introducción

En este tema estudiaremos a fondo la capa de red. Recordemos que seguimos el Modelo TCP/IP descrito en la Tabla 1.1.

2.1. Funcionalidades

Funcionalidades y servicios TCP/IP:

- **Direccionamiento:** identificación de equipos dentro de la red.
- **Encaminamiento:** llegar salto a salto desde el origen al destino. Especifica el camino que deben seguir los paquetes.
- **Fragmentación:** las tarjetas suelen tener un tamaño máximo de paquete, y si queremos enviar un paquete más grande tenemos que fragmentarlo y, en el destino, ensamblarlo.
- **Conmutación.**
- **Interconexión de redes.**
- En OSI: **control de gestión.**

El protocolo que desarrollaremos en este tema es IP por ser el que en la actualidad se ha impuesto, aunque existen otros como ATM, x25...

2.2. Conmutación

Definición 2.1 (Conmutación)**.** Acción de establecer o determinar caminos de extremo a extremo que permitan transmitir información.

Uno de los primeros ejemplos claros de conmutación que se vio en la tecnología de las comunicaciones fue la conmutación de circuitos para la telefonía, que desarrollamos a continuación.

2.2.1. Conmutación de circuitos

Antiguamente existía una conmutación física de circuitos, muy usada en telefonía. De esta forma, hay muchos cables entre los usuarios y las centrales (uno por cada usuario), y menos cables entre cada par de centrales, ya que no todos los usuarios hablan al mismo tiempo.

La comunicación por conmutación de circuitos implica tres fases:

1. El establecimiento del circuito. Cada central une los cables que correspondan y se genera el camino.

2. La trasferencia de datos a través del circuito dedicado.

3. La desconexión del circuito, se libera el circuito para su reutilización.

Beneficios

- Recursos dedicados (tenemos un cable solo para nosotros), lo que facilita las comunicaciones a tiempo real y sin retardos.

- El recurso se mantiene dedicado toda la sesión.

- No hay competición por conseguir el medio.

- El circuito es fijo, no hay decisiones de encaminamiento una vez establecido.

- Simplicidad en la gestión de los nodos intermedios.

Desventajas

- Cuando un usuario no usa su cable no lo usa nadie más. Uso ineficiente de recursos.

- Hay establecimiento de llamada (para que todos los cables se toquen).

- Es poco tolerable a fallos, si algo no funciona, todo deja de funcionar.

2.2.2. Conmutación de paquetes

En la actualidad, no se envía una señal analógica; sino que, como sabemos, se envía el SDU junto con la cabecera. En la capa de red, vimos que el SDU se denomina *datagrama*, y veremos que este se ha de fragmentar en distontos bloques, a los que denominaremos *paquetes*. Por tanto, los paquetes son cada uno de los bloques de un datagrama, y es lo que se envía como tal por la red.

Observación. En general, cuando se quiera hacer referencia a un conjunto de datos que se envía por la red, sin especificar en qué capa nos encontramos, o sin ser más precisos, también usaremos el término de *paquete*.

A la hora de realizar la conmutación, hay dos formas de hacerlo, la conmutación mediante datagramas y la conmutación mediante circuitos virtuales.

Conmutación de datagramas

Las características de la conmutación de datagramas son:

- No hay establecimiento de conexión: enviamos un paquete y no sabemos si el otro extremo está encendido.
- El envío de los distintos paquetes se hace independientemente. El encaminamiento se hace paquete a paquete, por lo que se pueden seguir caminos distintos. Por este motivo, los paquetes pueden llegar desordenados, algo que controlarán otras capas.

  Además, si se produce fragmentación, no se ensamblarán los paquetes hasta que lleguen al destino, ya que distintos paquetes de un mismo datagrama pueden seguir distintos caminos.
- En cada nodo intermedio los paquetes que llegan se almacenan en una cola, y cuando sea posible se envían al próximo nodo.
- Como el encaminamiento se hace salto a salto, todos los paquetes han de tener la dirección de origen (para las respuestas o encaminamientos específicos, aunque esto no lo veremos en la asignatura) y de destino. A veces, para hacer difusiones de datos, nos puede interesar taner varias direcciones de destino.

Como el medio es común, los nodo de interconexión necesitan colas para poder gestionar los paquetes que le llegan. A la hora de esta conmutación, se hace el mejor esfuerzo, pero si algo falla la capa de red no se encarga de gestionar el fallo.

Un protocolo que lleve a cabo esta conmutación es IP, que desarrollaremos más adelante. Este es el tipo de conmutación que se usa mayoritariamente en la actualidad, y es en el que nos centraremos en la asignatura.

2.2.3. Conmutación con circuitos virtuales

En este caso, la conmutación difiere ligeramente de la conmutación por datagramas vista, siendo una mezcla entre la conmutación de circuitos y la de paquetes.

En este caso, para enviar un paquete de un origen a un destino, aunque haya distintos caminos posibles, se establece el camino desde el principio denominado

circuito (siguiendo la idea de la conmutación de circuitos). Este circuito no obstante es virtual, ya que los recursos no son dedicados completamente, sino que se reservan temporalmente pero se pueden reutilizar.

Cada router decide el camino que seguirá cada paquete, y los paquetes del mismo datagrama seguirán el mismo camino. Por tanto, el primer paquete en llegar al router reservará los recursos para los próximos paquetes.

- Hay que establecer conexión para averiguar la ruta a seguir.
- Si un router se cae, se cambia el camino.

Un protocolo que lleva a cabo esta conmutación es Asyncronous Transfer Mode (ATM), que estaba presente en el inicio de la telefonía digital.

2.3. El protocolo IP

El Internet Protocol (IP) es un protocolo para la interconexión de redes. Existen dos versiones:

- IPv4: Es la que se diseñó inicialmente, aunque tiene una limitación en la cantidad de direcciones.
- IPv6: Pasó de 32 a 128 bits, lo que supone una cantidad en la práctica ilimitada de direcciones.

En la actualidad la limitación de direcciones se empieza a notar, por lo que hay una transición gradual hacia IPv6, aunque sigue predominando IPv4. Desorrallaremos IPv4 en este tema, aunque mencionaremos algunas diferencias con IPv6.

Características de IPv4

- Resuelve el direccionamiento en Internet en la capa de red, ya que cada tarjeta de red tiene una dirección IP.
- Realiza el encaminamiento (o retransmisión) salto a salto entre equipos y routers.
- Ofrece un servicio no orientado a conexión y no fiable, ya que:
  - No hay establecimiento de conexión lógica entre las entidades.
  - No hay control de errores ni de flujos. Los errores que se produzcan tienen que arreglarlos una capa superior si se precisa.
- Gestiona la fragmentación para adaptarse al MTU de cada tarjeta de red, como veremos. A la unidad de datos completa se le llama datagrama y a los fragmentos paquetes.
- Es un protocolo de máximo esfuerzo, los datagramas se pueden perder, duplicar, retrasar, llegar desordenados...

2.3.1. Direccionamiento

Para identificar cada equipo en la red, se usan direcciones IP. El lector posiblemente esté más familiarizado con las direcciones red, como `www.google.com`, pero estas en realidad son nombres de dominio que se traducen a direcciones IP, como veremos en el Capítulo dedicado a la capa de aplicación. Mientras tanto, hemos de saber que todo equipo en la red tiene una dirección IP asociada. Además, esta (a priori) es única y no se puede repetir, lo que supone una limitación. Como más adelante veremos, para solventar este problema se usan también direcciones privadas, algo que no contemplaremos por el momento.

Una dirección IP consta de 32 bits y la nomenclatura usada es: `A.B.C.D` donde cada letra es un número decimal en el rango `0-255` (ya que codificará 8 bits). El rango por tanto que tenemos es `0.0.0.0-255.255.255.255.` Una dirección tiene dos partes bien diferenciadas, la que identifica la red y la que identifica el equipo en cuestión (en realidad, identifica la tarjeta de red).

Para saber qué parte de la dirección IP identifica el equipo y cuál la red, se emplea la máscara de red.

Definición 2.2 (Máscara de red)**.** Es un conjunto de 32 bits (al igual que una dirección IP) que se usa para identificar qué parte de la dirección IP identifica la red y cuál el equipo.Contiene los primeros n bits consecutivos a 1, y el resto a 0.

¿Cómo se usa la máscara?
Para saber cuál es la dirección de la red, se hace un `AND` lógico entre la dirección IP y la máscara (por lo que nos quedaremos con los primeros n bits de la dirección IP). El resto de bits identificará al equipo dentro de dicha red.

Notemos por tanto que, dentro de las posibles direcciones IP de una misma red, la dirección con todos los bits de equipo a 0 está *reservada* para la dirección de la red, y no podrá asignarse a ningún equipo.

Notación. Es común querer dar una dirección IP junto a su máscara de red. Para esto, se podrá usar la notación `A.B.C.D/n`, donde `A.B.C.D` es la dirección IP en sí y `n` es el número de bits a 1 de la máscara de red. Como ya hemos mencionado que estos bits han de estar al inicio y consecutivos, sabiendo el valor de `n` sabremos cuál es la máscara de red.

Ejemplo. Supongamos que tenemos una dirección IP `192.168.1.27/24`, y queremos saber cuál es la dirección de la red. Pasando a binario y haciendo un `AND`, tenemos:

	1100 0000 . 1010 1000 . 0000 0001 . 0001 1011	(dirección IP)
AND	1111 1111 . 1111 1111 . 1111 1111 . 0000 0000	(máscara de red)
	1100 0000 . 1010 1000 . 0000 0001 . 0000 0000	(dirección de la red)

Por tanto, pasando de nuevo a decimal, la dirección de la red es `192.168.1.0`.

Direccionamiento jerárquico

Internet usa direccionamiento jerárquico basado en clases. Cada clase contiene las direcciones IP de un rango determinado:

- Clase A → `0xx...x/8` ⟹ `0.0.0.0 - 127.255.255.255`. Tenemos $2^7 = 128$ redes con $2^{24} \approx 16 \cdot 10^6$ equipos en cada una.
- Clase B → `10xx...x/16` ⟹ `128.0.0.0 - 191.255.255.255`. Tenemos $2^{14} = 16384$ redes con $2^{16} = 65536$ equipos en cada una.
- Clase C → `110xx...x/24` ⟹ `192.0.0.0 - 223.255.255.255`. Tenemos $2^{21} \approx 2 \cdot 10^6$ de redes con $2^8 = 256$ equipos en cada una.
- Clase D → `1110xx...x` ⟹ `224.0.0.0 - 239.255.255.255`. No se usa para identificar equipos ni redes sino para multidifusión (*multicast*). Cada dirección identifica a todo un grupo de equipos. Para gestionar esto existe el protocolo IGMP para suscribirse a grupos.
- Clase E → `1111xx...x` ⟹ `240.0.0.0 - 255.255.255.255`. Es el rango experimental; es decir, las direcciones que se dejan para hacer pruebas.

Direcciones reservadas

Además de las restricciones de cada clase, hay determinadas direcciones que están reservadas y no se pueden asignar a ningún equipo. Algunas de estas direcciones son:

- Dirección de red: Cualquier dirección IP con todos los bits de equipo a 0. Está dedicada para identificar la red en sí.
- Dirección de difusión (*broadcast*): Cualquier dirección IP con todos los bits de equipo a 1. Se usa para enviar un paquete a todos los equipos de la red.

  Cuando se tiene que encontrar un equipo y no se sabe cuál, se manda por la dirección de difusión y lo escuchará quien tenga que escucharlo.
- `127.a.b.c`: Denominada dirección de *loopback*, *localhost* o *localloop*. Se usa para hacer pruebas, y es una conexión que hacemos a nuestra propia máquina. Originalmente (y la más común) era `127.0.0.1`, pero en la actualidad se ha aumentado el rango. Estas redes no requieren de una tarjeta de red específica.

Llegados a este punto, podemos dar una definición más correcta de router, que ya habíamos mencionado anteriormente.

Definición 2.3 (Router)**.** Es un dispositivo de la capa de red cuya funcionalidad principal es conectar distintas redes y encaminar los paquetes a través de ellas.

Cuenta con varias tarjetas de red (también llamadas interfaces), una por cada red a la que se conecta, y cada una cuenta con una dirección IP asociada en cada red.

Como curiosidad, es posible crear routers en un ordenador con varias tarjetas de red con `Linux`. Con el comando `sysctl -a` podemos consultar el valor de la variable `net.ipv4.ip_forward`, que nos informa sobre si redirigimos paquetes o no. Si está con el valor 1, dicho equipo es un router.

Observación. Como un switch funciona a nivel de enlace, todo lo conectado a dicho switch está en la misma red. Por tanto, tampoco tiene dirección IP asignada.

Direccionamiento sin clases

Si usamos solo el direccionamiento con clases estaríamos desperdiciando muchísimas direcciones IP. Por ejemplo, si tenemos 1000 equipos ($2^8 < 1000 < 2^16$) tendríamos que usar una red de clase B, con la que desperdiciaríamos más de 60.000 direcciones. La solución a este problema es usar el direccionamiento sin clase, que nos permite usar la máscara de red deseada.

- **Subredes**

  Si, por ejemplo, queremos una red de menos de 256 equipos, podemos aumentar el número de bits de la máscara a 1, para conseguier más bits dedicados a identificar la red y menos para identicar equipos. Cada vez que añadimos un bit a la máscara, estamos dividiendo una red en dos mitades.

  Ejemplo. Supongamos que queremos identificar 100 equipos dentro de una misma red. Contando además con la dirección de red y la de difusión, necesitamos 102 direcciones. Como $2^6 < 102 < 2^7$, necesitamos 7 bits para identificar a los equipos. Por tanto, la máscara a usar será `/25`.

- **Superredes**

  Si hacemos el procedimiento inverso, quitarle un bit a la máscara, duplicamos la cantidad de equipos que podemos direccionar. Por ejemplo, en `/23` estamos juntando dos redes de clase C.

  Ejemplo. Supongamos que queremos una red de 1000 equipos. Contando con la dirección de red y la de difusión, necesitamos 1002 direcciones. Como $2^9 < 1002 < 2^{10}$, necesitamos 10 bits para identificar a los equipos. Por tanto, la máscara a usar será `/22`.

Como vemos el funcionamiento es igual que en el direccionamiento con clase, pero reduce significativamente (aunque no elimina) el desperdicio de direcciones. A nivel práctico red, subred y superred no se diferencian, y nos referimos a todas ellas como redes.

Direcciones privadas

Como hemos venido mencionando en distintas ocasiones, la escasez de direcciones es un gran problema presente en IPv4, ya que tan solo hay 2^{32} direcciones posibles, las cuales ya se agotaron en Noviembre de 2019. Aunque se vayan recopilando direcciones de sitios obsoletos, empresas desaparecidas, etc. el problema sigue existiendo.

Hay varias soluciones posibles para solventarlo.

- Direccionamiento sin clase: es una solucion que reduce el desperdicio de direcciones, peero aun así tiene la limitación de 2^{32} direcciones.

- IPv6, el cual usa 128 bits para las direcciones. La notación utilizada es

  `FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF`, en el que cada dígito es un número hexadecimal.

  En total hay 2^{128} direcciones posibles (más de 10^{37}), lo que en la práctica las hace ilimitadas. Aunque sea compatible con IPv4, la transición está siendo lenta.

- Direcciones privadas: esta es la principal solución que se usa en la actualidad, ya que hace el número de direcciones prácticamente ilimitado. Desarrollaremos este concepto a continuación.

Direcciones públicas: Cada dirección se asigna a un único dispositivo en todo Internet. Se asignan centralizadamente[1], y como son limitadas, hay que pagar por cada una.

Direcciones privadas: Solo se pueden usar en redes privadas o *intranets*, sin acceso directo al resto de Internet. Por tanto, al no ser accesibles desde fuera, se pueden repetir en distintas redes privadas, lo que aumenta el número de direcciones disponibles.

Para poder comunicarse con el resto de Internet (ya que si no tendrían poca utilidad), será necesario una dirección pública por la cual se haga la comunicación. Para esto se usa el NAT, que veremos más adelante.

Respecto al direccionamiento jerárquico con clases, dentro de cada clase se definen algunos rangos de direcciones a usar como IP privadas. Estos rangos son:

- Clase A → `10.x.y.z/8`
- Clase B → `172.16-32.y.z/16`
- Clase C → `192.168.y.z/24`

2.3.2. Network Address Translation (NAT)

Como hemos mencionado anteriormente, para que una red privada pueda comunicarse con el resto de Internet, es necesario que haya una dirección pública que haga de intermediario. Para esto se usa la técnica de NAT, que posibilita la traducción de direcciones. Al encontramos en la capa de red, el PDU contiene la cabecera IP que contiene, entre otros datos:

- Dirección IP origen (IPsc) junto con el puerto origen (sport).
- Dirección IP destino (IPdest) junto con el puerto destino (dport).

El concepto de puerto lo veremos más adelante y desarrollaremos a fondo en la Capa de Transporte. Por el momento, tan solo es necesario saber que es un número que se asigna a cada proceso que se comunica en la red, y que se usa para saber a

[1] Inicialmente por IANA, actualmente por ICANN.

qué proceso enviar la respuesta.

Para posibilitar la traducción, se usa una tabla de direcciones a modo de "diccionario", tal y como introducimos a continuación.

Definición 2.4 (Tabla de traducciones)**.** La Tabla de Traducciones es una tabla que se guarda en la memoria de todo router que haga NAT. Por cada traducción que deba hacerse, se guarda una entrada en la tabla que relaciona la dirección IP y puerto originales con la dirección IP y puerto traducidos.

La tabla se va actualizando con cada nueva traducción, y cuenta con un temporizador (normalmente de 5 minutos) que borra las entradas que lleven un tiempo sin usarse. De esta forma, se evita que la tabla se sature y se libera memoria.

En el caso de que llegue una petición que ya esté en la tabla, se reutiliza la información de la tabla, sin crearse una nueva entrada.

Definición 2.5 (*Masquerading*)**.** Proceso de enmascaramiento que hace el router al traducir la dirección privada del equipo en su dirección pública.

Se "enmascara" la dirección privada, de forma que el servidor no sabe a qué equipo de la red privada está respondiendo.

Observación. El uso de NAT plantea un problema de seguridad. Un atacante, conociendo la IP pública del router, puede hacer un barrido de puertos y puede conseguir que algún paquete entre. En tal caso, el router le responderá, y el atacante sabrá que hay un equipo detrás de esa IP pública y puerto, por lo que podrá intentar atacar a ese equipo.

Para evitar esto, para cada traducción puede guardarse tanto las IP y puerto de origen y destino sin traducir, como las traducidas. De esta forma, si llega una petición que coincide con la IP y puerto origen, pero no con la IP y puerto destino, se descarta directamente. Esta técnica se denomina *NAT estricto*.

Hay dos tipos de NAT, en función de dónde y cuándo se haga la traducción.

Source NAT (SNAT): el origen de los datos está en una red privada. Por tanto, al enviarse se cambia la dirección IP de origen, y la traducción a la correcta (en la respuesta) se hará tras el encaminamiento (*postrouting*).

Destination NAT (DNAT): el origen de los datos está en la red pública. Por tanto, al recibir los datos se cambia la dirección IP de destino, y la traducción a la correcta (en la respuesta) se hará antes del encaminamiento (*prerouting*).

En este caso la tabla de traducciones del router que realiza DNAT ha de ser estática (la inserción debe ser a mano), ya que en otro caso el router no sabrá a donde redirigir las peticiones entrantes. Este proceso se denomina *port forwarding*.

Planteemos un primer ejemplo de SNAT, que nos ayudará a comprender cómo funciona esta técnica.

Ejemplo. Supongamos la situación de la Figura 2.1, en la que un portátil dentro de una red privada quiere acceder a un servidor HTTP en Internet.

El equipo envía una petición al router (1), que este reenvía al servidor (2). El servidor responde al router (3), que a su vez reenvía la respuesta al equipo (4). Se

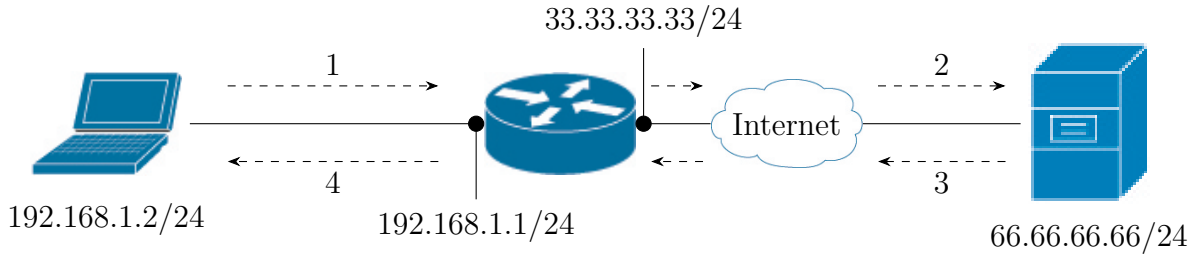


Figura 2.1: Ejemplo de red con SNAT.

trata de SNAT, ya que la petición parte de una red privada. Veamos qué ocurre en cada uno de los pasos:

(1) El ordenador envía una petición HTTP al router.

El puerto de origen, el cual asignará alteatoriamente el SO (ya se verá), pongamos que es el 1075. El puerto de destino, en el caso de HTTP, es el 80. Por tanto, la cabecera IP del paquete que envía el portátil al router contendrá:

- IPsc:sport: `192.168.1.2:1075`.
- IPdest:dport: `66.66.66.66:80`.

(2) El router ha de realizar la traducción de direcciones, ya que la IP del portátil es privada y no puede ser usada en Internet. Para esto, modifica la cabecera IP poniendo como IP origen su propia IP pública, y como puerto origen un puerto que aún no haya sido usado (por ejemplo, 12345). La cabecera IP así:

- IPsc:sport: `33.33.33.33:12345`.
- IPdest:dport: `66.66.66.66:80`.

La tabla de traducciones del router quedaria (donde notamos con "′" la traducida):

IPsc	sport	IPsc′	sport′
`192.168.1.2`	`1075`	`33.33.33.33`	`12345`

Tabla 2.1: Tabla de traducciones del router con SNAT.

Tras esta traducción, el router envía el paquete al servidor.

(3) Tras el procesamiento del paquete en el servidor, este envía la respuesta al router. La cabecera IP del paquete de respuesta contendrá:

- IPsc:sport: `66.66.66.66:80`.
- IPdest:dport: `33.33.33.33:12345`.

(4) El paquete llega sin problema al router, ya que la IP de destino es pública. El router debe realizar de nuevo la traducción para saber a qué equipo de la red privada debe enviar la respuesta. Para ello, consulta la tabla de traducciones (Tabla 2.1) y, tras modificar de nuevo la cabecera IP (*postrouting*), esta queda:

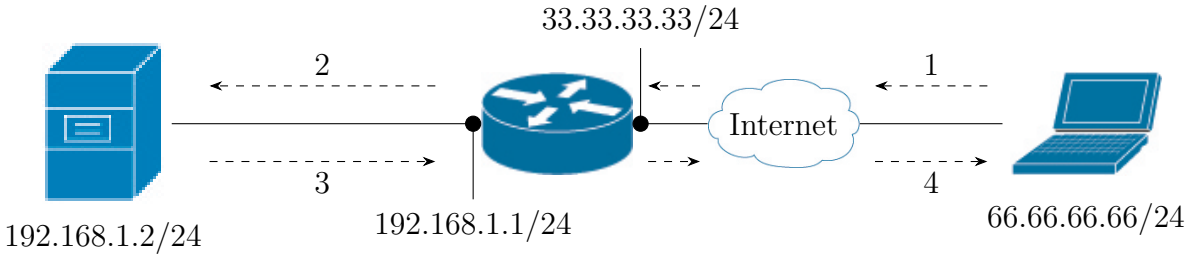


Figura 2.2: Ejemplo de red con DNAT.

- IPsc:sport: `66.66.66.66:80`.
- IPdest:dport: `192.168.1.2:1075`.

Planteamos ahora el siguiente ejemplo de DNAT, que nos ayudará ahora a comprender cómo funciona esta técnica.

Ejemplo. Supongamos la situación de la Figura 2.2, en la que un portátil dentro de una red privada quiere acceder a un servidor HTTP en Internet.

El equipo envía una petición al router (1), que este reenvía al servidor (2). El servidor responde al router (3), que a su vez reenvía la respuesta al equipo (4). Se trata de DNAT, ya que la petición parte de la red pública. Veamos qué ocurre en cada uno de los pasos:

(1) El ordenador envía una petición HTTP al router.

El puerto de origen pongamos que es el 1050. El puerto de destino, tras la traducción efectivamente ha de ser el 80 (ya que es un servidor HTTP). No obstante, antes de la traducción este puerto ha de ser el correspondiente al puerto que hayamos asignado al servidor HTTP al que queremos acceder. Por ejemplo, sea la tabla de traducciones del router la de la Tabla 2.2 (que hemos de haber configurado previamente en el *port forwarding*).

IPdest	dport	IPdest′	dport′
`33.33.33.33`	23456	`192.168.1.2`	80

Tabla 2.2: Tabla de traducciones del router con DNAT.

En tal caso, el puerto de destino será el 23456. Por tanto, la cabecera IP del paquete que envía el portátil al router contendrá:

- IPsc:sport: `66.66.66.66:1050`.
- IPdest:dport: `33.33.33.33:23456`.

Notemos que la IP de destino no es el servidor como tal, sino el router (ya que es es al que tiene acceso el portátil), y el puerto de destino es el que hemos asignado en la tabla de traducciones para el servidor HTTP al que queremos acceder.

(2) Tras llegar al router, este ha de realizar la traducción de direcciones (*prerouting*), ya que la IP de destino era el router mismo. Consultando la tabla de traducciones (Tabla 2.2), la cabecera IP del paquete que envía el router al servidor quedará:

- IPsc:sport: `66.66.66.66:1050`.
- IPdest:dport: `192.168.1.2:80`.

Tras esta traducción, el router envía el paquete al servidor (ya en la red privada).

(3) Tras el procesamiento del paquete en el servidor, este envía la respuesta al router. La cabecera IP del paquete de respuesta contendrá:

- IPsc:sport: `192.168.1.2:80`.
- IPdest:dport: `66.66.66.66:1050`.

(4) El paquete llega sin problema al router, ya que la IP de destino es pública. El router debe realizar de nuevo la traducción para saber ahora de qué equipo de la red privada proviene la petición. Para ello, consulta de nuevo la tabla de traducciones (Tabla 2.1) y, tras modificar de nuevo la cabecera IP, esta queda:

- IPdest:dport: `33.33.33.33:23456`.
- IPdest:dport: `66.66.66.66:1050`.

Notemos que, en el SNAT, la tabla de traducciones se va actualizando con cada nueva traducción, mientras que en el DNAT la tabla de traducciones ha de ser estática, ya que en otro caso el router no sabrá a donde redirigir las peticiones entrantes.

2.3.3. Encaminamiento

Se dice del proceso de encontrar el mejor camino para llevar la información (paquetes) de un origen a un destino dado.

Se realiza salto a salto y datagrama a datagrama en función de la IP destino del paquete y de las tablas de encaminamiento que hay en los routers.

Todo equipo en red tiene tablas de encaminamiento, tanto los hosts como los routers.

Tablas de encaminamiento

Veamos los campos que tienen estas tablas, primero los campos importantes y luego los menos relevantes (entre paréntesis).

- Red destino.
- Máscara.

- Siguiente salto.
- (Interfaz de salida del equipo), puede ser redundante.
- (Protocolo).
- (Flags).
- (Coste) esperado para llegar a dicho equipo, por ejemplo el número de saltos.

Tenemos varios tipos de rutas:

Rutas directas: (marcado con * en siguiente salto). Para llegar a alguna de las redes en las que estamos. No es necesario realizar ningún salto, ponemos el paquete en la red y será recibido por el destinatario. Normalmente, un router suele estar en varias redes (dado que hace de puente entre distintas redes), y un host suele estar en una única red.

Rutas indirectas: Para llegar a redes a las que no estamos directamente conectados a través de un intermediario. Es necesario pasar por mínimo un salto para llegar al destino.

Entrada por defecto (default): (notado por `0.0.0.0` o *default* en red destino y /0 en máscara). Hace referencia a cualquier red que no haya sido aceptada por el resto de entradas. Al equipo que sale en *siguiente salto* (el que nos conecta con el exterior) lo llamamos pasarela (*gateway* en inglés). No siempre es necesario. Poner simplemente un `\0` en máscara es equivalente a una entrada por defecto.

Tenemos dos tipos de encaminamientos:

Estático: se hace a mano.

Dinámico: se hace por protocolo. Puede ir cambiando, por ejemplo, si se rompe un router se busca otro camino para llegar al destino.

Uso de la tabla de encaminamiento

Dada una dirección IP, para buscar su entrada en la tabla de encaminamiento, se hace la operación lógica `AND` con la máscara de cada entrada. Si el resultado coincide con la red de destino entonces la entrada es válida para dicha IP.

- Si no hay ninguna entrada válida, se envía un mensaje de error ICMP, pero no se intenta arreglar dicho error.
- Si hay varias entradas válidas para una IP, se escoge la entrada más restrictiva.

Observación. localhost tiene que estar en la tabla de encaminamiento. Su interfaz es *lo*.

A veces es útil minimizar las tablas de encaminamiento, agrupando las redes con las que se trabaja. A menudo tenemos que compartir las tablas de encaminamiento (como veremos en algunos de los siguientes protocolos) y para ello lo mejor es que sean lo más compactas posible. Lo ideal es tener una entrada por cada interfaz del dispositivo.

Protocolos de intercambio de información de encaminamiento

Para facilitar la administración y aumentar la escalabilidad, Internet se jerarquiza en Autonomous System (AS), redes muy grandes gestionadas por una autoridad. Los AS suelen ser del orden de un país. Se definen dos niveles de encaminamiento (intercambio de tablas):

- Algoritmos Interior Gateway Protocol (IGP): RIP, OSPF... Es dentro de cada AS.
- Algoritmos Exterior Gateway Protocol (EGP): BGP. Es entre distintos AS.

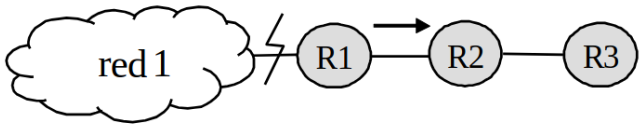
2.3.4. Routing Information Protocol (RIP)

Aunque es una funcionalidad de la capa de red, se implementa sobre la capa de aplicación, opera sobre UDP en el puerto 520 (son cosas independientes la funcionalidad y la implementación).

Es un protocolo que adopta un algoritmo vector-distancia, es decir, se basa exclusivamente en el número de saltos, ignorando la velocidad. Una vez que aprende un camino, no aprende otro a no ser que mejore el anterior.

Cuando un router RIP se enciende y es configurado, recibe de todos sus vecinos (dirección multicast 224.0.0.9) por defecto cada 30 segundos, los vectores-distancia para todos los posibles destinos, es decir, información sobre las rutas a las que saben llegar. De entre ellos, para un destino dado, se elige el camino de menor coste y se guarda la información sumando 1 al coste anunciado por dicho vecino.

La dirección multicast que mencionábamos antes la escuchan todos los routers que soportan RIP.



Un problema que puede ocurrir es que algún camino se rompa y, dado la naturaleza del protocolo, esto tarda en notificarse. En el ejemplo que vemos arriba, R1 es notificado de que ya no es posible llegar a la red 1 por el camino que tenía aprendido. Pero R2, que aún no ha sido notificado, le comunica que él sabe llegar, por tanto R1 lo aprende. Cuando R2 se entera pasa un poco lo mismo, que aprende el camino por R1. Así pasa hasta el infinito (que en el caso de RIP es equivalente a coste 16). Por esto a esto se le llama problema de la "cuenta al infinito". Veamos algunas posibles soluciones:

Split horizon: Se basa en que a un router se le prohíbe compartir una ruta por la misma interfaz por la que la aprendió en primer lugar.

Hold down: Retrasa los mensajes que nos llegan de una dirección que ya conocemos 180 segundos, esperando a que nos respondan los anteriores, si siguen activos.

Poison reverse: Si no sabemos llegar a un destino, decimos que el coste es infinito (coste = 16).

2.3.5. Open Shortest Path First (OSPF)

Este protocolo permite definir el coste. Tiene un criterio por defecto, en el que el coste de un enlace es el inverso del ancho de banda (la velocidad) de dicho enlace. Busca el camino global que minimiza la suma de todos los costes usando Dijkstra.

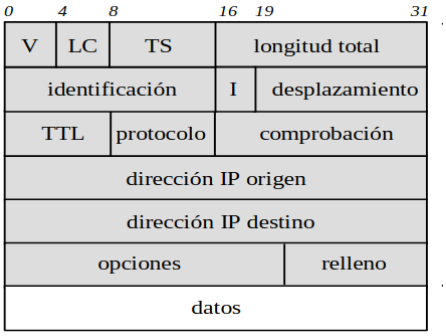
Permite definir áreas, de forma que la difusión se hace en unas áreas concretas. Esto hace que sea mucho más escalable, al contrario que RIP.

Los mensajes que tenemos son:

- Hello: para saludar a mis vecinos.
- Database description: para mandar la topología que conocemos.
- Link status request/update/ack: para consultar o enviar cambios.

2.3.6. Cabecera IP

La cabecera IP tiene 20 Bytes. Veamos los campos, en orden, que la componen:



- Versión (4 bits): La versión utilizada de IP, 0100 si IPv4 y 0110 si IPv6.
- Tamaño de cabecera (4 bits): longitud de la cabecera IP en palabras de 32 bits. Como mínimo 5 palabras y como máximo 15 palabras.
- Tipo de Servicio (8 bits): hace referencia a la calidad de servicio deseada durante el tránsito del paquete por una red. Hay redes que ofrecer prioridades de servicios, considerando determinados tipos de paquetes más prioritarios que otros.
- Longitud total (16 bits): es el tamaño total, en octetos, del datagrama, incluyendo el tamaño de la cabecera y el de los datos.

- Identificador (16 bits): identificador único del datagrama. Se utiliza en caso de que el datagrama deba ser fragmentado, para poder distinguir los fragmentos de un datagrama de los de otro.
- Flags (3 bits): En la actualidad se utiliza para especificar valores relativos a la fragmentación.
  - bit 0: Reservado, debe ser 0.
  - bit 1 (DF): 0 = divisible; 1 = no divisible.
  - bit 2 (MF): 0 = último fragmento; 1 = fragmento intermedio, le siguen más fragmentos.
- Desplazamiento (13 bits): en paquetes fragmentados indica la posición, en unidades de 64 bits, que ocupa dentro del datagrama original.
- Tiempo de vida, TTL (8 bits): indica el número de saltos máximo de un paquete en una red para evitar paquetes navegando en la red indefinidamente. En cada salto, el campo se reduce en una unidad, si llega a 0 se descarta.
- Protocolo (8 bits): tiene un identificador del protocolo de las capas superiores al que debe entregarse el paquete.
- Suma de Control de Cabecera (16 bits): es una comprobación de la corrección del datagrama. Se recalcula cada vez que algún nodo cambia alguno de sus campos. El método de cálculo consiste en suma en complemento a 1 cada palabra de 16 bits de la cabecera (poniendo 0 en este campo) y hacer el complemento a 1 del valor resultante. Así cuando llega al destino se hace esta misma operación y se comprueba si es correcta la cabecera.
- Dirección IP de origen (32 bits).
- Dirección IP de destino (32 bits).
- Opciones (opcional).
- Relleno: este campo tiene tantos como sea necesario para que la cabecera tenga un número de bits múltiplo de 32.

2.3.7. Fragmentación

Las redes no permiten paquetes de cualquier tamaño. El tamaño máximo es 2^{16} Bytes, aunque ninguna red suele aceptar dicho tamaño.

Las redes cuentan con un Maximum Transfer Unit (MTU): esta nos dice cuál es el tamaño máximo de lo que podemos transportar en la capa de datos a nivel de enlace, es decir, datos y cabecera de IP normalmente.

El MTU depende del estándar de una tarjeta:

- Ethernet: 1500 Bytes.

- Wifi: permite más pero normalmente el punto de acceso lo restringe a 1500 Bytes.

Tenemos los siguientes campos en la cabecera de fragmentación:

- Identificación: identifica al datagrama completo, no al paquete. Por esto, si un datagrama es fragmentado todos los fragmentos tendrán el mismo identificador.
- Campo de indicadores, como el *more fragments*: si hay más fragmentos será un 1, y si es el último será un 0; o el *don't fragment* que indica si un paquete puede ser fragmentado (si no puede y es necesario el paquete se descartará).
- Campo de desplazamiento (offset): sirve para ensamblar los paquetes en el destino.

Algunas observaciones importantes:

- Si hay algún error y no llegan todos los fragmentos de un datagrama se descarta todo y debe ser una capa superior la que se encargue de arreglar el problema.
- Solo fragmentamos cuando pasamos a un MTU menor y solo se ensamblan en el destino, puesto que distintos fragmentos pueden seguir caminos distintos, dependiendo del encaminamiento.

2.4. Asociación con la capa de enlace: el protocolo ARP

Cuando queremos enviar un paquete a un destino, mirando la tabla de encaminamiento sabemos la dirección IP del siguiente salto, pero para poder mandarle el paquete en el nivel de enlace necesitamos saber la dirección MAC.

A diferencia de las direcciones IP origen y destino, que no cambian en ningún salto (salvo que se use NAT); las direcciones MAC origen y destino cambian en cada reenvío del paquete, pues el nivel de enlace solo se encarga de encaminar punto a punto.

Entonces para mandar los paquetes necesitamos saber las direcciones MAC de los dispositivos. Esto se soluciona con el protocolo Address Resolution Protocol (ARP).

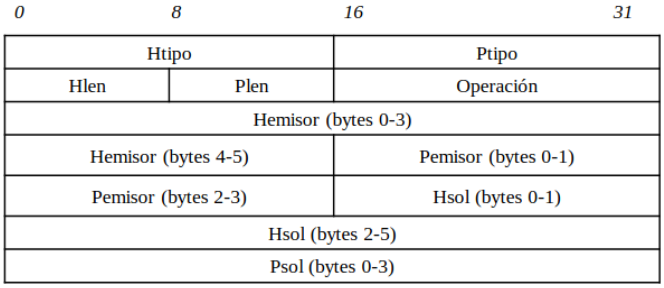
Funcionamiento

Supongamos que A quiere mandar un paquete a B y tiene como intermediario a R1. A sabe la IP de R1 (por las tablas de encaminamiento) pero necesita saber la dirección MAC.

- A manda un ARP Request a nivel de enlace por la dirección FF:...:FF (la dirección de difusión a nivel de enlace), preguntando por la MAC de la IP que conoce.

- R1 contesta con un mensaje ARP Reply con su dirección MAC en unicast (es decir, respuesta única, no difusión) pues se conoce la MAC del que pregunta.

Este proceso no se hace siempre, sino se introduciría mucho tráfico. Las direcciones MAC que recibimos se van guardando en una caché y cuando pasa mucho tiempo expiran.



Campos de la cabecera: (H es referido a hardware, para nivel 2 para abajo; P es referido a protocolo, para nivel 3 para arriba):

- Htipo: protocolo que se usa en el nivel de enlace.
- Ptipo: protocolo que se usa en el nivel de red.
- Hlen: longitud de la dirección hardware.
- Plen: longitud de la dirección del protocolo de red.
- Operación: Request o Reply.
- Hemisor: dirección hardware del emisor (MAC).
- Pemisor: dirección de red del emisor (IP).
- Hsol: dirección MAC del receptor.
- Psol: dirección IP del receptor.

Dependiendo de la operación que sea se rellenarán unos campos u otros.

2.5. El protocolo ICMP

Es un protocolo que no es imprescindible pero ayuda. Sirve en general para informar al origen de que ha habido un error. Este protocolo es útil pues IP no arregla ningún tipo de problema, pero al menos por este medio informa para que las capas superiores decidan si tomar acción.

Cuando ocurre un error, el equipo manda un paquete ICMP al origen. Es un nivel de red que se encapsula dentro de IP.

Paquete ICMP

La cabecera es una palabra de 32 bits:

- Tipo de mensaje, que indica el tipo de error que ocurrió.
- Código (subtipo de mensaje), para especificar más el error.
- Comprobación.

La parte de datos contiene los primeros 64 bytes del paquete que provocó el error, los 20 de la cabecera y 44 de datos del paquete IP. Esto es para ubicar el paquete que provocó el error.

De esta forma lo que tenemos es una cabecera IP y en la parte de datos un paquete ICMP, que a su vez se compone por una cabecera y en los datos tenemos una cabecera IP y algunos datos del paquete problemático.

Campo *tipo*	Mensaje ICMP
8/0	Solicitud/respuesta de eco (ping)
3	Destino inalcanzable
4	Ralentización del origen
5	Redireccionamiento
11	Tiempo de vida excedido
12	Problema de parámetros
13/14	Solicitud/respuesta de sello de tiempo
17/18	Solicitud/respuesta de máscara de red

2.6. Autoconfiguración de la capa de red (Dynamic Host Configuration Protocol (DHCP))

Es un protocolo automático para asignar direcciones IP, máscaras, pasarelas por defecto e IP del servidor DNS. Funciona a nivel de red pero se implementa en capa de aplicación, se encapsula en UDP.

Al principio cuando no tenemos IP, la IP es 0.0.0.0. Para conseguir una IP se intercambian los siguientes mensajes entre el cliente y el servidor DHCP:

- DHCP Discover: lo primero que hace es preguntar si hay alguien.
- DHCP Offer: se presenta el servido y le ofrece una IP a usar. Esto es solo una oferta, no una imperativa.
- DHCP Request: el equipo comprueba si ya ha tenido una IP en la red y solicita tener esa IP en caso de que sí.
- DHCP ACK: el servidor responde con la IP definitiva, y esta sí es imperativa.

La IP destino usada en toda la transacción es la de difusión 255.255.255.255, y la de origen es la 0.0.0.0 en caso de ser el cliente o la del servidor DHCP en caso de ser el mismo. Los puertos usados son el 67 para el servidor y el 68 para el cliente.

Los pares pregunt-respuesta se etiquetan con un identificador de transacción para que el cliente sepa que el mensaje va para él.

DHCP es un protocolo de *leasing* o alquiler, la IP (y el resto de cosas) se asignan de forma temporal. Cuando tiempo va a expirar es necesario que el equipo mande una solicitud para que el servidor refresque el alquiler. El **DHCP release** se manda cuando queremos liberar la IP que se nos ha asignado, por ejemplo antes de que el dispositivo se apague. Sin embargo, si el servidor no recibe una petición del equipo antes de que se expire su alquiler libera la IP igualmente, dado que es posible que el equipo ya no la esté usando y sea necesario liberarla para no quedarnos sin IPs disponibles.

Observación. Es posible configurar un servidor para que algunas IPs fijas se asignen a ciertos dispositivos.

3. Seguridad en redes

Objetivos

- Comprender la importancia de la seguridad en las comunicaciones y aprender cómo desplegar mecanismos básicos de seguridad en redes de computadores e Internet.
- Conocer los aspectos de seguridad en redes: confidencialidad, autentificación, no repudio, integridad y disponibilidad.
- Entender los conceptos básicos de la seguridad en redes, como el uso de algoritmos de clave secreta, de clave pública, intercambio de claves...
- Comprender qué son los certificados digitales y las autoridades de certificación, y los diferentes mecanismos que se pueden implementar con certificados.
- Conocer algunos de los principales protocolos de comunicación seguros, como TLS e IPSec, y los mecanismos que lo utilizan.

3.1. Introducción

Una red de comunicaciones es **segura** cuando se garantizan todos los aspectos de seguridad, con lo que no hay protocolos ni redes 100 % seguras. Definamos brevemente los aspectos de seguridad que vamos a estudiar:

- **Confidencialidad / privacidad:** cuando transmitimos algo a un receptor queremos que solo dicho receptor sea capaz de ver el mensaje. Se consigue con el cifrado.
- **Autenticación:** las entidades son quien dicen ser. Se consigue con algoritmos de Reto-Respuesta o doble cifrado.
- **No repudio o irrenunciabilidad:** no podemos renunciar de haber participado en una transacción, es una prueba legal ante un juez. Se consigue con la firma deigital o con el doble cifrado con certificado, pero ha de haber una entidad fiable.
- **Integridad:** que los datos no sean manipulados por el camino. Se consigue con funciones hash o compendios (resúmenes).
- **Disponibilidad:** el sistema mantiene las prestaciones de los servicios independientemente de la demanda. (No se ve en la asignatura).

Debe haber seguridad en todos los niveles de la red, el grado de seguridad lo fija el punto más débil.

Definición 3.1 (Ataque de seguridad)**.** Cualquier acción intencionada o no que menoscaba cualquiera de los aspectos de seguridad.

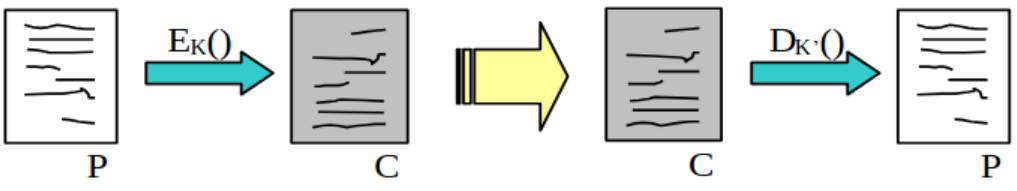
- **Sniffing:** escuchar comunicaciones, por ejemplo Wireshark.
- **Snooping (phishing):** suplantar a alguien.
- **Man in the Middle:** se pone alguien en medio de dos equipos que se comunican y intercepta todos los mensajes que se transmiten.
- **Distributed Denial of Service (DDoS):** mandar muchas peticiones hasta conseguir que el servicio deje de funcionar. Distributed si los ataques vienen de muchos sitios.
- **Malware:** software malicioso. Troyanos, gusanos, spyware, backdoors, rootkits, keyloggers, ransomware (se encriptan todos o parte de los datos y se pide un recate de los datos).

Los mecanismos de seguridad que vamos a estudiar son:

- Cifrado (simétrico y asimétrico)
- Autentificación con clave secreta (reto-respuesta)
- Intercambio de Diffie-Hellman (establecimiento de clave secreta)
- Funciones Hash (Hash Message Authentification Code)
- Firma digital
- Certificados digitales

3.2. Cifrado

Se trata de un procedimiento para garantizar la confidencialidad. Tenemos un texto plano a transmitir (P) y lo ciframos con una función E_k, que dará lugar a un texto cifrado (C), el cual se mandará a través del canal de comunicaciones. Llegará al otro extremo y será descifrado con una función $D_{k'}$ de desencriptado. Los algoritmos de cifrado y descifrado normalmente son conocidos y la dificultad reside en las claves k y k'.



Veremos dos tipos de algoritmos de cifrado:

- Cifrado simétrico (clave secreta), misma clave distintas funciones.
- Cifrado asimétrico (clave privada y clave pública), distinta clave misma función.

3.2.1. Cifrado simétrico

Se llama simétrico porque se usa la misma clave para cifrar y para descifrar. Esto significa que es una clave secreta que comparten los dos.

DES, (Data Encryption Standard)

Un algoritmo que se suele usar se basa en realizar permutaciones y funciones XOR encadenadas. Al final lo que obtenemos es una sustitución, por lo que con la misma entrada el resultado siempre será el mismo. Por tanto, DES no es más que un esquema de sustitución que usa palabras de 56 bits. Para arreglar la posibilidad de descifrarlo, se utiliza un esquema reentrante, donde la salida de aplicar una transformación se usa para la creación del cifrado de la siguiente palabra a cifrar. Se cifran palabras de 64 bits. De este modo, quien recibe el mensaje codificado necesita conocer la última entrada usada para codificar y podrá aplicar el proceso inverso.

DES doble y 3DES

Son mejoras de DES que proporcionan robustez al algoritmo. Se toman dos claves k_1 y k_2 y para cifrar se toma una función E y su inversa D y se concatenan E_{k_1}, D_{k_2}, E_{k_1} y para descifrar se concatenan D_{k_1}, E_{k_2}, D_{k_1}.

IDEA

Usa la misma idea que DES pero con claves de 128 bits, lo que aumenta la complejidad de violación del protocolo.

3.2.2. Cifrado asimétrico

Cada usuario A tiene una clave pública K_{pub_A} y una clave privada K_{pri_A} distintas. Conociendo la pública no es posible conocer la privada, por lo que la pública la conocen todos pero la privada solo la conoce A.

$$C = K_{pub_A}(P) \rightarrow P = K_{pri_A}(C)$$

$$C = K_{pri_A}(P) \rightarrow P = K_{pub_A}(C)$$

Esta es la forma de funcionar de cualquier cifrado con clave pública. Además hay una correspondencia biunívoca entre las claves públicas y privadas.

RSA

Es un algoritmo para sacar las claves del cifrado asimétrico.

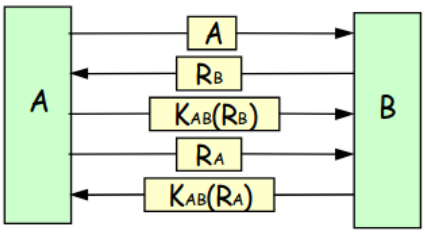
1. Elegimos p y q primos grandes ($> 10^{100}$)
2. $n = p \cdot q$ y $z = (p-1) \cdot (q-1)$
3. Elegimos d primo relativo de z
4. Calculamos e tal que $e \cdot dmodz = 1$
5. $K_pub = (e, n)$ y $K_{pri} = (d, n)$ de tal forma que:
   - $C = P^e modn$
   - $P = C^d modn$

3.3. Autenticación

Pongámonos en el supuesto de que dos equipos, A y B, quieren autenticarse. Lo más sencillo es que cada uno tenga una BBDD con el usuario y la clave que comparten con el otro y que cada uno le mande a otro su usuario y la clave y que el servidor confirme si son correctos o no. Esto es vulnerable pero se usa en muchos servicios. Se hacen algunas mejoras de este método como por ejemplo que el envío de información se haga a través de túneles cifrados.

3.3.1. Reto-respuesta

Tenemos como antes una BBDD con la clave que comparten A y B. Supongamos ahora que A quiere conectarse con B. Lo primero que hace es enviarle su identidad, lo que no es un dato sensible. Lo que ahora manda B es un reto, un número aleatorio. A calcula X, un cifrado con la clave compartida del reto. B cifra el reto también, X' y cuando A le responde comprueba que X=X' y así autentica a A. Ahora falta autenticar a B, y esto se hace de la misma forma.

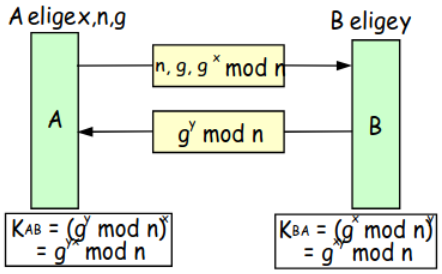


Este sistema tiene algunas vulnerabilidades:

- **Ataque por repetición:** el atacante escucha por mucho tiempo y va guardando las respuestas, hasta que se repita un reto y así puede responder. **Solución:** que el reto no se pueda repetir (NONCE), Ej: fecha + hora.
- **Ataque por reflexión:** el atacante antes de responder manda el mismo reto, espera a la respuesta y la reenvía. **Solución:** usar dominios de claves disjuntas.

3.3.2. Intercambio de Diffie-Hellman

Permite establecer una clave secreta entre dos entidades a través de un canal no seguro.



Este sistema es vulnerable al ataque man-in-the-middle:
El atacante se pone en medio e intercepta el mensaje de A, y le manda su propia clave a B, y esa misma clave se la responde a A, y espera que B le responda. De esta forma, hace de mensajero invisible, A y B ni siquiera saben que están escuchando sus mensajes.

3.4. Funciones Hash

Son funciones de forma que dada una palabra P, nos da una palabra a modo de resumen o compendio de los datos, R. P puede ser de cualquier longitud, R, sin embargo, suele ser de longitud fija y ademas la función es unidireccional, irreversible, es decir, no se puede obtener P a partir de R. Dado un mensaje P, se envía P junto con su hash.

Para que no se pueda modificar el mensaje a P' e incluir su resumen se tienen varias alternativas:

- En el resumen podemos meter la clave que se comparte. A este hash (P + K_{AB}) se le suele llamar MAC (Message Authentification Code).
- Cifrar el hash con la clave compartida.

Así conseguimos integridad del mensaje y por otra parte autenticación con los MAC.

MD5 (Message Digest)

Dada una palabra nos da un resumen de 128 bits. El algoritmo trabaja sobre bloques de 512 bits, por lo que si la palabra no tiene un número de bits múltiplo de 512 rellenamos con 100...0 hasta que sea congruente con 448módulo 512, y se le añade un campo de longitud de 64 bits. A continuación se divide el mensaje en bloques de 512 bits y hacemos un procesamiento secuencial por bloques, es decir, que cada salida sirve como entrada para la siguiente caja MD5.

SHA (Secure Hash Algorithm)

Funciona igual que MD5 pero los resúmenes son de 160 bits.

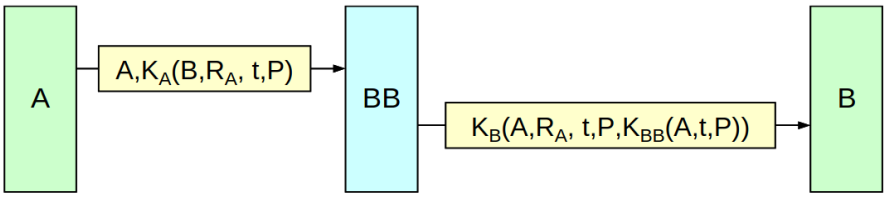
3.5. Firma digital y certificados digitales

Una **firma digital** intenta ser un sustituto de una firma escrita para poder garantizar el **no repudio** en nuestras acciones en Internet. Con ellas conseguimos:

- Autenticación por parte del receptor de la identidad del emisor.
- No repudio por parte del firmante.
- El emisor obtiene garantía de no falsificación. (Integridad).

3.5.1. Firma con clave secreta o Big Brother (BB)

El protocolo consiste en usar una entidad en la que todos los usuarios confían y que vigilará las transacciones de todos los usuarios. Es una especie de man-in-the-middle. Si A quiere enviar un mensaje a B, el BB formará parte de la comunicación haciendo de puente que guardará cada una de las transacciones realizadas.



El BB tiene una clave compartida con todos. A le manda a BB:

A: Identificador.

B: El destinatario.

R_A**:** Un resumen para dar integridad.

t: el instante de tiempo.

P: texto plano, el mensaje a enviar.

Todo menos el identificador va cifrado con la clave que comparte A con BB. El mensaje es recibido y reenviado por BB añadiéndole algunos detalles.

$K_{BB}(A, t, P)$**:** Esta clave solo la tiene el BB lo que prueba ante un juez quién ha hecho dicha transacción, en qué momento y el contenido de la transacción.

$K_B(\ldots)$**:** Además todo va cifrado con la clave de B para confidencialidad.

3.5.2. Firma digital con clave asimétrica o Doble cifrado

Supongamos que A le quiere mandar un mensaje a B. La idea se basa en lo siguiente:

- $k_{pri_A} \equiv$ autenticación, solo A ha podido cifrarlo.
- $k_{pub_B} \equiv$ confidencialidad, solo B podrá descifrarlo.

De esta manera, juntando las dos obtenemos autenticación + confidencialidad: $k_{pub_B}(K_{pri_A}(P))$ (puede cifrarse al revés, no es relevante).

Sin embargo todo esto no garantiza el **no repudio**, puesto que nada nos garantiza que A sea el dueño de la clave. Para garantizarlo necesitamos los certificados digitales que deben ser emitidos por autoridades de certificación, que son entidades fiables.

Definición 3.2 (Autoridades de certificación (AC))**.** Entidad que garantiza la asociación entre identidad y claves.

Certificado digital

El usuario obtiene sus claves pública y privada, envía una solicitud, firmada digitalmente, a la AC indicando su identidad y clave pública. La AC comprueba la firma y emite el certificado: este tiene los datos que a continuación detallaremos y va firmado digitalmente por la clave firmada de la AC con objeto de que el certificado no pueda falseficarse. Campos de un certificado X.509:

- Identidad del usuario.
- Su clave pública.
- La AC que lo ha emitido.
- Periodo de validez.
- Algunos datos más como la versión del certificado, el número de serie...

3.6. Protocolos seguros

La seguridad se divide en dos tipos:

- **Perimetral:** uso de *firewalls*, *sistemas de detección de intrusiones* o *sistemas de respuesta*.
- **Seguridad en protocolos:** consiste en usar protocolos para garantizar seguridad.
  - Capa de aplicación: PGP o SSH.
  - Capa de sesión (entre aplicación y transporte): TLS, SSL.
  - Capa de red: IPSec.

3.6.1. PGP (Pretty Good Privacy)

Es un protocolo de correo electrónico seguro.

Emisor:
- R = MD5(P)
- FD = $Kpr_A(R)$
- Z = ZIP(FD + P)
- C = $IDEA_K(Z)$ + $Kpu_B(K)$
- M = B64(C)

Receptor:
- C = $B64^{-1}(M)$
- K = $Kpr_B(Kpu_B(K))$
- Z = $IDEA_K^{-1}(IDEA_K(Z))$
- FD + P = $ZIP^{-1}(Z)$
- R = $Kpu_A(FD)$
- R' = MD5(P)
- R' = R ??

El emisor hace un resumen del mensaje, lo firma; esto lo comprime junto con el mensaje, cifra esa compresión con IDEA y una clave de sesión K (generada solo para esa sesión), que manda cifrada con la clave pública del receptor y esto lo codifica con base 64 (esto no tiene nada que ver con seguridad). El resultado de esto último es lo que le manda al destinatario. El receptor simplemente tiene que hacer el proceso inverso para sacar el mensaje y con el resumen comprobar que no se ha modificado.

Con este proceso conseguimos:

- Confidencialidad: el mensaje va cifrado.
- Integridad: gracias al resumen.
- Autenticación: gracias al cifrado con la clave privada.
- No repudio: solo si hay un certificado digital.

3.6.2. TLS, SSL

Se usan para muchos protocolos (HHTPS, IMAPS, SSL-POP). TLS fue el original y SSL se acabó popularizando más. Hacen más o menos lo mismo, pero no son compatibles. Lo que hacen en esencia es crear túneles cifrados.

SSL

No es un protocolo sino una familia de ellos.

- SSL Record Protocol: encapsula los protocolos y ofrece un canal seguro con privacidad, autenticación e integridad. Lo que se hace es que cuando voy a mandar datos, los parto en fragmentos, cada fragmento lo comprimos, hago un resumen y cifro datos + resumen. Y esto último es lo que se transmite, encapsulado en un paquete TCP.
- SSL Handshake Protocol: se negocia el algoritmo de cifrado, la función hash, autentica al servidor, el cliente genera claves de sesión (temporales) con el algoritmo de Diffie-Hellman o aleatorias y que irán cifradas con la clave pública del servidor.
  El problema del hombre en medio se resuelve autenticando al servidor. Así, si el mensaje que me llega con $g^y modn$ no viene del servidor no lo tomo.

- SSL Assert Protocol: informa sobre errores en la sesión.
- Change Cipher Spec Protocol: para notificar cambios en el cifrado.

3.6.3. IPSec

Su objetivo es garantizar autenticación, integridad y opcionalmente privacidad a nivel IP. Crea túneles unidireccionales.

Definición 3.3 (Túnel)**.** Es un sitio donde entra un paquete y a la salida tendremos exactamente el mismo paquete. Para ello encapsulamos el paquete dentro de otro paquete. Si el túnel va cifrado (la parte de los datos va cifrada) entonces el túnel es seguro.

Son tres procedimientos:

1. Establecimiento de una "Asociación de seguridad": con el objetivo de establecer una clave secreta (Diffie-Hellman), con la previa autenticación. Es simplex. Vulnera el carácter NO orientado a conexión.
2. Garantizar la autenticación e integridad de los datos mediante las "Cabeceras de autenticación".
3. (Opcional) Garantizar la privacidad de los datos mediante el protocolo de "Encapsulado de seguridad de la carga".

Dos tipos de túneles:

- Modo transporte: la asociación se hace extremo a extremo entre el host origen y destino.
- Modo túnel: la asociación se hace entre dos routers intermediarios. Útil por ejemplo si una empresa quiere comunicar dos sucursales, en vez de comunicar cada dos trabajadores, se comunican mediante esos routers intermediarios y solo hacen falta dos túneles (para ambas direcciones).

4. Relaciones de Problemas

4.1. Introducción

Ejercicio 4.1.1. Explique brevemente las funciones de cada una de las capas del modelo de comunicación de datos OSI.

El modelo de comunicación de datos OSI cuenta con 7 niveles o capas:

1. Capa física: Se encarga de la parte física de la transmisión de los datos. Encontramos distintas formas de codificar los bits para su envío.

   Realiza funciones adicionales, como la codificación del canal.

2. Capa de enlace: Se encarga de los mecanismos de acceso al medio. En caso de haber un medio compartido por varios dispositivos, debe encargarse de no transmitir datos cuando otro medio lo está haciendo y de hacerlo cuando el canal se encuentre libre.

   En esta capa nos encontramos con los protocolos MAC y LLC.

3. Capa de red: Se encarga principalmente del direccionamiento de equipos (saber dar una dirección y tener un identificador dentro de la red) y del encaminamiento de datos (saber cómo mandar los paquetes al destinatario).

4. Capa de transporte: Se encarga principalmente de la fiabilidad de las comunicaciones:

   - Corrección de errores.
   - Manejar el congestionamiento de la red.
   - Controlar el flujo de datos (reducir velocidades si el receptor no es capaz de adecuar la velocidad de recibo a la de envío).
   - Realizar la multiplexación de los datos (ya que un mismo equipo puede tener varias aplicaciones que estén recibiendo datos a la vez).

5. Capa de sesión.

6. Capa de presentación[1].

7. Capa de aplicación: Se encarga de decidir qué datos envía a qué equipo, así como de interpretar los datos recibidos por otros equipos.

[1] No se han mencionado en clase las funcionalidades de las capas de presentación ni de sesión.

Ejercicio 4.1.2. Si la unidad de datos de protocolo en la capa de enlace se llama trama y la unidad de datos de protocolo en la capa de red se llama paquete, ¿son las tramas las que encapsulan los paquetes o son los paquete los que encapsulan las tramas? Explicar la respuesta.

Son las tramas las que encapsulan a los paquetes, ya que son las capas inferiores (en este caso, la de enlace) las que encapsulan la información de las capas superiores (en este caso, la de red) para su envío.

De esta forma, los paquetes son el PDU de la capa de red, que se convierte en el SDU de la capa de enlace, la cual añade su cabecera al mismo convirtiéndolo en su PDU.

Ejercicio 4.1.3. Averigüe qué son los sistemas de representación de datos "*Little Endian*" y "*Big Endian*". ¿Puede un host que utilice representación *Little Endian* interpretar mensajes de datos numéricos provenientes de un host que utilice representación *Big Endian* y viceversa? Discuta la respuesta.

Sí que puede, para ello, debe haber un determinado protocolo que permita indicar qué codificación llevan los datos en binario. De esta forma, en alguna parte de la cabecera de los paquetes enviados, debe haber un bit que indique si los valores numéricos que se envían estén en *Big Endian* o en *Little Endian*.

Ejercicio 4.1.4. Cuando se intercambia un fichero entre dos hosts se pueden seguir dos estrategias de confirmación. En la primera, el fichero se divide en paquetes que se confirman individualmente por el receptor, pero el fichero en conjunto no se confirma. En la segunda, los paquetes individuales no se confirman individualmente, es el fichero entero el que se confirma cuando llega completo. Discutir las dos opciones.

Suponiendo que enviamos n paquetes de datos, la primera forma envía de vuelta al emisor n paquetes de confirmación, uno por cada paquete. De la segunda forma, el receptor espera a unir todos los paquetes en un fichero completo (y a verificar que no se ha perdido ningún paquete del mismo) para enviar el mensaje de verificación.

De la segunda forma se envían menos mensajes de verificación al emisor, por lo que la posibilidad de congestión de red por paquetes de verificación es menor. Sin embargo, en caso de que un paquete no consiga llegar o llegue en mal estado, no será hasta el final del envío de todos los paquetes que el receptor no genere el mensaje al emisor, por lo que en caso de errores en la comunicación, hay un mayor tiempo en la comunicación, al tener que esperar a que el receptor tenga todos los paquetes. Además, en el caso de error, el receptor no informará de qué paquete ha llegado mal, por lo que deberá pedir al emisor que reenvíe todos los paquetes de nuevo.

Resumiendo, ambas estrategias de confirmación tienen sus pros y sus contras. Dependiendo de la situación (si queremos mayor velocidad en la comunicación o si queremos menor saturación de red), puede interesarnos una u otra.

Ejercicio 4.1.5. ¿Para qué sirve el programa *ping*? ¿y el programa *traceroute*?

El programa *ping* usa el protocolo ICMP para enviar un paquete a un equipo, el cual tratará de responder con un paquete de confirmación de recepción del primer paquete. Sirve para comprobar la conexión y el buen funcionamiento de la red existente entre dos equipos. También sirve para calcular empíricamente la latencia de la conexión.

Por otra parte, el programa *traceroute* sirve para consultar todos los nodos intermedios por los que pasan los paquetes que salen de un emisor y llegan a un receptor, junto con la latencia de cada salto. Se usan varios paquetes ICMP con un valor creciente del campo TTL (1,2,...) para que cada salto intermedio devuelva un paquete de error ICMP por TTL excedido. De esta forma, el emisor puede saber cuántos saltos intermedios hay entre él y el receptor, así como la latencia de cada uno de ellos.

Ejercicio 4.1.6. ¿Qué protocolos de un paquete puede cambiar un router? ¿En qué circunstancias?

Un router puede cambiar los protocolos situados debajo de la capa de red, siempre que sea necesario debido a que las redes que interconecta tengan dichos protocolos diferentes.

Por ejemplo, una red doméstica típica es aquella basada en Wi-Fi y con acceso a Internet contratado con tecnología ADSL. En este caso, el router inalámbrico deberá modificar el protocolo de las capas físicas y de enlace convenientemente.

Ejercicio 4.1.7. Averigue qué ISP operan en España.

Algunos de los ISP que operan en España son:

- Movistar.
- Vodafone.
- Orange.
- Jazztel.
- MásMóvil.
- Yoigo.
- Digi.

Ejercicio 4.1.8. ¿Qué es una aplicación cliente-servidor? ¿y una aplicación *peer-to-peer*?

Una aplicación cliente-servidor es una aplicación que depende de otra que probablemente esté en un equipo remoto (llamado servidor) para su funcionamiento.

Un ejemplo de aplicación cliente-servidor es una página web: tenemos aplicaciones que se ejecutan en local en cada equipo que accede a una determinada url. Dicha aplicación solicita datos a una aplicación que se encuentra en un equipo remoto, la cual proporciona datos (por ejemplo, accediendo a una base de datos) como respuesta

a los datos solicitados por la aplicación que se ejecuta en cada equipo de forma local.

Por otra parte, una aplicación *peer-to-peer*[2] es una aplicación que se distribuye entre varios equipos (que pueden estar muy lejanos entre sí) de forma que todas las aplicaciones tienen la misma relevancia en el buen funcionamiento del sistema.

Ambos tipos de aplicaciones se estudiarán en el Capítulo dedicado a la Capa de Aplicación.

Ejercicio 4.1.9. Describa brevemente la diferencia entre un *switch*, *router* y un *hub*.

Para responder a la pregunta, usamos además información que hemos aprendido en el Tema 2:

- Un switch es un nodo en una red que permite conectar tantos equipos como deseemos (normalmente, estos tienen 48 bocas de entrada RJ45 en el caso de conectar los equipos por ethernet) a una red. Funcionan a nivel de enlace, luego no tienen una dirección IP asociada.
- Un router es un nodo en una red que permite conectar redes distintas entre sí. Para ello, disponen de distintas tarjetas de red, cada una asociada a una red que se encuentra conectada al router. Disponen por tanto de varias direcciones IP, una por cada red a la que se conecta. Funciona a nivel de red.

  Presenta en su interior la tabla de enrutamientos, que permite el encaminamiento en la capa de red. Cuenta además con el NAT, que permite traducir direcciones de IP privadas a públicas y viceversa.
- Un *hub* es un nodo en una red que permite implementar la difusión. Se trata de un conjunto de bocas ethernet que internamente funcionan como un bus. Cada vez que un paquete se envía por una de las bocas, este es reenviado a todas las demás bocas, por lo que todos los equipos conectados al *hub* reciben el paquete.

Ejercicio 4.1.10. ¿Qué diferencia, en el contexto de una red de computadores, existe entre la tecnología de difusión y la tecnología punto-a-punto?

La tecnología de difusión permite enviar un paquete desde un equipoo y hacer que este sea recibido por el resto de equipos que estén conectados a la misma red (o hacer llegar estos a equipos en distintas redes). Es cada dispositivo el que decide si el paquete es para él o no.

Por otra parte, la teconología punto-a-punto permite el envío de paquetes desde un equipo a otro usando un medio directo, por lo que el destino está implícito desde que se envía el paquete. Esta tecnología es más rápida y segura, pero su escalabilidad es mucho menor.

Ejercicio 4.1.11. Un sistema tiene una jerarquía de protocolos de n capas. Las aplicaciones generan mensajes de M bytes de longitud. En cada capa se añade una cabecera de h bytes. ¿Qué fracción del ancho de banda de la red se llena con cabeceras? Aplique el resultado a una conexión a 512 kbps con tamaño de datos de 1500

[2]En español, podemos pensar en "entre pares".

bytes y 4 capas, cada una de las cuales añade 64 bytes cabecera. ¿Qué velocidad real de envío de datos resulta?

Debemos sumar a los M bytes iniciales que proporcionan las aplicaciones n veces (la capa de aplicación también incluye una cabecera) h bytes, por lo que la longitud de los mensajes que de verdad se envían es de $n \cdot h + M$ bytes. Por tanto, por cada $n \cdot h + M$ bytes enviados, $n \cdot h$ de ellos son de cabeceras:

$$\frac{n \cdot h}{n \cdot h + M} \cdot 100 \ \% \text{ de ancho de banda que se llena de cabeceras}$$

Si ahora partimos de 1500 bytes iniciales y añadimos 4 veces (una por capa) 64 bytes, estamos en realidad enviando mensajes de longitud:

$$4 \cdot 64 + 1500 = 256 + 1500 = 1756 \text{ bytes}$$

Por tanto, el ($^{256}/_{1756} = 0{,}145786$) 14.58 % de la red se emplea para enviar cabeceras, luego se aprovecha el ($1 - 0{,}145786 = 0{,}854214\,\%$) 85.42 % de la red para el envío de datos.

Si enviamos paquetes a una velocidad de 512kbps, en realidad estaremos enviando datos a una velocidad real de:

$$0{,}854214 \cdot 512 = 437{,}357568 \text{ kbps}$$

Ejercicio 4.1.12. Clasifique como de *difusión* o *punto a punto* cada uno de los siguientes sistemas de transmisión:

1. Radio y TV: Difusión, ya que es un emisor (en este caso, una cadena de televisión o radio) que difunde paquetes a cualquiera que tenga sintonizado dicho canal.

2. Redes inalámbricas (WLAN): Difusión, ya que cualquier equipo puede conectarse a la red y recibir los paquetes que se envían de forma inalámbrica.

3. ADSL: Punto a punto, ya que la conexión se establece mediante un cable. Usa en medio único.

4. Redes de cable: Puede implementar ambas tecnologías, ya que puede ser punto a punto (si cada equipo tiene su propio cable) o de difusión (si todos los equipos comparten el mismo cable).

5. Comunicaciones móviles (por ejemplo, GSM, UMTS, ...): Difusión, ya que (de nuevo) cualquier equipo puede recibir los paquetes que se envían por la red.

Ejercicio 4.1.13. Clasifique los siguientes servicios como orientados a conexión/no orientados a conexión y confirmados/sin confirmación. Justifique la respuesta.

Recordamos que los medios orientados a conexión son aquellos que comprueban si el receptor está disponible antes de enviar la información, y que los confirmados son aquellos que confirman la recepción del mensaje.

1. Correo postal ordinario: No es orientado a conexión (ya que no comprobamos anteriormente que el destinatario esté disponible, simplemente enviamos la carta) y es sin confirmación, ya que tras enviar la carta nada nos garantiza que el destinatario nos responda, pese a pedirlo explícitamente.

2. Correo certificado: No es orientado a conexión por la misma razón que el correo normal. Sin embargo, es confirmado, ya que al recibir el destinatario la carta debe firmar para que el emisor sea consciente de que la carta ha sido recibida.

3. Envío y recepción de fax: Orientado a conexión y confirmado.

4. Conversación telefónica. Es orientado a conexión, puesto que no se puede establecer una llamada si la otra persona no coge el teléfono. Además, es confirmado, ya que la otra persona ha de responder para que se produzca una comunicación.

5. Domiciliación bancaria de recibos: No es orientado a conexión, ya que no se comprueba si el destinatario está disponible antes de enviar la información. Además, es confirmado, puesto que el banco envía un mensaje de confirmación al emisor del recibo.

6. Solicitud de certificado de empadronamiento. Es no orientado a conexión, ya que no se comprueba si el destinatario está disponible antes de enviar la solicitud. Además, es con confirmación, ya que el ayuntamiento envía un documento que certifica que el solicitante está empadronado.

Ejercicio 4.1.14. ¿Cuál es el tiempo necesario en enviar un paquete de 1000 Bytes, incluidos 50 Bytes de cabecera, por un enlace de 100 Mbps y 10Km? ¿cuál es el tiempo mínimo desde que se envía hasta que se recibe confirmación? ¿qué relación hay entre este tiempo y los temporizadores en, por ejemplo, las capas de enlace y transporte?

En primer lugar, hemos de calcular el retardo de transmisión T_t, que es el tiempo que se tarda en enviar el paquete por el enlace. Para ello, tenemos que:

$$T_t = 10^3\,\mathrm{B} \cdot \frac{8\,\mathrm{b}}{1\,\mathrm{B}} \cdot \frac{1\,\mathrm{s}}{100 \cdot 10^6\,\mathrm{b}} = 80 \cdot 10^{-6}\,\mathrm{s} = 80\,\mu\mathrm{s}$$

Por otra parte, el retardo de propagación T_p es el tiempo que se tarda en enviar el paquete por los 10Km de cable. Para ello, suponiendo que la velocidad de transmisión es $^2/_3 c = 2 \cdot 10^8\,\mathrm{m/s}$, tenemos que:

$$T_p = 10 \cdot 10^3\,\mathrm{m} \cdot \frac{1\,\mathrm{s}}{2 \cdot 10^8\,\mathrm{m/s}} = 50 \cdot 10^{-6}\,\mathrm{s} = 50\,\mu\mathrm{s}$$

Por tanto, el tiempo total que se tarda en enviar el paquete es de $T_t + T_p = 130\,\mu\mathrm{s}$.

Veamos ahora el tiempo mínimo desde que se envía hasta que se recibe confirmación. Además de los tiempos anteriores, hemos de tener en cuenta el tiempo de procesamiento del paquete, el retardo de transmisión del paquete de confirmación y el retardo de propagación del paquete de confirmación. Tenemos que:

- No se proporciona información del tiempo de procesamiento del paquete. No obstante, en los dispositivos modernos, este retardo es de varios órdenes de magnitud menor que los otros, por lo que se puede considerar despreciable.
- El retardo de propagación del paquete de confirmación es el mismo, puesto que la distancia recorrida es la misma.
- El retardo de transmisión sí difiere, puesto que el tamaño del paquete de confirmación difiere. Normalmente, estos solo incluyen una cabecera, por lo que este tiempo, notado por T_{ACK}, es:

$$T_{\text{ACK}} = 50\,\text{B} \cdot \frac{8\,\text{b}}{1\,\text{B}} \cdot \frac{1\,\text{s}}{100 \cdot 10^6\,\text{b}} = 4 \cdot 10^{-6}\,\text{s} = 4\,\mu\text{s}$$

  Un temporizador de control de flujo en capa de enlace o de transporte debe ser suficientemente mayor a este tiempo mínimo para evitar un re-envío inmediato de paquetes ante cualquier eventualidad mínima en la red, como un retardo en las colas (mayor retardo de procesamiento) por un cierto nivel de congestión.

Por tanto, el tiempo total mínimo que se tarda en enviar el paquete y recibir confirmación es de:

$$T_{\text{total}} = T_t + T_p + T_{\text{ACK}} + T_p = 80\,\mu\text{s} + 50\,\mu\text{s} + 4\,\mu\text{s} + 50\,\mu\text{s} = 184\,\mu\text{s}$$

4.2. Capa de red

Ejercicio 4.2.1. Estime el tiempo involucrado en la transmisión de un mensaje de datos para la técnicas de conmutación de circuitos (CC) y de paquetes mediante datagramas (CDP) y mediante circuitos virtuales (CPCV) considerando los siguientes parámetros:

- M: longitud en bits del mensaje a enviar.
- V: velocidad de transmisión de las líneas en bps.
- P: longitud en bits de los paquetes, tanto en CPD como en CPCV.
- H_d: bits de cabecera de los paquetes en CPD.
- H_c: bits de cabecera de los paquetes en CPCV.
- T: longitud en bits de los mensajes intercambiados para el establecimiento y cierre de conexión, tanto en CC como en CPCV.
- N: número de nodos intermedios entre las estaciones finales.
- D: tiempo de procesamiento en segundos en cada nodo, tanto en CC como en CPD y en CPCV.
- R: retardo de propagación, en segundos, asociado a cada enlace, en CC, en CPD y en CPCV.

Ejercicio 4.2.2. Un mensaje de 64 kB se transmite a lo largo de dos saltos de una red. Ésta limita la longitud máxima de los paquetes a 2 kB y cada paquete tiene una cabecera de 32 bytes. Las líneas de transmisión de la red no presentan errores y tienen una capacidad de 50 Mbps. Cada salto corresponde a una distancia de 1000 km. ¿Qué tiempo se emplea en la transmisión del mensaje mediante datagramas?

Ejercicio 4.2.3. Suponga que una red de datagramas usa cabeceras de H bytes y que una red de paquetes de circuitos virtuales utiliza cabeceras de h bytes. Determine la longitud M de un mensaje que se consigue transmitir más rápido haciendo uso de la técnica de conmutación de circuitos virtuales que mediante la de datagramas. Suponga que los paquetes tienen la misma longitud en ambas redes y que los retardos de procesamiento son idénticos

Ejercicio 4.2.4. Una aplicación audiovisual en tiempo real hace uso de conmutación de paquetes para transmitir voz a 32 kbps y vídeo a 64 kbps a través de la conexión de red de la figura **??**. Se consideran paquetes de voz e información de audio con dos longitudes distintas: 10 ms y 100 ms. Cada paquete tiene además una cabecera de 40 octetos.

a. Encuentre para ambos casos el porcentaje de bits suplementarios que supone la cabecera.

b. Dibuje un diagrama temporal e identifique todas las componentes del retardo extremo a extremo en la conexión anterior. Recuerde que un paquete no puede ser transmitido hasta que esté completo y que no se puede retransmitir hasta que no se haya recibido completamente. Suponga despreciables los errores a nivel de bit.

c. Evalúe todas las componentes del retardo de las que se dispone suficiente información. Considere las dos longitudes de paquete aceptadas. Suponga que la señal se propaga a una velocidad de 1 km/5 microsegundos y considere dos velocidades para la red troncal: 45 Mbps y 1,5 Mbps. Resuma el resultado para los cuatro posibles casos en una tabla con cuatro entradas.

d. ¿Cuál de las componentes anteriores implica la existencia de retardos de cola?

Ejercicio 4.2.5. Imagine una situación donde hay cinco routers RA-RE. RA, RB y RC se conectan cada uno a una red local A, B y C, siendo cada router única puerta de enlace de cada red. RA, RB y RD están conectados entre sí a través de un switch. RC, RD y RE están conectados entre sí a través de un switch. RE conecta a Internet a través de la puerta de acceso especificada por el ISP. Especifique tablas de encaminamiento en los routers. Asigne a voluntad las direcciones IP e interfaces necesarias.

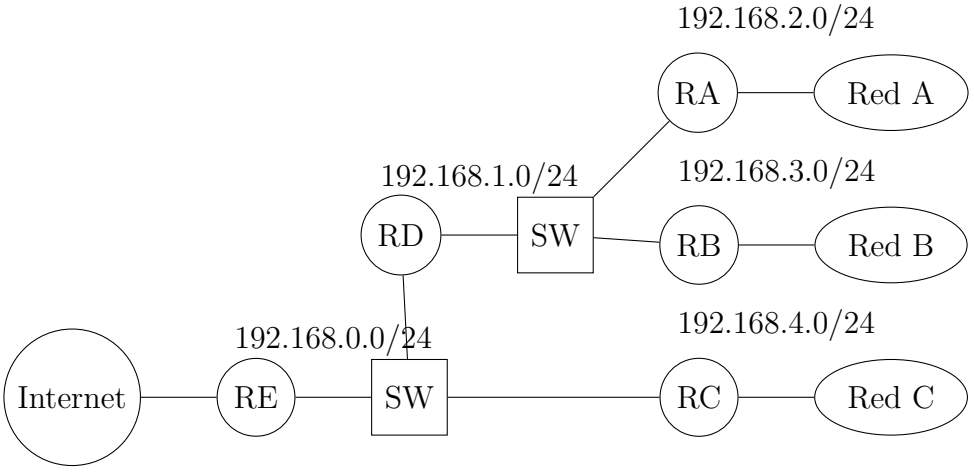


Figura 4.1: Situación del ejercicio 5

En primer lugar, asignaremos las direcciones IP y las interfaces de cada router. Cada router presenta una conexión por la izquierda y otra por la derecha, por lo que sólo usaremos dos interfaces de cada router, luego asociaremos dos direcciones IP a cada router.

Por comodidad, asociaremos las conexiones de la derecha de cada router a la interfaz *ether0*, y las conexiones de la izquierda de cada router a la interfaz *ether1*.

Una vez añadidas las interfaces, procederemos a asocciar direcciones IP a cada interfaz de cada router:

- RA:
  - Tendrá IP 192.168.2.1 en la red 192.168.2.0/24.

  - Tendrá IP 192.168.1.2 en la red 192.168.1.0/24.

- RB:
  - Tendrá IP 192.168.3.1 en la red 192.168.3.0/24.
  - Tendrá IP 192.168.1.3 en la red 192.168.1.0/24.
- RD:
  - Tendrá IP 192.168.1.1 en la red 192.168.1.0/24.
  - Tendrá IP 192.168.0.2 en la red 192.168.0.0/24.
- RC:
  - Tendrá IP 192.168.4.1 en la red 192.168.4.0/24.
  - Tendrá IP 192.168.0.3 en la red 192.168.0.0/24.
- RE:
  - Tendrá IP 192.168.0.1 en la red 192.168.0.0/24.
  - Su IP en la red que le conecta con Internet la proveerá el ISP.
- Suponemos que RE se conecta a Internet a través de un router con IP 33.33.33.33 en la red 33.33.0.0/16.

Procedemos ahora a rellenar las tablas de encaminamiento de cada router:

Red destino	Máscara	Siguiente salto	Interfaz
192.168.2.0	255.255.255.0	*	ether0
192.168.1.0	255.255.255.0	*	ether1
192.168.3.0	255.255.255.0	192.168.1.3 (RB)	ether1
0.0.0.0	0.0.0.0	192.168.1.1 (RD)	ether1

Tabla 4.1: Tabla de encaminamiento para RA.

Red destino	Máscara	Siguiente salto	Interfaz
192.168.3.0	255.255.255.0	*	ether0
192.168.1.0	255.255.255.0	*	ether1
192.168.2.0	255.255.255.0	192.168.1.2 (RA)	ether1
0.0.0.0	0.0.0.0	192.168.1.1 (RD)	ether1

Tabla 4.2: Tabla de encaminamiento para RB.

Red destino	Máscara	Siguiente salto	Interfaz
192.168.4.0	255.255.255.0	*	ether0
192.168.0.0	255.255.255.0	*	ether1
192.168.0.0	**255.255.252.0**	192.168.0.2 (RD)	ether1
0.0.0.0	0.0.0.0	192.168.0.1 (RE)	ether1

Tabla 4.3: Tabla de encaminamiento para RC.

Donde hemos agrupado la Red A (192.168.2.0/24), B (192.168.3.0/24) y la red 192.168.1.0/24 en la superred 192.168.0.0/22. Notemos que dentro de las direcciones de la superred se encuentran las direcciones de la forma 192.168.0.x, que no se encuentran en dicha superred. Sin embargo, tenemos una entrada específica para dichas direcciones, con una máscara de mayor prioridad (más 1s), por lo que no tenemos problema[3]

Red destino	Máscara	Siguiente salto	Interfaz
192.168.1.0	255.255.255.0	*	ether0
192.168.0.0	255.255.255.0	*	ether1
192.168.2.0	255.255.255.0	192.168.1.2 (RA)	ether0
192.168.3.0	255.255.255.0	192.168.1.3 (RB)	ether0
192.168.4.0	255.255.255.0	192.168.0.3 (RC)	ether1
0.0.0.0	0.0.0.0	192.168.0.1 (RE)	ether1

Tabla 4.4: Tabla de encaminamiento para RD.

Red destino	Máscara	Siguiente salto	Interfaz
192.168.0.0	255.255.255.0	*	ether0
33.33.0.0	255.255.0.0	*	ether1
192.168.4.0	255.255.255.0	192.168.0.3 (RC)	ether0
192.168.0.0	**255.255.252.0**	192.168.0.2 (RD)	ether0
0.0.0.0	0.0.0.0	33.33.33.33 (Router ISP)	ether1

Tabla 4.5: Tabla de encaminamiento para RE.

Donde hemos vuelto a usar la superred 192.168.0.0/22 que engloba a Red A, B y 192.168.1.0/24.

Ejercicio 4.2.6. Asigne las direcciones de subred en la siguiente topología a partir de 192.168.0.0 para minimizar el número de entradas en las tablas de encaminamiento, asumiendo que en las redes LAN puede haber hasta 50 PCs.

[3]Si no tuviéramos dicha entrada, tendríamos un problema, ya que si mandamos un paquete a 192.168.0.26, por ejemplo, iría a la superred que hemos definido pero algún router se daría cuenta de que no sabe llegar a 192.168.0.0/24.

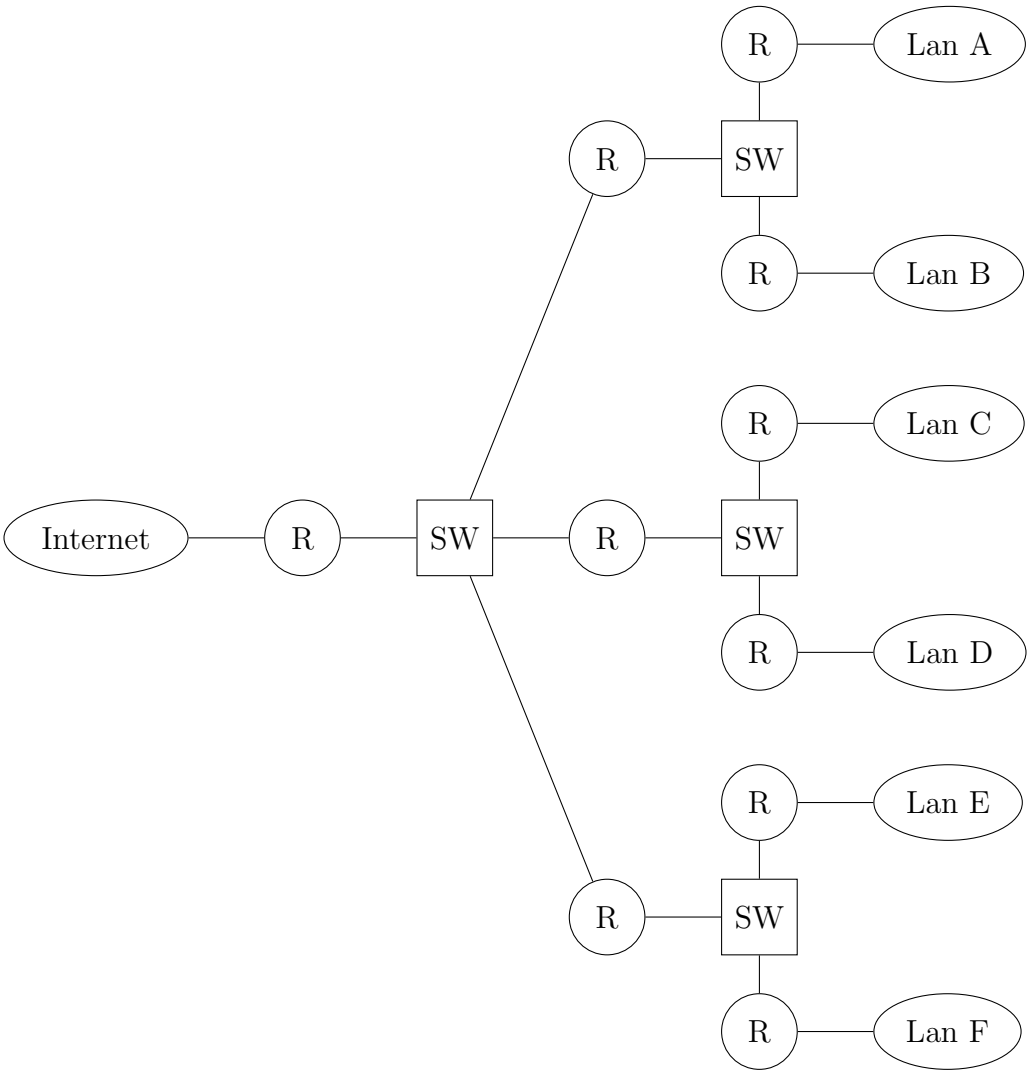


Figura 4.2: Situación del ejercicio 6

Ejercicio 4.2.7. Un datagrama de 4020 bytes pasa de una red Token Ring con THT 8 ms (MTU 4400) a una Ethernet (MTU 1500) y después pasa por un enlace PPP con bajo retardo (MTU 296). Si ese mismo datagrama pasara directamente de la red Token Ring al enlace PPP (sin pasa por la red Ethernet) ¿habría alguna diferencia en la forma como se produce la fragmentación? Especifique en ambos casos los fragmentos obtenidos.

Ejercicio 4.2.8. ¿Cómo podría utilizar ICMP para hacer una estimación de la latencia entre dos entidades finales? ¿Y para estimar la latencia de un enlace en particular entre dos routers?

Ejercicio 4.2.9. Considere la subred de la figura. Se utiliza el algoritmo de encaminamiento de vector distancia, habiéndose recibido en el encaminador C los siguientes vectores de encaminamiento: desde B (5, 0, 8, 12, 6, 2), desde D (16, 12, 6, 0, 9, 10) y desde E (7, 6, 3, 9, 0, 4). Los retardos medidos a B, D y E son, respectivamente, 6, 3 y 5. ¿Cuál es la nueva tabla de encaminamiento de C? Indique la línea de salida y el retardo esperado.

Ejercicio 4.2.10. Considere la red mostrada en la figura 4.3, en la que se representan 4 nodos unidos con enlaces. En cada enlace se indica el retardo sufrido por los mensajes al atravesarlo. Los nodos utilizan un protocolo de encaminamiento dinámico de tipo distribuido en el que la métrica está basada en el retardo. Se pide lo siguiente:

a. Escriba las tablas de encaminamiento para todos los nodos de la red una vez haya pasado el tiempo suficiente para que dichas tablas se construyan de forma estable.

b. Considere que los nodos envían y actualizan sus tablas cada 5 segundos, siendo la primera actualización en $t = 0$ s. Suponga que, en $t = 12$ s., el enlace BD pasa a tener un retardo de 3 s. ¿Cuál será el encaminamiento desde el nodo A hasta el nodo B cuando las tablas se estabilicen de nuevo?

c. ¿En qué instante comenzará dicho encaminamiento a funcionar? Justifique su respuesta explicando qué sucederá desde $t = 12$ s. hasta dicho instante y también después del mismo.

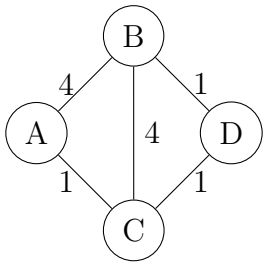


Figura 4.3: Grafo para el ejercicio 10.

Ejercicio 4.2.11. En la topología de red adjunta se indica la capacidad, en Kbps, de las líneas de transmisión entre los distintos nodos intermedios. Considérese al respecto que los enlaces son *full-duplex* y que la velocidad es la misma para cada uno de los sentidos. Por otra parte, la tabla anexa especifica el tráfico, en paquetes/segundo, entre cada par de nodos. Además, en cursiva se indica la ruta (secuencia de nodos) seguida en la transmisión. Teniendo en cuenta todo lo anterior, determine el retardo medio en el envío de un paquete sobre la red global.

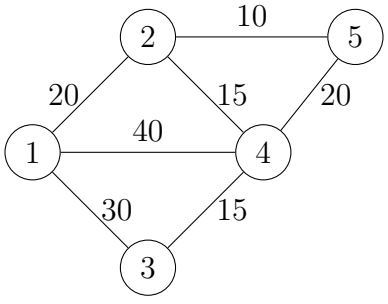


Figura 4.4: Grafo para el ejercicio 11.

		Nodo destino				
		1	2	3	4	5
Nodo origen	1		2-*12*	3-*13*	1-*14*	2-*145*
	2	2-*21*		4-*243*	2-*24*	2-*25*
	3	3-*31*	4-*342*		3-*34*	5-*345*
	4	1-*41*	2-*42*	3-*43*		1-*45*
	5	2-*541*	2-*52*	5-*543*	1-*54*	