ESCUELA POLITECNICA NACIONAL
FACULTAD DE INGENIERIA ELECTRICA

ESTUDIO Y ANALISIS DE LAS DISTINTAS TECNOLOGIAS DE ACCESO QUE UN PROVEEDOR DE SERVICIOS DE INTERNET PUEDE IMPLEMENTAR EN ECUADOR

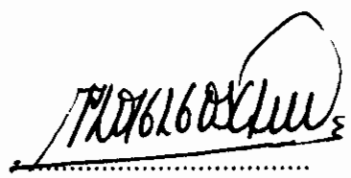
TOMO I

TESIS PREVIA A LA OBTENCION
DEL TITULO DE INGENIERO EN
ELECTRONICA Y TELECOMUNICACIONES

ALEXIS DIMITRI BARRETO MUÑOZ

QUITO, DICIEMBRE, 1999

Dejo constancia de que la presente tesis ha sido elaborada en su totalidad por el Sr. Alexis Barreto M.



Ing. Pablo Hidalgo
Director de Tesis

AGRADECIMIENTO

A mi familia por su apoyo siempre incondicional e infinita comprensión.

Al Ing. Pablo Hidalgo, por su acertada dirección en la elaboración de esta tesis.

A la Ing. Verónica Yerovi en la Superintendencia de Telecomunicaciones, por su enorme ayuda en la realización de mi tesis.

Un especial agradecimiento a Jaime Del Castillo y David Ramírez, de IBM del Ecuador, por su valiosa colaboración.

A todos ustedes, mis verdaderos amigos, gracias por su amistad sincera.

INDICE

TOMO I

TOMO II

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCION

Hoy en día, Internet se ha convertido en el sistema universal de telecomunicaciones, capaz de interconectar en forma transparente todo tipo de equipos mediante el empleo de redes basadas en distintas tecnologías y haciendo uso de diversos medios de transmisión.

Internet es una colección de redes interconectadas entre sí, sin topología definida. Está constituida por un gran *backbone,* formado por enlaces de gran ancho de banda y ruteadores de alta velocidad, al cual se conectan redes regionales, y a las cuales a su vez están conectadas redes WAN, redes LAN, proveedores de servicios, etc.

El Internet se ha difundido a nivel mundial y está cambiando día a día la forma en que las personas viven y se interrelacionan. Muchos lo usan pero muy pocos saben en realidad como funciona. La mayoría de los usuarios creen que la única forma de acceder a Internet es a través de la línea telefónica mediante un módem, sin embargo no saben que existen un sinnúmero de otras alternativas de acceso, las cuales incluso soportan de mejor forma los servicios y aplicaciones cada vez más complejos que se brindan a través de Internet.

Las aplicaciones con gran contenido de multimedia, aplicaciones en tiempo real, etc., que día a día surgen, requieren el uso de enlaces de alta velocidad para acceder a Internet. Sin embargo, los elevados costos de estos enlaces hacen que únicamente grandes empresas e instituciones puedan hacer uso de ellos, quedando fuera del alcance de las pequeñas y medianas empresas, y mucho menos para un usuario común, los cuales deben conformarse con utilizar enlaces de menor capacidad.

Las nuevas tecnologías de acceso a Internet existentes actualmente o que se están desarrollando permiten ofrecer al usuario altas velocidades de conexión, tratando de cubrir todos los sectores del mercado y a un costo que les permita competir.

El objetivo de este trabajo es dar a conocer estas nuevas tecnologías de acceso que el proveedor de servicio puede brindar al usuario para la conexión a Internet, analizando las ventajas y desventajas de cada una de ellas.

En el capítulo I se analiza detalladamente el modelo de referencia TCP/IP, sus capas y protocolos. El modelo de referencia TCP/IP constituye la base de Internet y por lo tanto este capítulo es clave para el entendimiento de los restantes.

En el capítulo II se revisan varios de los protocolos utilizados por un Proveedor de Servicios de Internet (ISP) para realizar tareas tales como autenticación de usuarios, tarifación, asignación de direcciones IP, etc. Además se detallan protocolos relacionados con PPP (*Point to Point Protocol*) tales como L2TP y MP-PPP.

En el capítulo III se describe la arquitectura de un ISP típico y se analiza la infraestructura necesaria para formar dicho ISP.

En el capítulo IV se presentan en forma detallada algunos de los métodos de acceso considerados más efectivos en la actualidad por su alta velocidad, cobertura y ventajas respecto a otros métodos, como son el acceso a Internet mediante la red telefónica utilizando tecnología xDSL, el acceso a través de la red de televisión por cable, el acceso inalámbrico entre otros, analizando las ventajas y desventajas de cada uno de ellos.

En el capitulo V se revisan las distintas tecnologías empleadas en el mercado ecuatoriano por los proveedores de servicio de Internet y se analiza la posibilidad o no de implementación de nuevas tecnologías de acceso.

En el capítulo VI se presentan las conclusiones y recomendaciones que se desprenden de este trabajo.

En los Anexos se ofrece valiosa información que complementa aspectos no cubiertos en esta tesis, tales como una comparación entre el modelo de referencia OSI y el modelo

TCP/IP, introducción a ciertas tecnologías tales como ATM y Frame Relay, técnicas de modulación usadas en tecnologías xDSL, etc. Además se presentan algunos criterios para el diseño y dimensionamiento de un ISP.

INDICE - CAPITULO I

CAPITULO I

PROTOCOLOS DEL MODELO DE REFERENCIA TCP/IP

El Internet es un conjunto de redes de computadoras interconectadas entre sí, una red de redes. En sus inicios esta gran red fue concebida por la Agencia de Proyectos de Investigación Avanzada (ARPA - *Advanced Research Projects Agency*) del gobierno de Estados Unidos en 1969. [1] Inicialmente se la denominó ARPANET. El propósito original de sus creadores fue establecer una red que permitiese a científicos universitarios dedicados a la investigación de la computación ser capaces de intercambiar sus trabajos con investigadores en otras universidades. El principal objetivo en que se basó el diseño original fue el de permitir que los mensajes pudiesen ser enrutados o reenrutados en más de una dirección, de tal forma que la red pudiese continuar funcionando aun si partes de ésta fuesen destruidas en el caso de un ataque militar u otro desastre.

Desde aquellos días hasta hoy, la red de redes ha estado y continuará en constante evolución y expansión. Hoy en día, el Internet es un red pública, cooperativa y que provee servicios y aplicaciones que son utilizadas por cientos de miles de personas en todo el mundo. Así, el correo electrónico (*e-mail*) prácticamente ha reemplazado el servicio postal, llegándose a convertir en una de las aplicaciones más utilizadas en la red. El IRC (*Internet Relay Chat*), permite a un usuario mantener conversaciones con otros usuarios de la red. Mas recientemente, la telefonía a través de Internet y otros paquetes de *software* que permiten comunicación en tiempo real a través de la red están cambiando la forma en que se interrelacionan las personas. La aplicación más difundida del Internet, el WWW (*World Wide Web*), permite al usuario tener acceso a millones de páginas de la más variada y actualizada información en forma inmediata e incluso realizar negocios a través de la red. Estas y muchas otras aplicaciones han hecho que el Internet pase a formar parte, en ciertos casos indispensable, de nuestra vida cotidiana.

Técnicamente, el Internet está formado por una colección de distintas redes físicas interconectadas entre sí, muchas de las cuales utilizan distintas tecnologías. Ante el usuario, Internet aparenta ser una única gran red virtual, ya que el *software* de red, basado en un conjunto de protocolos conocidos como TCP/IP, se encarga de ocultar las diversas tecnologías de red subyacentes. Esta gran red virtual puede imaginarse formada por varios *hosts* que ejecutan aplicaciones, los cuales se conectan a una subred de comunicaciones, la misma que se encarga de intercambiar información entre ellos.

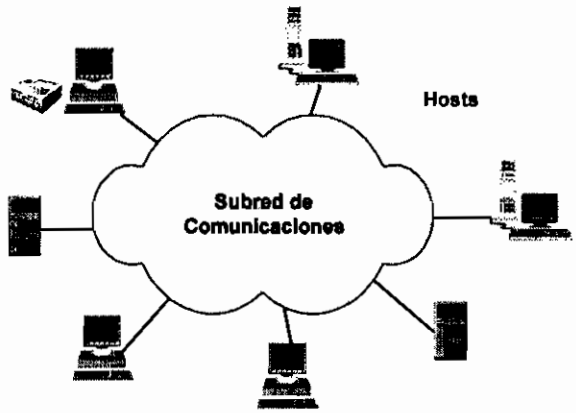


Fig. 1.1. *Hosts* y Subred de comunicaciones

La subred de comunicaciones en Internet utiliza la conmutación de paquetes como la base de su arquitectura. La **conmutación de paquetes** consiste en la división de los datos a transmitirse en pequeños fragmentos denominados paquetes, los cuales se los conmuta individualmente a través de diferentes caminos hacia su destino mediante dispositivos denominados ruteadores o nodos de conmutación. Para ello, se marca cada paquete con su dirección de destino, de forma que si un nodo de conmutación es incapaz de tratar los paquetes que le llegan, un enlace falla, etc., el resto de la red será capaz de enviar ese paquete a su destino a través de otro camino.

Una de las ventajas de la conmutación de paquetes es que permite utilizar una misma conexión para enviar paquetes pertenecientes a múltiples comunicaciones entre distintas máquinas en forma simultánea, es decir permite multiplexar en una misma conexión varias comunicaciones. La desventaja de la conmutación de paquetes es la falta de una capacidad garantizada. Esto se debe a que cuando se incrementa el número de comunicaciones simultáneas que hacen uso de una misma conexión, la capacidad para

cada una de ellas disminuye, pudiendo llegarse a dar problemas de sobrecarga de la red.

Existe otra tecnología totalmente opuesta a la conmutación de paquetes, conocida como conmutación de circuitos, en la que se basan algunos tipos de redes físicas, las cuales pueden estar formando parte de Internet. Un ejemplo típico de red que usa conmutación de circuitos es la red telefónica pública, la cual es un elemento fundamental en el acceso a Internet de muchos usuarios. La **conmutación de circuitos** implica el establecimiento, uso y liberación de una conexión física entre los puntos de origen y destino denominada circuito. Mientras este circuito se mantiene, el emisor podrá transmitir sus datos hacia el receptor. La ventaja de la conmutación de circuitos reside en la capacidad de transmisión fija garantizada que ofrece el circuito, es decir una vez establecido el circuito ninguna otra actividad de la red disminuirá dicha capacidad. El costo por circuito es fijo, independiente de si se cursa o no tráfico en determinado instante de la comunicación, lo cual resulta en una desventaja cuando el tráfico no es continuo ya que se desperdicia ancho de banda. Otra desventaja es que si la conexión falla, por algún motivo, la información se perderá.

Como se verá posteriormente el *software* de red TCP/IP de Internet está diseñado para ser independiente del tipo de red física subyacente y se encargará de enviar los paquetes a través de redes que posiblemente usen conmutación de circuitos u otras distintas tecnologías aparentemente incompatibles con la idea de conmutación de paquetes.

En general existen dos tipos de servicios que se pueden implementar en una red de computadores: [1]

- Servicios orientados a conexión
- Servicios no orientados a conexión

Un servicio se dice **orientado a conexión** cuando es necesario establecer un enlace entre origen y destino al inicio de la comunicación, mantenerlo durante la comunicación y liberarlo al finalizarla. La transmisión de datos mediante conmutación de circuitos es un ejemplo de servicio orientado a conexión.

Un servicio se dice **no orientado a conexión** cuando no es necesario establecer ni liberar un enlace al inicio y fin de la comunicación. Los datos utilizan dinámicamente distintos enlaces en su trayecto entre origen y destino. La transmisión de datos mediante datagramas proporciona un servicio no orientado a conexión.

1.1 MODELO DE REFERENCIA TCP/IP

Una red de computadores, como Internet, se la analiza en base a un modelo estratificado en capas, una sobre otra, donde cada capa ofrece servicios a capas superiores, ocultando a la capa superior los detalles de implementación de los mismos.

Un modelo de estratificación de capas se basa en la comunicación virtual entre capas correspondientes en máquinas distintas, mediante un conjunto de reglas y procedimientos denominados protocolos.[1] A los procesos que se ejecutan en capas correspondientes en máquinas diferentes se denominan procesos pares. Son los procesos pares los que se comunican mediante un protocolo.

La función de cada capa es la de ofrecer determinados servicios a las capas superiores. Entre cada par de capas existen las denominadas interfaces, las cuales definen las operaciones primitivas y servicios que la capa inferior ofrece a la capa superior. Es decir cada capa se apoya en los servicios que le ofrece la capa inferior y ofrece sus servicios a la capa superior a través de las interfaces.

En la comunicación real, cada capa añade información de control en forma de cabeceras y *trailers* a la información enviada por la capa superior, para permitir la comunicación virtual entre procesos pares. En el destino, las unidades de información pasan de una capa inferior a una superior. Cada capa solo interpreta la información añadida y enviada por la capa correspondiente en el origen.

Una de las ventajas de un modelo estratificado en capas es el hecho de que permite sustituir la forma de realizar un servicio en una capa sin necesidad de alterar el resto de los servicios en otras capas. Esto es posible, ya que los detalles de implementación de un servicio dentro de una capa no interesan a la capa superior.

Internet se fundamenta en el modelo de referencia TCP/IP el cual en forma general define tres tipos de servicios relacionados entre sí. En el nivel inferior ofrece un servicio no confiable de entrega de paquetes sin conexión, sobre el cual se basa toda la arquitectura y que permite que el conjunto TCP/IP sea adaptable a un número importante de topologías de red. Sobre este nivel ofrece el servicio de transporte extremo a extremo, que es la base para los servicios de aplicación.

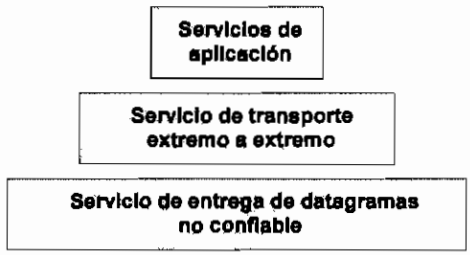


Fig. 1.2. Tipos de servicios definidos en el modelo TCP/IP [2]

El conjunto de protocolos TCP/IP es una combinación de protocolos estructurados en cuatro capas, denominadas: Aplicación, Transporte, Internet e Interfaz de red. [2] A continuación se presenta un esquema de la estratificación de capas en Internet. Es necesario notar que únicamente los *hosts* utilizan en su comunicación las cuatro capas del modelo TCP/IP, un ruteador en la subred de comunicaciones únicamente requiere de las dos capas inferiores como se verá posteriormente.

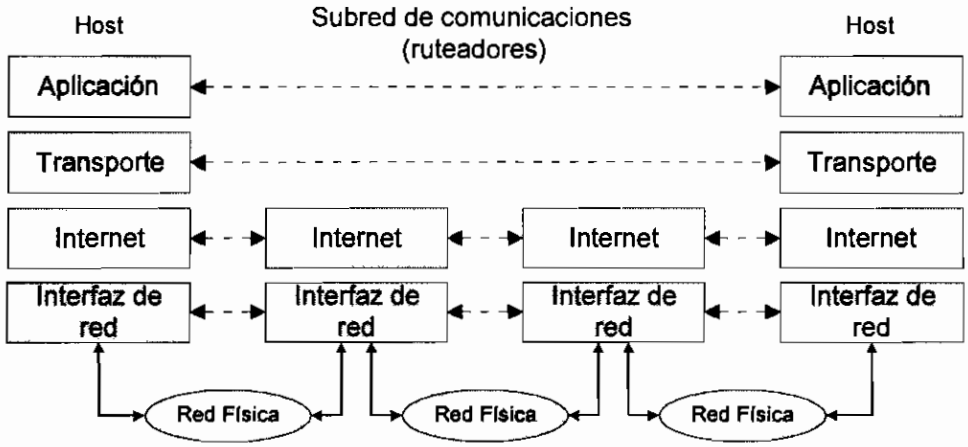


Fig. 1.3. Modelo de estratificación de capas TCP/IP

1.2 CAPAS DEL MODELO TCP/IP

1.2.1 Capa Aplicación

En esta capa se encuentran los protocolos de alto nivel o aplicaciones. Algunas de las aplicaciones mas usadas son: TELNET (Acceso remoto), FTP (Transferencia de Archivos), DNS (Nombres de dominio), SMTP (Correo Electrónico), NNTP (Grupos de noticias), HTTP (Transferencia de documentos de Hipertexto) base del WWW (*World Wide Web*), etc.

La aplicación interactúa con los protocolos de la capa transporte a fin de que esta capa se encargue de enviar y recibir datos en forma de mensajes o flujos continuos de octetos requeridos según la aplicación. Dependiendo de la aplicación se seleccionará un tipo de transporte.

1.2.2 Capa Transporte

La capa transporte se encarga de establecer la comunicación extremo a extremo entre procesos pares de capa aplicación en *hosts* diferentes. En esta capa se definen dos tipos de protocolos: TCP (*Transmission Control Protocol*) que es un protocolo orientado a conexión y UDP (*User Datagram Protocol*) que es un protocolo no orientado a conexión.

El protocolo TCP es un protocolo de transporte orientado a conexión que asegura que los datos lleguen a su destino sin errores y en una secuencia correcta. TCP proporciona una comunicación eficiente, confiable y en términos de costos efectiva ya que permite multiplexar varias conexiones de transporte utilizadas por varios procesos de capa aplicación que se ejecutan simultáneamente en una misma máquina. El protocolo TCP divide los datos de los programas de aplicación en pequeños paquetes, denominados segmentos, que son pasados a la capa internet para que se encargue de su entrega al destino.

Por otro lado, el protocolo UDP es un protocolo de transporte no orientado a conexión, no confiable. UDP envía los datos de la aplicación en paquetes, llamados datagramas de usuario, de un *host* a otro sin garantizar la correcta recepción de los mismos. Por esta razón, la capa de aplicación deberá encargarse de los detalles correspondientes para obtener las garantías de transmisión. La utilización de TCP o de UDP depende del tipo de aplicación y del tipo de transporte que la aplicación requiera: orientado a conexión o no orientado a conexión.

1.2.3 Capa Internet

La capa internet o capa red es la responsable del movimiento de los paquetes procedentes de la capa transporte de una máquina a otra a través de la red. Esta capa provee una imagen de red virtual a Internet, esto es, se encarga de aislar a las capas superiores de la tecnología de red utilizada debajo de ellas haciendo que, para el usuario, aparente funcionar como una red única.

En esta capa se define el protocolo IP (*Internet Protocol*). El protocolo IP es un protocolo no orientado a conexión. IP no provee confiabilidad, control de flujo o control de errores. Estas funciones deben ser provistas en un nivel superior sea en capa transporte usando TCP como el protocolo de transporte, o en capa aplicación si UDP es utilizado como protocolo de transporte

La capa internet define el datagrama IP como la unidad básica de transferencia de datos a través de una red TCP/IP. Su función es la de encapsular el paquete de capa transporte en un datagrama IP, añadiéndole una cabecera que contiene las direcciones de red, conocidas como direcciones IP, de las máquinas origen y destino. Además, esta capa se encarga del encaminamiento (ruteo) de los datagramas.

El servicio de entrega de datagramas sin conexión, no confiable y con el mejor esfuerzo es la base del modelo TCP/IP. Se dice que es un servicio no confiable ya que no se garantiza la entrega de los datagramas, es sin conexión por que cada datagrama es tratado en forma independiente de los demás y es con el mejor esfuerzo ya que el *software* de red hace su mejor intento por entregar los datagramas al destino.

Además en esta capa se encuentran los protocolos ICMP (*Internet Control Message Protocol*) e IGMP (*Internet Group Management Protocol*). [A] Las capas internet de *hosts* o ruteadores utilizan el protocolo ICMP para intercambiarse mensajes de error y control. En una situación de funcionamiento temporal incorrecto, por ejemplo en un ruteador, la capa internet desecha el datagrama y envía un mensaje al *host* emisor mediante ICMP. También existen algunos servicios de aplicación, como las aplicaciones *Ping* y *Traceroute*, que se basan directamente en el protocolo ICMP. El protocolo IGMP permite a un *host* informar a un ruteador local que desea recibir paquetes *multicast*.[B] Un *host* puede unirse a uno o más grupos de *multicast* y recibir paquetes *multicast*. Los ruteadores que reciben paquetes de *multicast* utilizan IGMP para averiguar si existen *hosts*, en la red a la que el ruteador está conectado, que pertenezcan al grupo al cual va direccionado el paquete. Si existen *hosts* pertenecientes al grupo envía el paquete, caso contrario lo desecha.

1.2.4 Capa Interfaz de Red

También denominada capa de enlace o capa *host*-red, es la encargada de establecer la verdadera interfaz con el *hardware* de red. El modelo de referencia TCP/IP no define ningún protocolo específico para esta capa, ni características del medio físico de transmisión sobre el cual deba implementarse esta arquitectura. Unicamente puntualiza que esta capa debe aceptar los datagramas IP, unidad de transferencia de la capa internet, y transmitirlos sobre alguna red física, que utilice algún protocolo y medio de transmisión tal que le permita enviar datagramas IP sobre él.

Esta libertad en elección del protocolo y medio de transmisión tiene una gran ventaja ya que permite comunicarse entre sí a todo tipo de equipos (superordenadores, ordenadores personales, impresoras, etc.) en forma transparente mediante el empleo de redes (LAN, MAN, WAN) que utilizan distintas tecnologías (Ethernet, Token Ring, ATM, Frame Relay, FDDI, red telefónica, ISDN, X.25, etc.)[3] utilizando distintos medios de transmisión (cables de cobre, cable coaxial, fibra óptica, ondas de radio, satélites).

[A] ICMP e IGMP en verdad se consideran como parte integral de IP y no protocolos independientes.
[B] El *Multicasting* es una transmisión simultánea de uno a varios.

El *host* utiliza una tarjeta de interfaz de red dentro del ordenador y su correspondiente controlador dentro del sistema operativo dependiendo del tipo de red física a la que esté conectado.

Esta capa se encarga de transmitir los datagramas IP entregados por la capa internet sobre una red específica encapsulándolos dentro de tramas. La trama es la unidad de intercambio de información a nivel de capa interfaz de red. Una trama encapsula el datagrama IP añadiéndole generalmente una cabecera al inicio y un *trailer* al final. El formato exacto de una trama depende del tipo de red física empleada. A nivel de capa interfaz de red, se utilizan las direcciones físicas de las tarjetas de interfaz de red para enviar y recibir tramas. Esta capa no entiende las direcciones de red utilizadas por la capa internet, por lo cual antes de enviar un datagrama dentro de una trama, la capa interfaz de red debe encargarse de transformar la dirección IP de destino del datagrama en una dirección física de destino de una interfaz de red.

En esta capa se emplean dos protocolos de bajo nivel: ARP (*Address Resolution Protocol*) y RARP (*Reverse Address Resolution Protocol*), útiles solo en ciertos tipos de redes que soporten difusión (Ejm. Ethernet y Token Ring), que se encargan de realizar la conversión entre direcciones IP y direcciones físicas. Además se utilizan otros protocolos tales como PPP (*Point to Point Protocol*) o SLIP (*Serial Line IP*) en caso de que se utilicen enlaces seriales como medios físicos de transmisión.

A continuación se examinará con mayor detalle los protocolos existentes en cada una de las capas y la forma en que permiten implementar los servicios mencionados en cada una de ellas. La información completa que describe un protocolo, un estándar, una aplicación, o aspecto relacionado con Internet está publicada en documentos conocidos como *Request For Comments* (RFCs). Cada RFC posee un número y cada uno trata sobre un determinado aspecto de Internet. Estos documentos son accesibles a través de la red, ya que son actualizados permanentemente y si fuesen publicados en otro medio podrían quedar obsoletos muy pronto. [C]

[C] http://www.cis.ohio-state.edu/htbin/rfc/rfc*XXXX*.html, es una dirección URL donde se mantiene una copia de cada RFC existente. XXXX es el número del RFC que se desee revisar. Ejm. 0768, 1332, etc.

1.3 PROTOCOLOS DE LA CAPA INTERFAZ DE RED

1.3.1 *Address Resolution Protocol* (ARP)

En una misma red física los *hosts* individuales son identificados por medio de direcciones físicas o de *hardware*. Sin embargo, los protocolos de alto nivel utilizan direcciones simbólicas, en el caso del protocolo IP, direcciones IP, para identificar a un determinado *host* en la red. Cuando un *host* desea enviar un datagrama a otro *host* que posee una dirección IP de destino dada, el *driver* de la tarjeta de red del *host* origen no entiende la dirección IP del destino, pues éste utiliza direcciones físicas y no direcciones de red, y no sabría a quien enviar.

El Protocolo de Resolución de Direcciones (ARP - *Address Resolution Protocol*), se encarga de traducir la dirección IP del *host* de destino en su dirección física. ARP utiliza una tabla de búsqueda para realizar esta traducción. La tabla ARP contiene la dirección física correspondiente a cada dirección IP que el *host* conoce. Cuando el *host* debe comunicarse con un destino del cual no tiene información en su tabla ARP, la máquina origen envía una trama cuyo formato depende del tipo de red física, conteniendo en su campo de datos un paquete de pedido ARP. Esta trama es enviada utilizando como dirección física de destino, la dirección física de difusión que la red, en la que se encuentre conectado el *host*, utilice. Mediante la difusión (*broadcast*) se asegura que todos los *hosts* conectados a la red física recibirán la trama conteniendo el pedido ARP. Si una de las máquinas reconoce su dirección IP, especificada en el pedido, enviará una respuesta ARP de regreso al *host* que hizo la solicitud en forma *unicast*, es decir, directamente sin utilizar difusión.

La respuesta contiene la dirección física del *host* de destino que corresponde a su dirección IP. Ambas direcciones son almacenadas en la tabla ARP del *host* que solicitó el pedido. Todos los subsecuentes datagramas enviados a esta dirección IP de destino podrán ser traducidos a su respectiva dirección física al realizar la búsqueda en la tabla ARP. ARP fue diseñado para ser utilizado en redes que soporten difusión por medio de *hardware*, como por ejemplo, en redes LAN 802.3, 802.4, 802.5, etc. ARP no funciona en una red X.25 o ATM las cuales no soportan difusión.

Todos los *host* de la red, que reciben el pedido ARP a través de la difusión, actualizan las direcciones de *hardware* y de red de la máquina que envía un pedido de ARP, ya que un pedido ARP además de la dirección IP del destino buscado, contiene la dirección IP y la dirección física de la máquina que realizó el pedido ARP. Esto permite optimizar el protocolo y mantener actualizadas las tablas ARP ante cambios en el *hardware* de red.

Cuando a un *host* se le ha cambiado su tarjeta de interfaz de red, o la dirección de red asociada a la interfaz de red, envía durante su inicialización un paquete de difusión ARP para actualizar las tablas de los demas *hosts* en la red.

El formato del paquete ARP es el siguiente:

Fig. 1.4. Formato de un paquete ARP [2]

- **Tipo de *Hardware*:** Especifica el tipo de *hardware* de la red física subyacente a la que está conectada el *host*.
- **Tipo de Protocolo:** Especifica el tipo de protocolo de red al cual corresponde la dirección de red usada por los *hosts*. El código hexadecimal para el protocolo IP es 0800. [2]
- **HLEN:** Longitud, medida en octetos, del tipo de direcciones físicas utilizadas.
- **PLEN:** Longitud, medida en octetos, del tipo de direcciones de protocolo de red utilizado. En el caso de IP, las direcciones de red tiene una longitud de cuatro octetos.
- **Operación:** Especifica si la operación es una petición ARP (1) o una respuesta ARP (2).

- **Dirección *Hardware* Emisor:** Dirección física del emisor.
- **Dirección IP Emisor:** Dirección de red del emisor. En el caso de IP, dirección IP.
- **Dirección *Hardware* Destino:** Dirección física del destino.
- **Dirección IP Destino:** Dirección de red del destino. En el caso de IP, dirección IP.

El paquete de respuesta intercambia la posición de las direcciones de emior y destino.

1.3.2 *Proxy* ARP

El protocolo ARP únicamente es aplicable en redes que soporten difusión. En el caso en que un *host* desee enviar datagramas a un *host* ubicado en una red distinta deberá enviar sus datagramas a través de un ruteador. Sin embargo, al intentar el *host* de origen averiguar la dirección física del destino mediante un pedido ARP, éste fallará ya que un ruteador no soporta difusión a nivel de capa interfaz de red.

Existen dos formas de solucionar el problema. La más simple es que el *host* examine la dirección IP de destino contenida en el datagrama. Si pertenece a su red local puede utilizar ARP para determinar la dirección física de destino. En caso de que el datagrama vaya direccionado a una red que no sea la local, deberá ser enviado dentro de una trama a la dirección física de un ruteador por defecto (configurado en el *host*) que maneje el tráfico remoto, el cual se encargará de buscar la dirección física de destino asociada a la dirección IP en la siguiente red mediante ARP o sino pasar el datagrama al siguiente ruteador por defecto, encapsulándolo en el tipo de trama que se utilice en esa red y así sucesivamente hasta finalmente llegar al destino. El último ruteador, aquel que está directamente conectado a la red a la cual el *host* de destino pertenece, utilizará ARP para transformar la dirección IP en la dirección física del *host* de destino.

Otra solución es la denominada técnica *proxy* ARP, en la cual un ruteador corre *software proxy* ARP, que le permite capturar solicitudes de difusión ARP de una red a la cual está conectado, analizar si el destino no se encuentra en la red local y en ese caso, enviar como respuesta al pedido ARP su propia dirección física, la cual será utilizada por la máquina origen como destino de sus tramas. Se denomina *proxy* ARP o sustituto ARP ya que el ruteador simula ser el destino. El ruteador al recibir el

datagrama lo enviará contenido en una trama al siguiente salto hacia el destino. El último ruteador, conectado a la red a la cual pertenece el *host* de destino, utilizará ARP para transformar la dirección IP en la dirección física del *host* de destino.

1.3.3 *Reverse Address Resolution Protocol* (RARP)

Algunos *hosts* en una red, tales como terminales que no poseen discos de almacenamiento, no conocen su dirección IP cuando arrancan. Para determinar su dirección IP utilizan el Protocolo de Resolución Reversa de Direcciones (RARP - *Reverse Address Resolution Protocol*). RARP utiliza un mecanismo similar a ARP, pero en el cual la dirección de *hardware* es un parámetro conocido, y la dirección IP es el parámetro que se busca. RARP difiere de ARP en el sentido de que en la red en la que se encuentre conectado el *host* debe siempre existir un servidor RARP, el cual debe tener una base de datos que asocie direcciones de *hardware* con direcciones IP y que se encargue de dar respuesta a los pedidos RARP de los terminales.

Cada sistema de una red tiene una única dirección física, asignada por el fabricante. El principio de RARP es que una estación sin disco lea la dirección física (única) de su propia tarjeta de interfaz de red y envíe una petición RARP dentro de una trama que utilice como dirección física de destino la dirección física de difusión (*broadcast*). El servidor RARP responderá al pedido enviando la dirección IP asociada a dicho sistema sin disco, contenida en un paquete de respuesta RARP, el cual viaja encapsulado dentro de una trama que utiliza como dirección física de destino la dirección física del *host* de origen. Esta trama de respuesta consecuentemente viaja en forma *unicast*. RARP utiliza el mismo formato de mensaje que ARP descrito anteriormente, con la excepción que el campo Operación lleva los valores: (3) para indicar un pedido RARP y (4) para una respuesta RARP.

1.3.4 *Serial Line IP* (SLIP)

La familia de protocolos TCP/IP corre sobre una variedad de redes entre ellas: LANs IEEE 802.3, 802.4, 802.5, X.25, enlaces satelitales, enlaces seriales, etc. Se han definido estándares para la encapsulación de datagramas para muchas de estas redes, pero no

existe ningún estándar para líneas seriales, a pesar de que los circuitos punto a punto asincrónicos y sincrónicos son muy utilizados en la comunicación de datos.[4] El protocolo de Línea Serial IP (SLIP - *Serial Line Internet Protocol*) es actualmente el estándar de facto, comúnmente utilizado para conexiones seriales punto a punto asincrónicas sobre las cuales corre TCP/IP. SLIP no es un estándar de Internet.

SLIP como su nombre lo indica permite que un ordenador utilice el protocolo IP sobre un enlace serial asincrónico, como puede ser por ejemplo en el acceso a Internet a través de la línea telefónica *dial up* o un enlace WAN entre dos ruteadores. Cuando un ordenador se encuentra conectado a un proveedor de servicios de Internet que emplee SLIP, puede enviar y recibir paquetes IP como si fuese cualquier otro *host* directamente conectado a Internet.

SLIP es un protocolo muy simple diseñado hace mucho tiempo atrás y es únicamente un protocolo delimitador de paquetes. Define una secuencia de caracteres que delimitan paquetes en una línea serial y nada más.

El *host* que utiliza SLIP envía paquetes IP sobre la línea serial utilizando la técnica conocida como "bandera y relleno de caracteres" para delimitar las tramas. Esta técnica en el caso de SLIP consiste en utilizar, como bandera delimitadora de trama, un byte conteniendo el valor hexadecimal 0xC0 al final de la trama.[D] Debido a que es posible que el caracter utilizado como delimitador de trama pueda ocurrir dentro del datagrama IP que se va a encapsular en la trama, es necesario prevenir esto para lograr transparencia. Si se da este caso, el caracter 0xC0 dentro del datagrama, es reemplazado por la secuencia de 2 bytes: 0xDB, 0xDC. De igual forma, si el caracter 0xDB aparece dentro del datagrama IP, a su vez es reemplazado por la secuencia de 2 bytes: 0xDB, 0xDD. En el destino todos los caracteres de transparencia son removidos y reemplazados por los verdaderos.[4]

[D] Existen varias formas de notación hexadecimal. Ejm. 0xC0 = C0 = $C0_{16}$ = $C0_{h}$, que equivalen a 1100 0000 en binario.

1.3.5 *Point-to-Point Protocol* (PPP)

SLIP ha sido el protocolo estándar de facto para conexiones seriales punto a punto y su uso está aún muy difundido en las conexiones TCP/IP mediante línea telefónica (*dial up*), sin embargo, SLIP tiene un sinnúmero de inconvenientes:

- SLIP define sólo un protocolo de encapsulación, ninguna forma de *handshake* o control de enlace. Los enlaces se deben conectar y configurar manualmente, incluyendo la especificación de la dirección IP de las máquinas.
- SLIP esta únicamente definido para enlaces asincrónicos.
- SLIP no puede soportar múltiples protocolos a través de un único enlace, todos los paquetes deben ser del mismo tipo (Ejm. datagramas IP).
- SLIP no hace ningún control de detección de errores, lo cual forza para que la retransmisión sea manejada por protocolos de mayor nivel, en el caso de que se den errores en líneas ruidosas.
- SLIP no provee de ninguna forma de autenticación, por lo cual ninguno de los extremos sabe con quien mismo se está comunicando. En el caso de líneas dedicadas no es ningún problema, pero resulta serlo en el caso de una comunicación a través de líneas conmutadas.
- SLIP no provee un mecanismo de compresión de cabeceras en datagramas IP, lo cual para algunas aplicaciones interactivas (Ejm. Telnet) provoca una baja eficiencia en el uso del enlace, debido a que el tamaño de los datos emitidos resulta ser más pequeño en comparación al tamaño de las cabeceras que añaden los protocolos TCP e IP.

Algunas de las versiones actuales de SLIP ya utilizan una técnica de compresión de cabeceras, denominada compresión de cabeceras de Van Jacobson, descrita en el RFC 1144. Este tipo de compresión permite reducir el tamaño de la combinación de cabeceras IP y TCP de 40 bytes a unos 3 bytes en promedio,[5] tomando ventaja del hecho que datagramas consecutivos a menudo tienen muchos de los campos de las cabeceras en común. Para lograr la compresión se omiten aquellos campos que no han cambiado respecto al datagrama anteriormente enviado. Además, aquellos campos que varían respecto a los del paquete anterior no son enviados por completo, sino

únicamente como incrementos al valor previo.

El Protocolo Punto a Punto (PPP - *Point to Point Protocol*) soluciona las deficiencias de SLIP. PPP define tres aspectos principales: [6]

1. Un método para encapsular datagramas sobre líneas seriales, el cual delimita el inicio y fin de cada trama. El formato de la trama permite manejar detección de errores.
2. Un Protocolo de Control de Enlace (LCP - *Link Control Protocol*) para: establecer, configurar, probar, mantener y terminar la conexión del enlace de datos.
3. Una familia de Protocolos de Control de Red (NCPs - *Network Control Protocols*) para: establecer y configurar diferentes protocolos y opciones de capa red. PPP está diseñado para permitir el uso simultáneo de múltiples protocolos de capa red.

PPP puede implementarse con cualquier tipo de interfaz serial DTE/DCE (Ejm. EIA RS 232-C, EIA RS-422-A, EIA RS-423-A, V.35, etc.).[7] El único requerimiento impuesto por PPP es que debe implementarse sobre un circuito *full duplex,* el cual puede ser dedicado o conmutado y a su vez puede operar en forma asincrónica o en modo de transferencia de bits serial sincrónica. PPP no impone restricciones respecto a la velocidad de transmisión impuesta por una interfaz DTE/DCE particular que se utilice, ni tampoco requiere de ninguna técnica de codificación particular para transmisiones sincrónicas.

Un acceso a Internet utilizando módem analógico sobre línea telefónica *dial up* o dedicada y una conexión entre dos ruteadores son ejemplos típicos de enlaces seriales, en los cuales se requiere utilizar uno de los protocolos SLIP o PPP para encapsular los paquetes IP. PPP en la actualidad es el protocolo más utilizado por las ventajas que presenta sobre SLIP.

En el caso de un acceso a Internet mediante módem analógico antes de que un enlace se considere listo para su uso por los protocolos de capa red, una secuencia específica de eventos deben darse:

1. **Establecimiento de la conexión física:** Primeramente se establece la conexión física entre el módem de la PC del usuario y el módem del proveedor de servicios de Internet. El módem del usuario debe realizar la llamada telefónica al módem del proveedor de servicio. Cuando este último contesta se establece la conexión física.
2. **Establecimiento del enlace y negociación de configuración:** A continuación se intercambian una serie de paquetes LCP en el campo de datos de la trama PPP, los cuales negocian opciones de la configuración del enlace. Una vez que se llega a un acuerdo en las opciones, el enlace se abre, pero aún no se lo habilita para que los protocolos de capa red lo utilicen.
3. **Determinación de la calidad del enlace:** Esta fase es opcional. PPP no especifica las políticas para determinar calidad del enlace, pero provee herramientas de bajo nivel, tal como petición de eco y respuesta al eco, para probarlo y ver si el destino está accesible.
4. **Autenticación:** Esta fase es opcional. Cada extremo del enlace se identifica con el extremo remoto, utilizando un protocolo de autenticación acordado durante la fase de configuración del enlace mediante LCP. Si la autenticación falla, el enlace se termina inmediatamente.
5. **Negociación de la configuración del protocolo de capa red:** Una vez que LCP ha terminado la fase anterior, los protocolos de capa red pueden ser separadamente configurados mediante la transmisión de una serie de paquetes del NCP apropiado, según el protocolo de red que se vaya a utilizar.

   El Protocolo de Control IP (IPCP - *IP Control Protocol*) es el NCP para el caso en que IP vaya a ser utilizado como protocolo de capa red. IPCP es el responsable de: habilitar, configurar y deshabilitar el protocolo IP en ambos extremos de un enlace punto a punto.

   Durante esta fase, el proveedor de servicios de Internet (ISP) asigna dinámicamente al PC del usuario una dirección IP, de las que tiene disponibles, para que éste pueda correr los protocolos TCP/IP. Una vez asignada la dirección IP al usuario, éste puede enviar y recibir datagramas IP como si fuese una máquina permanentemente conectada a Internet.

IPCP también asigna al usuario las direcciones IP de uno o varios servidores de nombres (DNS) que deberá utilizar en la red del ISP.

IPCP permite configurar opcionalmente, el uso o no de la compresión de cabeceras TCP/IP de Van Jacobson

6. **Terminación del enlace:** Cuando el usuario ha terminado su comunicación usualmente realizando un pedido de desconexión o por causa de un evento físico no previsto, IPCP se encarga de deshabilitar el uso del protocolo IP y liberar la dirección IP asignada al usuario. A continuación, LCP termina el enlace y la computadora indica al módem que libere la conexión física.

El formato de la trama PPP se presenta a continuación:

Bytes	1	1	1	1 o 2	Variable	2 o 4	1
	Bandera 01111110	Dirección 11111111	Control 00000011	Protocolo	Datos	Checksum	Bandera 01111110

Fig. 1.5. Formato de una trama PPP [1]

El formato de PPP se basa en el formato del protocolo de capa enlace HDLC (*High level Data Link Control*) definido por la ISO (*International Standards Organization*).

- **Bandera:** PPP utiliza el caracter 01II1110 (0x7E en hexadecimal), denominado bandera, como delimitador de inicio y fin de trama. En el caso de que este mismo caracter ocurra dentro del campo de datos de la trama, se emplea un mecanismo de transparencia a fin de no conducir a indicaciones erróneas de fin de trama. En enlaces sincronizados a bit, cada 5 bits 1s consecutivos se inserta un bit cero. En enlaces asincrónicos (orientados a byte) se reemplaza el octeto 0x7E por la secuencia 0x7D, 0x5E. Si aparece 0x7D en el campo de datos, se lo reemplaza por la secuencia 0x7D, 0x5D. En recepción se retiran estos bits/caracteres de transparencia. Entre tramas enviadas en forma consecutiva se utiliza una sola bandera. La bandera que sirve de fin de una trama sirve como bandera de inicio de la siguiente trama. [6]

En enlaces sincrónicos, en períodos en que no existe información por transmitir, se envía consecutivamente la secuencia de bandera (0x7E). En enlaces asincrónicos, en cambio, se transmiten consecutivamente bits 1s. Los enlaces asincrónicos emplean 1 bit de inicio, 8 bits de datos y 1 bit de parada.[6]

- **Dirección y Control:** El campo de dirección y el campo de control utilizan siempre 11111111 y 00000011 respectivamente. Según el formato de una trama HDLC, 11111111 en el campo dirección indica una dirección física de difusión para que todas las estaciones acepten la trama y 00000011 en el campo control indica una trama no numerada, es decir que no contiene números de secuencia o acuses de recibo, pues PPP no los utiliza.

  HDLC permite enviar varias tramas en forma consecutiva. Durante el viaje alguna trama puede dañarse, llegando al destino con error. El destino se encarga de indicar cuales tramas han sido recibidas correctamente, enviando acuses de recibo. Un acuse de recibo contiene el número de la siguiente trama que el destino espera recibir. En el caso de PPP, la comunicación durante el establecimiento y la configuración del enlace se basa en un modelo pedido/respuesta. El origen envía un pedido al destino, el destino responde a ese pedido. En este caso no se necesitan números de secuencia, pues siempre se intercambia entre origen y destino o viceversa una única trama.

- **Protocolo:** Especifica el tipo de protocolo que debe aceptar el paquete contenido en el campo de datos. Estos pueden ser: LCP, NCPs, IP, etc. El tipo de protocolo está representado por un número único asignado por la *Internet Assigned Numbers Authority* (IANA). [E]

- **Datos:** Contiene uno o más octetos de datos. El tamaño del campo de datos puede ser negociado durante el establecimiento del enlace, sin embargo el valor por defecto es de 1500 bytes [6] y se utiliza caracteres de relleno en caso de que la información enviada en el campo de datos sea más pequeña que este valor.

[E] http://www.iana.org/numbers.html contiene un listado de todos los números asignados por IANA.

- **Suma de verificación (*checksum*):** Permite realizar la detección de errores de transmisión y solicitar retransmisión. Este campo se encarga durante el establecimiento y configuración del enlace de detectar errores de transmisión en el contenido de la trama. Si una trama llega con error al destino, éste simplemente no responde al pedido del origen, lo cual provoca que un contador en el origen expire y se retransmita la trama. En caso de que llegue bien, el destino responderá al origen prosiguiendo la comunicación.

1.3.5.1 *Link Control Protocol* (LCP)

PPP define el Protocolo de Control de Enlace (LCP - *Link Control Protocol*) como el protocolo encargado de establecer, configurar, probar y terminar la conexión de enlace.

LCP entre otras cosas es el responsable de establecer el enlace, negociar opciones de encapsulación, negociar el tamaño de los paquetes que se van a transmitir, configurar el protocolo de autenticación que se utilizará durante la fase de autenticación, determinar cuando un enlace está funcionando adecuadamente y cuando ha fallado, detectar errores de configuración del enlace y terminar el enlace.

Para realizar estas funciones LCP define tres clases de paquetes: [7]

1. Paquetes utilizados para establecer y configurar un enlace (*Configure-Request*, *Configure-Ack*, *Configure-Nak* y *Configure-Reject*).
2. Paquetes utilizados para terminar un enlace (*Terminate-Request* y *Terminate-Ack*).
3. Paquetes para mantener y administrar un enlace (*Code-Reject*, *Protocol-Reject*, *Echo-Request*, *Echo-Reply*, y *Discard-Request*).

Todo paquete LCP es encapsulado en el campo de datos de la trama PPP. PPP identifica que su campo de datos contiene un paquete LCP utilizando el número hexadecimal C021 en el campo Protocolo de su trama.

El formato del paquete utilizado por LCP se presenta a continuación:

Fig. 1.6. Formato de un paquete LCP [7]

- **Código:** Define la clase de paquete LCP. Si se recibe un paquete con código inválido se envía un paquete *Code-Reject*. Los valores de los códigos de los distintos tipos de paquetes se presentan a continuación:

Código	Tipo de Paquete	Código	Tipo de Paquete
1	*Configure-Request*	7	*Code-Reject*
2	*Configure-Ack*	8	*Protocol-Reject*
3	*Configure-Nak*	9	*Echo-Request*
4	*Configure-Reject*	10	*Echo-Reply*
5	*Terminate-Request*	11	*Discard-Request*
6	*Terminate-Ack*	12	Reservado

Tabla. 1.1. Códigos y Tipos de paquetes LCP [7]

- **Identificador:** Contiene un número aleatorio que permite asociar un pedido con una respuesta. Cuando un extremo envía un paquete de pedido, LCP coloca un número aleatorio en este campo. La respuesta enviada desde el otro extremo asociada al paquete de pedido contendrá el mismo número enviado por el extremo de origen.
- **Longitud:** Establece la longitud en octetos del paquete LCP. Este campo permite determinar hasta que punto van los datos de LCP en el campo de datos del paquete PPP y distinguirlos de los caracteres de relleno que pueden existir.
- **Datos:** Contiene cero o más octetos según el tipo de paquete LCP.

Después de establecida la conexión física entre los extremos, el extremo de origen utiliza LCP para iniciar el establecimiento del enlace. El origen transmite un paquete *Configure-Request* el cual contiene las opciones de configuración deseadas por ese extremo para su enlace, tales como: el máximo tamaño de los paquetes que se transmitirán en el campo de datos de la trama PPP, protocolo de autenticación que se utilizará, uso de compresión para los campos: Dirección, Control y Protocolo en la cabecera de la trama PPP, etc. En caso de que no se transmita una opción se utilizará un valor por defecto asociado a ella.

Si el extremo destino acepta todas las opciones de configuración solicitadas, transmite un paquete *Configure-Ack* como acuse de recibo conteniendo una copia del campo Identificador del paquete *Configure-Request* y de las opciones solicitadas en él. Si el extremo destino no acepta alguna de las opciones, transmite el paquete *Configure-Nak* conteniendo una copia del campo Identificador del paquete *Configure-Request*, una copia de las opciones no aceptadas y una contrapropuesta con los valores apropiados de las opciones que él puede brindar. En respuesta a esto, el extremo de origen envía otro paquete *Configure-Request* aceptando las opciones enviadas en la contrapropuesta del extremo de destino u otras nuevas. Puede darse el caso de que alguna de las opciones enviadas por el extremo origen no sean reconocidas o aceptables mediante negociación en el otro extremo. En este caso, el extremo destino envía un paquete *Configure-Reject* conteniendo todas la opciones rechazadas. En respuesta a este paquete, el origen deberá enviar un nuevo paquete *Configure-Request* cambiando sus opciones de configuración.

Para terminar un enlace, un extremo envía uno o varios paquetes *Terminate-Request*, el cual debe ser respondido por el otro extremo con un paquete *Terminate-Ack*. Si no hay respuesta a un cierto número de paquetes *Terminate-Request,* se dará por terminado el enlace asumiendo que la conexión física ha tenido algún problema o que el otro extremo ha fallado.

Los paquetes *Code-Reject* y *Protocol-Reject*, son utilizados para indicar un error de código en un paquete LCP enviado o que un protocolo no especificado está siendo enviado en una trama PPP, respectivamente. El extremo que recibe estos paquetes deberá modificar el código errado o dejar de transmitir con el protocolo no reconocido, según sea el caso.

LCP permite probar el estado de un enlace mediante los paquetes *Echo-Request* y *Echo-Reply*. Cuando un extremo envía un paquete *Echo-Request* conteniendo un identificador y el campo de datos con cualquier tipo de datos, el otro extremo del enlace deberá enviar un paquete *Echo-Reply* conteniendo el mismo identificador y la misma información en su campo de datos. En caso de no recibirse un *Echo-Reply*, se puede suponer que el enlace o el otro extremo ha fallado y se puede proceder a terminarlo.

El paquete *Discard-Request* puede ser utilizado por LCP para ejecutar y probar un sentido del enlace. Un extremo que recibe este paquete simplemente lo rechaza.

1.3.5.2 *Network Control Protocol* (NCP)

PPP define una familia de Protocolos de Control de Red (NCPs - *Network Control Protocols*) encargados cada uno de ellos de establecer y configurar un protocolo distinto de capa red a fin de que estos últimos puedan enviar y recibir datagramas sobre un enlace serial. PPP permite trabajar simultáneamente con varios protocolos de capa red sobre un mismo enlace serial, cada uno de los cuales deberá ser individualmente configurado con su respectivo NCP.

El NCP para el caso en que IP (*Internet Protocol*) vaya a ser utilizado como protocolo de red se denomina IPCP (*Internet Protocol Control Protocol*).

1.3.5.3 *Internet Protocol Control Protocol* (IPCP)

El Protocolo de Control IP (IPCP - *Internet Protocol Control Protocol*) es el responsable de: configurar, habilitar y deshabilitar el uso del protocolo IP en ambos extremos de un enlace punto a punto.

Generalmente un proveedor de servicios de Internet posee un conjunto limitado de direcciones IP, que debe administrar dinámicamente, asignando a cada nuevo *host* que se conecta a él, a través de un enlace serial, una dirección IP durante el período que dure su sesión de trabajo. IPCP es el protocolo encargado de la asignación y administración de direcciones IP requeridas por el protocolo IP en cada extremo de un enlace punto a punto. Una vez que IPCP termina el uso del protocolo IP sobre el enlace serial, libera la dirección IP para que ésta pueda ser reutilizada por otro *host*.

El funcionamiento de IPCP es similar al del protocolo LCP. IPCP utiliza el mismo formato de paquete LCP y algunos de los mensajes utilizados por LCP para configurar el uso de IP sobre el enlace serial. Todo paquete IPCP es encapsulado en el campo de datos de la trama PPP. PPP identifica que su campo de datos contiene un paquete IPCP

utilizando el número hexadecimal 8021 en el campo Protocolo de su trama.[8]

El formato del paquete utilizado por IPCP se presenta a continuación:

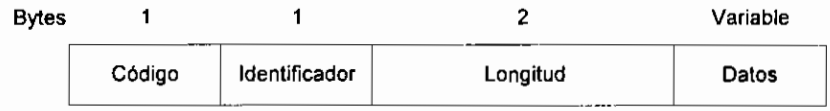


Fig. 1.7. Formato de un paquete IPCP [8]

El significado de los campos es el mismo que el del formato LCP. A diferencia de LCP, IPCP únicamente define los tipos de mensajes, mostrados en la tabla 1.2, como válidos. El significado y uso de estos paquetes es el mismo que el detallado en el caso de LCP, sólo que en este caso, los mensajes IPCP configuran y terminan el protocolo de capa red (IP), por lo que las opciones de configuración también cambian.

Código	Tipo de Paquete
1	*Configure-Request*
2	*Configure-Ack*
3	*Configure-Nak*
4	*Configure-Reject*
5	*Terminate-Request*
6	*Terminate-Ack*
7	*Code-Reject*

Tabla. 1.2. Códigos y tipos de paquetes IPCP [8]

IPCP define un formato de paquete para las opciones:

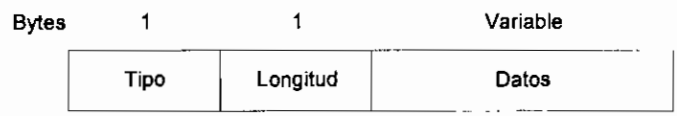


Fig. 1.8. Formato de un paquete de opción IPCP [8]

Este paquete de opciones va encapsulado en el campo de datos de un mensaje IPCP. IPCP define las siguientes opciones de configuración, cada una de las cuales tiene asociada un código en el campo Tipo: [8]

Código	Opción
1	Direcciones IP
2	Protocolo de compresión IP
3	Dirección IP
129	Dirección IP del servidor DNS primario
131	Dirección IP del servidor DNS secundario

Tabla. 1.3. Tipos de opciones de configuración IPCP

El campo Longitud establece la longitud del campo de datos de la opción y el campo de datos contiene la información asociada a esa opción.

La primera opción, direcciones IP, está en desuso. La opción de protocolo de compresión IP permite negociar el uso de un protocolo de compresión de cabeceras TCP/IP específico. Por defecto no se utiliza compresión. En caso de usar compresión, el protocolo utilizado es el protocolo de compresión TCP/IP de Van Jacobson, descrito en el RFC 1144.

La opción de Dirección IP provee un mecanismo para negociar la dirección IP que debe ser utilizada por un extremo del enlace serial. Esta opción permite al extremo de origen enviar un mensaje *Configure-Request* indicando la dirección que desea le sea asignada, o enviar la dirección 0.0.0.0 solicitando que el proceso par (*peer*) en el extremo destino sea el encargado de asignarle una nueva dirección en forma dinámica. Si el proceso par no dispone de la dirección IP solicitada por el extremo origen, puede enviarle un mensaje *Configure-Nak*, rechazando la dirección solicitada y asignándole una dirección IP disponible.

Las opciones de configuración 129 y 131 son en realidad extensiones definidas en el RFC 1877 que permiten usar IPCP para obtener las direcciones de servidores DNS (primarios y secundarios) en la red remota. El usuario puede enviar en el pedido la dirección IP de un servidor DNS conocido o emplear el valor 0.0.0.0, para indicar que el extremo remoto sea el encargado de asignarle la dirección del servidor DNS primario y si fuese el caso de un secundario.

Una vez que IPCP ha terminado de configurar IP como protocolo de red, se puede iniciar el intercambio de datagramas IP a través del enlace serial. Cada datagrama IP viaja encapsulado en el campo de datos de la trama PPP. El campo Protocolo utiliza el número hexadecimal 0021 [6] para indicar que en el campo de datos está contenido un datagrama IP.

1.4 PROTOCOLOS DE CAPA INTERNET

1.4.1 *Internet Protocol* (IP)

El protocolo IP es un protocolo no confiable, no orientado a conexión, encargado de:

- Definir el formato del datagrama IP, que es la unidad básica de transferencia de datos en la capa internet.
- Realizar el ruteo de los datagramas.
- Definir el conjunto de reglas que caracterizan la distribución de datagramas en forma no confiable y sin conexión

En la figura 1.9 se presenta el formato de un datagrama IP. Un datagrama IP está formado por una cabecera y un campo de datos. En el campo de datos se encapsulan los paquetes: TCP, UDP, ICMP, IGMP, etc.

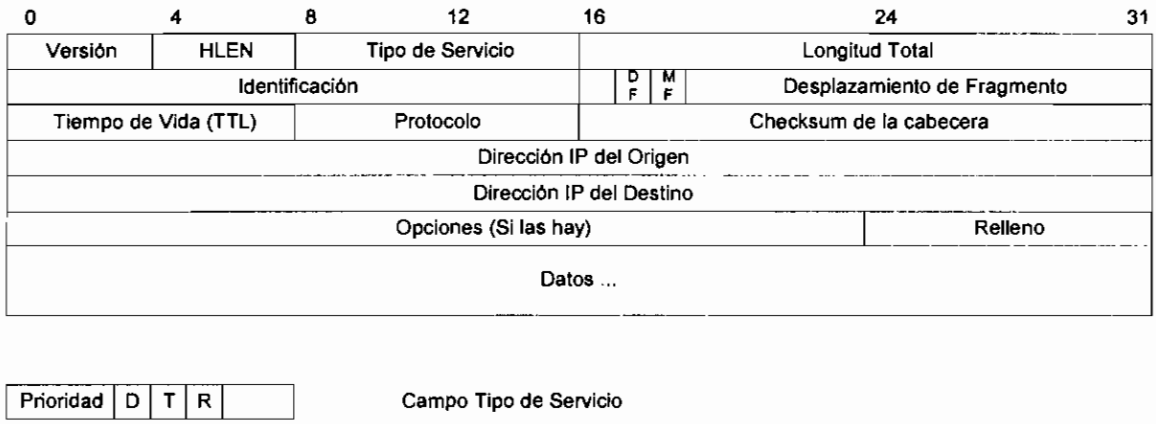


Fig. 1.9. Formato de un datagrama IP [2]

- **Versión:** Indica la versión del protocolo IP utilizado. Actualmente, versión 4 (IPv4).
- **HLEN:** Define la longitud de la cabecera en unidades de 32 bits.
- **Tipo de Servicio:** Especifica como un protocolo de capa superior desea le sean enviados sus datagramas a través de la subred de comunicaciones. Para lo cual el campo Tipo de Servicio posee los siguientes subcampos: Prioridad del datagrama, retardo de transmisión (D - *Delay*), capacidad (T - *Throughput*), confiabilidad (R - *Reliability*), según los cuales el algoritmo de ruteo escogerá una ruta u otra para enviar el datagrama a su destino.
- **Longitud Total:** Define la longitud total del datagrama (cabecera y datos) medida en octetos.
- **Identificación:** Contiene un número entero que permite identificar un datagrama, su uso se aclarará posteriormente. Cuando se crea un nuevo datagrama el valor del campo Identificación es el valor que se tenía en el último datagrama aumentado en una unidad.
- **Bit DF:** Indica si un datagrama puede (DF = 0) o no (DF = 1) ser fragmentado.
- **Bit MF:** Indica si un fragmento es (MF = 0) o no (MF =1) el final de un datagrama.
- **Desplazamiento de fragmento:** Posición de un fragmento dentro de un datagrama.
- **Tiempo de Vida (TTL):** Teoricamente indica el tiempo máximo en segundos que un datagrama puede estar circulando por la red antes de ser descartado. El valor máximo es de 255 segundos. En la práctica cuenta el número de saltos que ha dado el datagrama. Cada vez que un datagrama llega a un ruteador, el ruteador decrementa el valor contenido en el campo TTL en una unidad. Cuando el campo TTL llega a cero, un ruteador descarta el datagrama y envía un paquete ICMP al origen que envió el datagrama, indicándole que el paquete ha sido eliminado. Este campo evita tener datagramas perdidos en la subred de comunicaciones que consuman recursos innecesariamente.
- **Protocolo:** Contiene el código numérico, asignado por IANA, de un protocolo de capa superior que en el destino final deberá recibir los paquetes de datos contenidos en el campo de datos del datagrama.
- **Suma de verificación (*checksum*):** Permite detectar errores que puedan ocurrir únicamente en la cabecera del datagrama durante su transmisión por la red. El *checksum* no detecta errores sobre los datos transmitidos en el datagrama.
- **Dirección IP del origen:** Contiene la dirección IP de la máquina origen.

- **Dirección IP de destino:** Contiene la dirección IP de la máquina destino.
- **Opciones:** Es un campo opcional y de longitud variable que permite implementar pruebas y control de la red. Por ejemplo se define una opción para especificar la ruta por la cual debe circular un datagrama, otra opción permite almacenar las direcciones IP e instantes de tiempo de los ruteadores por los cuales atravesó el datagrama en su viaje hacia el destino, etc. A los instantes de tiempo se los llama sellos de tiempo. El sello de tiempo contiene la hora local y fecha en la que un datagrama llegó a un determinado ruteador.

Un problema que debe afrontar la capa internet antes de enviar un datagrama a la capa interfaz de red es la determinación del tamaño máximo de datagrama que se puede enviar en el campo de datos de una trama. El problema se debe a que un datagrama en su viaje entre un origen y un destino debe atravesar un sinnúmero de distintos tipos de redes, cada una de las cuales posee una capacidad máxima de transmisión de datos en el campo de datos de sus tramas; así por ejemplo, una red Ethernet permite una transferencia de datos de hasta 1500 octetos, en tanto que una red FDDI permite aproximadamente 4470 octetos. [2]

Se define el MTU (*Maximun Transfer Unit*) o unidad de transferencia máxima de una red como el tamaño máximo de datos que se puede transferir en una determinada trama de un protocolo en un determinado tipo de red.

La capa internet debe seleccionar el tamaño de datagrama más conveniente y establecer una forma de dividir en fragmentos pequeños un datagrama cuando éste debe viajar a través de una red que tiene una MTU más pequeña. El proceso de dividir un datagrama en pequeños fragmentos se denomina Fragmentación. En el *host* de destino, el datagrama debe ser reensamblado a partir de los fragmentos antes de poderlo procesar. Cada fragmento de un datagrama poseerá la misma cabecera del datagrama original con ligeras variantes en algunos de sus campos. El campo de Identificación en la cabecera IP, especificará a qué datagrama pertenece un determinado fragmento, ya que todos los fragmentos de un datagrama poseen el mismo número de identificación. El campo de Desplazamiento de Fragmento indica, durante el reensamblado, la posición del fragmento dentro del datagrama original. El bit MF (*More Fragments*) indicará si

existen más fragmentos cuando está en uno o indicará, cuando está en cero, que es el último fragmento de un datagrama. El *checksum* será calculado independientemente para cada fragmento.

Una forma de determinar el MTU máximo de una red es enviar tramas al ruteador conteniendo un datagrama con el bit DF (*Don't Fragment*) puesto en uno, el cual indica que el datagrama no debe ser fragmentado por ningún motivo. El proceso se inicia con un datagrama de tamaño pequeño, en los siguientes datagramas se va incrementando su longitud hasta llegar a un punto en el cual el ruteador tratará de fragmentar, pero como está impedido de hacerlo, enviará una trama conteniendo un paquete indicando que ha ocurrido un error. Este paquete ICMP a su vez viaja contenido en un datagrama IP. Este evento permite determinar el MTU óptimo y enviar los restantes datagramas con este MTU.

El protocolo IP se dice no confiable ya que no garantiza la entrega de datagramas. Los datagramas pueden perderse, duplicarse o dañarse a lo largo del trayecto y depende de las capas superiores el implementar algoritmos para enfrentar estos problemas.

1.4.1.1 Direcciones IP

Cada *host* y cada ruteador en Internet tienen asociada una dirección, denominada dirección IP, la cual es utilizada por el protocolo IP para identificar en forma única una conexión de red con dicha máquina. Una dirección IP no identifica a una máquina en si misma, sino una conexión de red; es decir, cuando un *host* o un ruteador está conectado a más de una red debe tener asignada una dirección IP por cada interfaz de red.

Una dirección IP está formada por una combinación de 32 bits, representada como cuatro números decimales separados entre si por un punto. Cada número decimal corresponde a cada uno de los cuatro bytes de los 32 bits de la dirección. Esta notación se denomina *dotted-decimal*. Así, por ejemplo, la dirección hexadecimal C0BC393C se la representa como 192.188.57.60

Una dirección IP está formada por dos partes: un número de red (*netid*) que identifica una red y un número de *host* (*hostid*) que identifica un *host* en esa red.

Dirección IP = <número de red><número de *host*>

Existen cinco clases de direcciones IP, las cuales se muestran a continuación:

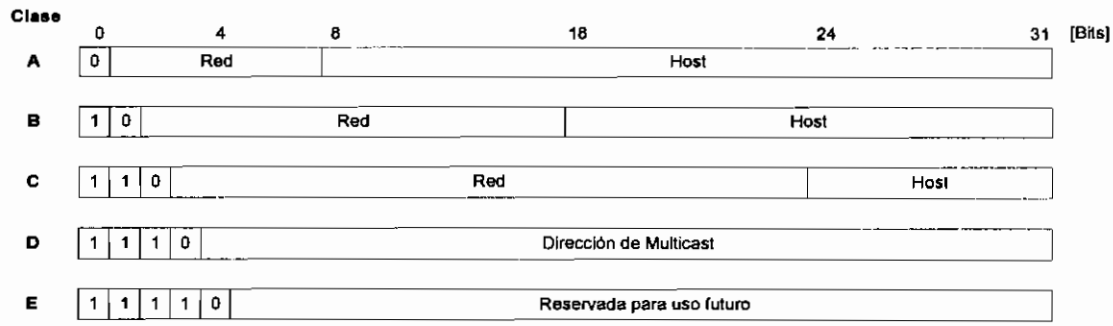


Fig. 1.10. Clases de direcciones IP [1]

Los primeros bits de la dirección indican la clase de dirección IP. A continuación viene el número de red y finalmente el número de *host*.

- Las direcciones clase A son utilizadas para redes grandes con muchas máquinas. Se identifican porque inician con 0. Definen con 7 bits el número de red y con 24 bits el número de *host*.
- Las direcciones clase B son utilizadas para redes de tamaño medio. Se identifican porque inician con 10. Definen con 14 bits el número de red y con 16 bits el número de *host*.
- Las direcciones clase C son para redes pequeñas con menos de 256 máquinas. Se identifican por que inician con 110. Definen con 21 bits el número de red y con 8 bits el número de *host*.
- Las direcciones clase D son utilizadas para permitir comunicaciones punto-multipunto (*multicasting*), las cuales permiten enviar un mismo datagrama a un grupo de *hosts*. Las direcciones clase D o de *multicast* se identifican porque inician con 1110.
- Las direcciones clase E inician con 11110 y su uso está reservado para el futuro.

La tabla 1.4 presenta los rangos de direcciones correspondientes a cada clase, el número máximo de redes, y el número máximo de *hosts* por red, que cada rango de direcciones puede tener:

Clase	Rango	Número de Redes	Número de *hosts*/red
A	1.0.0.0 a 127.255.255.255	126	16777214
B	128.0.0.0 a 191.255.255.255	16382	65534
C	192.0.0.0 a 223.255.255.255	2097150	254
D	224.0.0.0 a 239.255.255.255	-	-
E	240.0.0.0 a 247.255.255.255	-	-

Tabla. 1.4. Rangos de direcciones IP, número de redes y *hosts*/red en cada clase [1,3]

El valor de todos los bits 0 o todos los bits 1, en un campo *netid, hostid* o ambos, tienen significados especiales en una dirección IP y no pueden ser asignadas a ningún *host* en ninguna red.

El valor de todos los bits 0 en un campo *hostid*, *netid* o ambos sirve de identificador de red, un *host* en una red, o "este *host*" respectivamente.

El valor de todos los bits 1 en un campo *hostid* significa difusión (*broadcast*) a todos los *hosts* en una red remota, en tanto que si los 32 bits de una dirección son 1, significa difusión en la red local.

Las direcciones 127.x.y.z , pertenecientes a la clase A son utilizadas como *loopback*, es decir para probar la comunicación de procesos internos en una máquina local. Al enviar un datagrama a esta dirección como destino no se genera tráfico en la red ya que los datos son devueltos al *software* TCP/IP para su procesamiento local. Ninguna dirección en la red puede poseer un número de red 127. Este número no representa un número de red.

A continuación se presenta un resumen de las direcciones IP especiales, que no pueden ser asignadas a ningún *host*, y su respectivo significado:

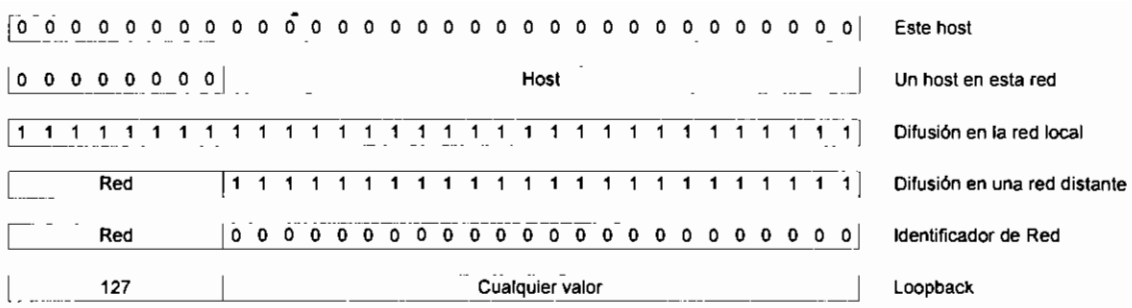


Fig. 1.11. Direcciones IP especiales [1]

1.4.1.2 Asignación de direcciones IP

Cuando una institución, organización o empresa, desea conectar su red privada a Internet, debe utilizar direcciones IP válidas y únicas.

El organismo encargado de asignar direcciones IP a nivel mundial es el Centro de Información de la Red Internet (INTERNIC - *Internet Network Information Center*).

Cuando una organización realiza la solicitud de asignación de direcciones IP a INTERNIC, directamente o través de sus representantes nacionales, éste dependiendo del tamaño de la red de la organización, le asignará generalmente un único número de red, correspondiente a una red tipo A, B o C, ó una subred como se verá posteriormente. Cabe notar que INTERNIC asigna a la organización solo el número de red (campo *netid*) de la dirección IP; el administrador de la red de la organización es libre de administrar el campo *hostid* a su voluntad, sin necesidad de consultar a INTERNIC.

El campo *hostid*, como lo indica la tabla 1.4., podrá tomar 16777214, 65534 o 254 valores dependiendo de que el número de red asignado por INTERNIC haya sido el de una red clase A, B o C respectivamente.

El administrador de red podrá asignar libremente a cada *host* (conexión de red), un único valor (no repetido) de los posibles que puede tomar el campo *hostid*. La combinación número de red - número de *host*, es decir, la dirección IP, identificará en forma única a un *host* de esa red en Internet.

A fin de diferenciar qué parte de los 32 bits de una dirección IP identifican a un *host* y que parte identifica la red, se utiliza lo que se denomina una máscara de red. Una máscara de red no es sino un conjunto de 32 bits, en los cuales los bits con valor cero indican los bits disponibles para identificar a los *hosts* dentro de una red (campo *hostid*), en tanto que los bits con valor uno indican los bits con que se identifica una red (campo *netid*). La máscara de red se la representa en notación *dotted-decimal.*

A partir de la figura 1.10, se desprende que las redes tipo A, B y C poseen respectivamente las siguientes máscaras de red:

Tipo de Red	Máscara por defecto
A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

Tabla. 1.5. Máscaras de red correspondientes a redes tipo A, B y C

Como se verá a continuación, el administrador de red es libre de dividir el campo *hostid*, en un campo denominado campo de subred y otro denominado campo de *host* para crear lo que se denominan subredes.

1.4.1.3 Subredes

Cuando una nueva red física es instalada en una localidad o el crecimiento del número de *hosts* requiere dividir las redes locales en dos o más redes separadas, muchas instituciones encuentran conveniente dividir sus redes en subredes para evitar tener que solicitar direcciones IP adicionales al INTERNIC.

Para formar subredes, la parte del número de *host* de la dirección IP original es subdividida en un número de subred o red interna y un número de *host*. La combinación del número de subred y el número de *host* se la denomina dirección local o parte local diferenciándolo del número de red que representa la parte externa.

La dirección IP en este caso se la interpreta como:

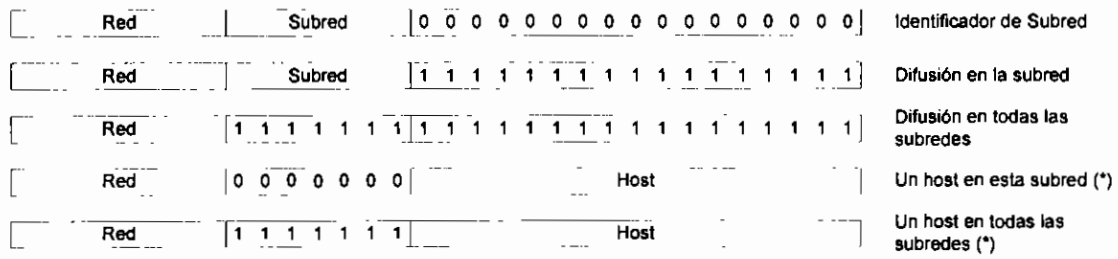
Dir. IP subneteada = <número de red ><número de subred ><número de *host*>

La subdivisión en subredes se la realiza empleando una máscara de subred. Una máscara de subred, es una extensión al concepto de máscara de red, en la cual los bits con valor uno dentro del campo *hostid* de la máscara de red original indican los bits que identifican una subred dentro de la red original, en tanto que los bits con valor cero indican los bits que identifican a los *hosts* dentro de la subred.

El valor de todos los bits 0 o todos los bits 1, en un campo número de red, número de subred o número de *host* de una dirección IP subneteada, tienen significados especiales de identificación o de difusión respectivamente, y no pueden estar presentes en ninguna dirección IP asignada a un *host*.

En la figura 1.12 se presenta un resumen de las direcciones IP especiales, que no pueden ser asignadas a ningún *host* o subred, y su respectivo significado.

Como se verá en un ejemplo posterior, ya que el valor todos cero o todos unos no está permitido, ni en el campo de número de subred, ni en el campo número de *host*, esto hace que siempre se reduzca en "dos" el número de posibles subredes que se pueden crear y también reduce en dos el número de *hosts* que una subred puede tener.



(*) : No tiene significado práctico

Fig. 1.12. Direcciones IP especiales utilizadas con subredes

El administrador local de la red es libre de dividir la parte local de la dirección IP en un número de subred y un número de *host* sin tener que contactar con el INTERNIC. El número de red identificará a todas las subredes contenidas en ella.

La división en subredes, también conocida como direccionamiento de subred, resulta transparente para las redes remotas, es decir un *host* dentro de una red dividida en subredes conoce que existen subredes, pero un *host* en una red externa no lo sabe. Para las redes externas, una red con subredes actúa como una única red. Esto tiene ventajas para el ruteo de los datagramas como se verá posteriormente.

Ejemplo 1.1.

Se ha asignado a una organización una red tipo B, 129.112.0.0 con máscara de red 255.255.0.0. Dividir esta red en cuatro subredes ■

Si se utilizase los 2 bits más significativos del campo número de *host* de la máscara original para identificar el campo número de subred, es decir, modificando la máscara original 255.255.0.0 a 255.255.**11**000000.00000000 = 255.255.**192.**0, se tendría 4 posibles valores (00, 01, 10, 11) que el número de subred podría tomar. Sin embargo, los valores 00 y 11 no son permitidos para ninguna subred pues tienen significados especiales como se ha mencionado anteriormente. Esto resultaría únicamente en dos subredes válidas (129.112.**01**000000.0 = 129.112.**64**.0 y 129.112.**10**000000.0 = 129.112.**128**.0), lo cual no satisface la condición pedida.

Usando los 3 bits más significativos del campo número de *host* en la máscara original para identificar el campo número de subred, tendríamos 8 posibles valores (000, 001, 010, 011, 100, 101, 110, 111) que el número de subred podría tomar. Al igual que en el caso anterior, 000 y 111 tienen significados especiales y no pueden ser utilizados para identificar a ninguna subred. Quedan por lo tanto seis valores posibles que el campo número de subred puede tomar. Cada uno de estos valores identifica una subred diferente. Como se necesitan 4 subredes y no hay forma de obtener el valor justo, se utilizarán únicamente 4 de las 6 posibles subredes, las otras dos quedan reservadas para uso futuro o para interconectar las subredes creadas.

La máscara de subred que permite realizar la división deseada sería de la siguiente forma:

Máscara: 255.255.**111**000000.00000000 = 255.255.**224**.0

Con esta máscara las subredes formadas, sus respectivas direcciones de difusión y el número de *hosts* que cada una de ellas tendría serían:

Subred	Dirección IP de la subred		Dirección de Difusión	Número de *host*/subred
1	129.112	.32.0	129.112.63.255	8190
2		.64.0	129.112.95.255	8190
3		.96.0	129.112.127.255	8190
4		.128.0	129.112.159.255	8190
5		.160.0	129.112.191.255	8190
6		.192.0	129.112.223.255	8190

Tabla. 1.6. Características de las subredes creadas en el ejemplo 1.1.

Los bits cero, en la máscara 255.255.224.0 = 255.255.11100000.0000000, indican que pueden ser asignados para identificar *hosts* en una determinada subred. En este caso hay 13 bits que identifican el campo *hostid*, por lo cual hay $2^{13} = 8192$ posibles valores que puede tomar este campo, exceptuando los valores de todos los bits cero y todos los bits uno, que poseen significados especiales, quedan 8190 posibles valores que pueden ser asignados a *hosts* en esa subred. Es decir, en cada una de las subredes creadas pueden existir hasta 8190 *hosts*.

Cada subred tendrá ahora como máscara el valor 255.255.224.0. El administrador de cada subred podría subdividirla en más subredes, desplazando la máscara hacia la derecha.

La división en subredes realizada en el ejemplo anterior permitiría, por ejemplo, dividir una misma red física en cuatro subredes lógicas tal como se muestra a continuación. Estas subredes a pesar de estar conectadas a una misma red física, no podrían comunicarse entre sí. Para permitir la comunicación se requeriría de dispositivos denominados ruteadores como se verá posteriormente.

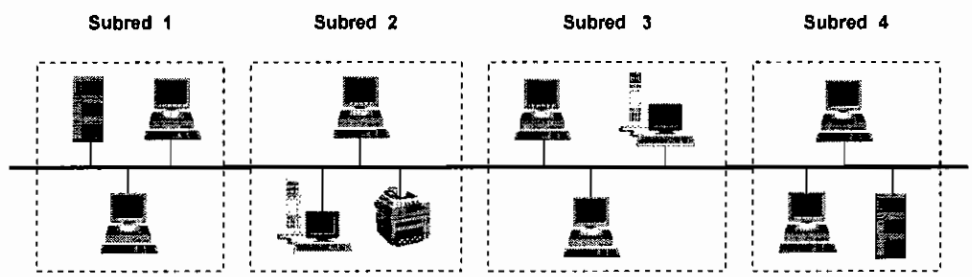


Fig. 1.13. Ejemplo de división de una red física en cuatro subredes lógicas

La misma división en subredes del ejemplo permitiría asignar cada uno de estos conjuntos de direcciones de subred a cuatro redes físicas dentro de una misma organización como se muestra a continuación.

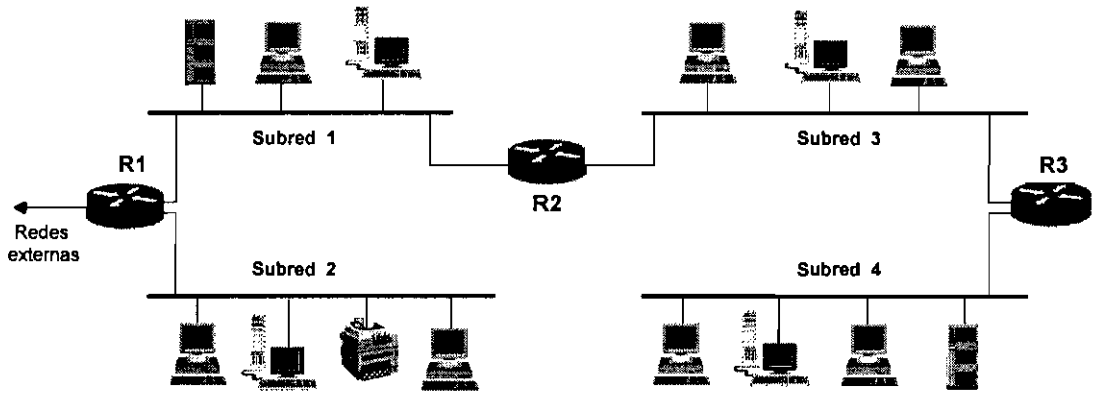


Fig. 1.14. Ejemplo de cuatro subredes físicas que utilizan direccionamiento de subred

Cuando se tienen varias redes físicas puede ser necesario interconectarlas entre si para permitir la comunicación entre *hosts* ubicados en distintas redes. Por ejemplo en la figura anterior R1, R2 y R3 son ruteadores que permiten la interconexión de las subredes mostradas.

En el Anexo B, se presentan 3 tablas que permiten realizar la división de subredes en forma inmediata, dependiendo del tipo de red A, B o C original, sin necesidad de realizar ningún cálculo. En las tablas se identifican el valor que debe ser añadido a la máscara original, el valor binario correspondiente, el número de subredes creadas, número de *hosts* por subred, rangos donde inicia y termina cada subred, rangos de posibles direcciones IP que puede tener un *host* en la red.

Ejemplo 1.2.

Usando la tabla 2 del Anexo B, se puede dividir la red tipo B 129.112.0.0, del ejemplo anterior, en 10 subredes. En la fila 3, que indica el número de posibles subredes que se pueden crear con diferentes máscaras, se busca el valor exacto (en este caso 10) o el valor inmediato superior al valor de posibles subredes que se desean crear. En este ejemplo, puesto que los únicos posibles valores de subdivisión son 2, 6, 14, 30 o 62

subredes, se toma el valor inmediatamente superior a 10, que es 14 y que se encuentra en la columna 3. La fila 1 de esta columna indica que la máscara 255.255.0.0 de la red B original deberá modificarse a 255.255.**240**.0. La red original quedará dividida en 14 subredes, de las cuales únicamente se emplearán 10.

Algunas de las subredes creadas serían por ejemplo: 129.112.**16.0**, 129.112.**32.0**, 129.112.**48.0**, etc. Cada subred tendrá hasta 4094 *hosts* (valor obtenido de la fila 4). En la subred 129.112.16.0, los *hosts* podrán tomar valores comprendidos entre 129.112.16.1 y 129.112.31.254. La dirección de difusión de la subred, sería el valor inmediatamente inferior al identificador de la siguiente subred. En el caso de la subred 129.112.16.0, su dirección de difusión sería 129.112.31.255.

1.4.1.4 Direcciones Reservadas para Intranet

La asignación de una clase de dirección IP, por parte de INTERNIC a una organización, depende del tamaño de la red que ésta disponga. Así las direcciones clase A serán asignadas a redes con un gran número de *hosts*, y las direcciones clase C serán adecuadas para redes con un pequeño número de *hosts*. Las direcciones tipo B serían adecuadas para redes de tamaño medio, es decir, aquellas con más de 254 *hosts* o donde se espera que pueda haber más de 254 *hosts* en el futuro. Sin embargo, el número de redes entre pequeñas y medianas ha estado creciendo rápidamente en los últimos años y está previsto que si el crecimiento continua así, el número de direcciones clase B se agotará muy pronto.

El RFC 1597 ha definido los siguientes rangos de direcciones IP para ser utilizadas por Intranets o redes TCP/IP privadas, sin necesidad de que deban solicitar permiso alguno al INTERNIC.

Cantidad	Tipo	Rango de direcciones IP	
		Desde	Hasta
1	Red Clase A	10.0.0.0	10.255.255.255
16	Redes Clase B	172.16.0.0	172.31.255.255
256	Redes Clase C	192.168.0.0	192.168.255.255

Tabla. 1.7. Rango de direcciones IP reservadas para Intranet [9]

Estos rangos de direcciones se consideran como direcciones no válidas, es decir que ninguna máquina conectada a Internet puede poseer. Si una Intranet desea conectarse a Internet, deberá utilizar direcciones válidas de Internet asignadas por el INTERNIC.

Como se verá en el capítulo III, la Intranet podrá utilizar tranquilamente las direcciones reservadas (no válidas) en la red interna y a través de un servidor *proxy* que posee una o varias direcciones IP válidas conectarse a Internet. El servidor *proxy* se encargará de realizar la transformación de direcciones no válidas en una o varias direcciones válidas y viceversa.

El uso de las direcciones recomendadas en el RFC 1597 es una forma de solucionar el agotamiento de direcciones IP, antes mencionado, evitando que una Intranet deba solicitar toda una red B o A para conectar sus máquinas con direcciones válidas.

Otra técnica empleada por INTERNIC para evitar el agotamiento de direcciones IP, ha sido el asignar a una organización, no una red A o B completa, sino únicamente una o varias redes tipo C, o una o varias subredes de una red tipo A o B.

1.4.2 Interconexión de redes

Para poder interconectar dos redes entre si se necesita un dispositivo que se encuentre conectado a ambas redes y que pueda enviar paquetes desde la una a la otra. Existen varios dispositivos que permiten interconectar redes entre si, entre los cuales están: repetidores, puentes, ruteadores y *gateways*. Cada uno de éstos permite interconectividad a determinado nivel.

1.4.2.1 Repetidores

Son dispositivos que trabajan a nivel físico y que permiten extender el tamaño de una red. Se encargan de regenerar las señales al nivel adecuado para su transmisión de un segmento a otro.

1.4.2.2 Puentes o *Bridges*

Sin entrar en detalle, se puede mencionar que la capa enlace en redes LAN 802.x está formada por dos subcapas: la subcapa de Control de Acceso al Medio de transmisión (MAC - *Medium Access Control*) y la subcapa de Control de Enlace Lógico (LLC - *Logical Link Control*). La subcapa MAC define los protocolos encargados de establecer el control del acceso al medio de transmisión entre varias máquinas que compiten por él para transmitir sus tramas. A nivel de subcapa MAC, las redes LAN 802 difieren en el formato de sus tramas. La subcapa LLC se encarga de ocultar las diferencias entre las diferentes redes 802 utilizando un formato único y una misma interfaz hacia la capa red.

Los paquetes enviados por la capa red son encapsulados en la trama LLC cuyo formato es el mismo para todas las redes LAN 802. Esta trama LLC a su vez es encapsulada en una trama de subcapa MAC, la cual posee cabecera y *trailer* que varían según el tipo de LAN. En la máquina de destino se da la secuencia contraria hasta entregar el paquete respectivo a la capa red.

Un puente da la posibilidad de interconectar múltiples LANs a nivel de capa enlace, permitiendo el envío de tramas entre las distintas LANs en base a la dirección de trama. Un puente se encarga de realizar la conversión de capa MAC cuando se requiere. En forma resumida su función es la de extraer la trama LLC de una trama MAC de una red y acomodarla en una trama MAC de otra red cuando la trama está destinada a esa red.

Un puente puede ser implementado en base a *software* corriendo en un *host* o en *hardware* mediante un dispositivo de propósito especial.

Un puente permite reducir la congestión de tráfico entre redes LAN conectadas entre si, ya que únicamente envía tramas direccionadas hacia otra red y no las tramas cuyo destino se encuentran en el mismo segmento de LAN.

Un puente se dice transparente a IP; es decir cuando un *host* envía un datagrama a otro *host* en una red distinta conectada a la primera por un puente, el datagrama es enviado directamente al *host* de destino. Para el protocolo IP de capa internet el puente

simplemente no existe, ya que éste trabaja a nivel de capa enlace. Un puente consecuentemente no posee ninguna dirección IP.

1.4.2.3 Ruteadores

Un ruteador es un dispositivo que permite conectar una red con una o más redes a nivel de capa internet. Un ruteador puede enviar datagramas provenientes de una interfaz de red hacia otra interfaz en función de la dirección IP de destino del datagrama.

Un ruteador puede ser un *hardware* de propósito especial o un computador con múltiples interfaces de red, una por cada red a la que esté conectado y con una dirección IP por cada interfaz de red que posea. Este computador puede actuar como un ruteador entre las distintas redes a las que esté conectado. En la mayoría de casos, los ruteadores de propósito especial proveen mejor desempeño y confiabilidad que sistemas de propósito general actuando como ruteadores.

1.4.2.4 *Gateways*

Un *gateway* es un dispositivo que permite interconectar redes a nivel de capas superiores a la que lo hacen los puentes y ruteadores. Un *gateway* permite convertir el formato de los datos utilizados por una aplicación en una red a un formato distinto utilizado por una aplicación similar en otra red.

Un *gateway* permitiría interconectar dos redes cada una de las cuales utiliza distintos protocolos a nivel de capa red y transporte. El *gateway* se encarga de extraer el mensaje enviado por una máquina en una red y enviarlo a la otra red, utilizando los formatos y protocolos de capa red y transporte de dicha red.

Los *gateways* típicamente limitan la interconectividad de dos redes a un subconjunto de protocolos de aplicación soportados en cada uno.

Un *gateway* se dice opaco para IP; es decir un *host* no puede enviar un datagrama IP a través de un *gateway*, sólo puede enviarlo a un *gateway*. La información del protocolo

de mayor nivel enviada en un datagrama es luego pasada por el *gateway* usando los protocolos adecuados de la arquitectura de red que sea utilizada en el otro lado del *gateway.*

1.4.3 Ruteo de datagramas IP

El ruteo de los datagramas desde un *host* de origen a un *host* de destino es la principal función de la capa internet. El protocolo IP es un protocolo de ruteo. IP se encarga de rutear los datagramas haciendo uso de algoritmos de enrutamiento que le permitirán crear tablas de ruteo en cada máquina, mediante las cuales decidirá el camino por el cual un datagrama debe ser transmitido. En el ruteo de los datagramas participan tanto *hosts* como ruteadores.

Una tabla de ruteo, tanto en *hosts* como en ruteadores, contiene registros de la siguiente forma: [F]

<máscara de red o subred>,<dirección de red o subred>,<dirección IP del siguiente salto>

Cuando una red no está dividida en subredes, el valor del campo <máscara de red> será: 255.0.0.0, 255.255.0.0 o 255.255.255.0 dependiendo de si la red es tipo A, B o C respectivamente.

Aquellas redes que estén dividas en subredes, emplean en el campo <máscara de subred> de la tabla de ruteo, el valor de máscara de subred adecuado.

El campo <dirección de red o subred> contiene la dirección IP de una red o subred de destino (identificador de red o subred). El campo <dirección IP del siguiente salto>, contiene la dirección IP del ruteador al cual se debe enviar un datagrama para llegar a la red/subred indicada. Cuando el ruteador está conectado a la misma red o subred en la que se encuentra el *host* de destino, la entrega del datagrama se la hace en forma directa

[F] Según el algoritmo unificado de ruteo, el cual abarca ruteo de subred. Algunos ruteadores no son capaces de realizar ruteo de subred por lo cual no utilizan el campo <máscara de subred> en sus tablas.

Es necesario aclarar que el campo dirección de red en una tabla de ruteo, en forma general no contiene direcciones IP de un *host* individual, sino una dirección de red o subred de destino. Esto permite minimizar el tamaño de las tablas de ruteo y como se verá, hace que el ruteo sea más eficiente. En casos especiales se puede utilizar la dirección IP de un *host* individual de destino en el campo de dirección de red. En este caso se emplea en el campo <máscara de red o subred> el valor 255.255.255.255.

Cuando un datagrama llega a un ruteador, éste realiza la operación booleana AND entre el campo <máscara de red o subred> de un registro y la dirección IP de destino contenida en el datagrama entrante. El resultado de esta operación booleana será el identificador de dirección de red o subred, (ver figuras 1.11 y 1.12) en la que se encuentra el destino, o la dirección IP de un *host* individual en un caso especial. El resultado de la operación booleana se lo compara con el valor contenido en el campo <dirección de red o subred> de ese registro. En caso de que no sean iguales, se repite la operación con el siguiente registro, y así sucesivamente con todos los registros de la tabla, hasta que en alguno de ellos el resultado de la operación booleana y el campo dirección de red sean iguales. Cuando se produce la igualdad, el ruteador enviará el datagrama recibido a la dirección especificada en el campo <dirección IP al siguiente salto>. En caso de que ninguna dirección de red corresponda al resultado de la operación booleana el ruteador enviará un mensaje de error, mediante el protocolo ICMP, especificando un destino como inalcanzable. En caso de que para un mismo destino existan distintas rutas, el ruteador podrá seleccionar la mejor ruta según el pedido solicitado en el campo Tipo de Servicio en el datagrama.

En la figura 1.15 se resume el proceso general de ruteo.

El proceso descrito anteriormente se cumple también cuando un *host* desea enviar un datagrama a un destino. El *host* primeramente chequea si la dirección de destino está en su misma red local o no. Cuando el *host* de destino pertenece a la misma red que el *host* de origen no requiere utilizar ruteadores. En este caso después de determinar la dirección de *hardware* del destino haciendo uso del protocolo ARP, el datagrama es encapsulado en una trama de la red física y entregado directamente a su destino.

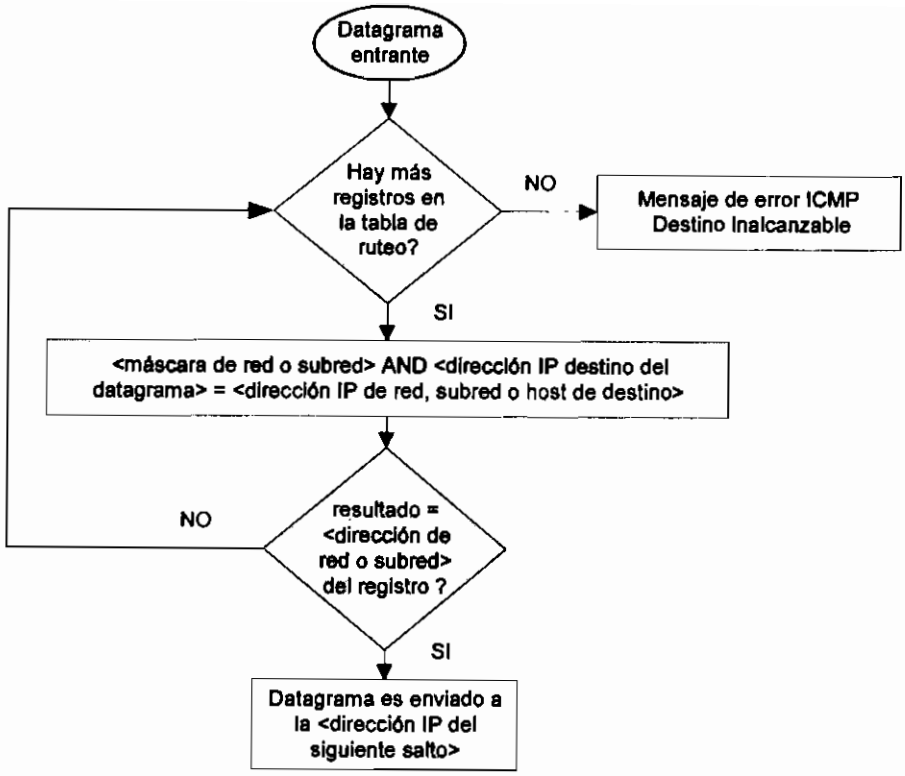


Fig. 1.15. Proceso general de ruteo

Cuando el destino no pertenece a la misma red que el *host* de origen es necesario el uso de ruteadores. El *host* utiliza su tabla de ruteo para determinar la dirección de un ruteador al cual deba enviar sus datagramas para poder ir a la red de destino. Esta dirección se encuentra en el campo dirección al salto siguiente de la tabla de ruteo. Este ruteador no necesariamente tiene que estar conectado directamente a la red de destino; solamente debe ser el siguiente salto en el camino hacia dicha red. El datagrama es enviado al ruteador que está asignado a esa ruta en la tabla. El ruteador, en base a su respectiva tabla de ruteo, se encargará de seleccionar la mejor ruta de transmisión y el tamaño óptimo de los paquetes, en función de la MTU de la trama dependiendo del tipo de red física a la que esté conectado, realizando la fragmentación del datagrama si es necesario. El datagrama viajará de un ruteador a otro hasta encontrar un ruteador que se encuentre directamente conectado a la misma red en la que se ubica el *host* de destino. Este último ruteador hará una entrega directa del datagrama al destino a través de la red local.

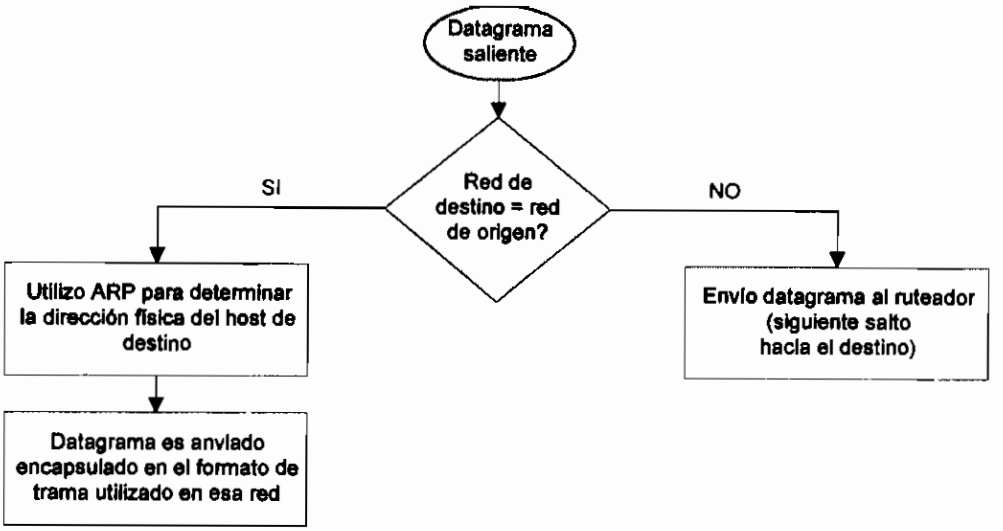


Fig. 1.16. Proceso de ruteo en un *host*

Para reducir el tamaño de las tablas de ruteo en los *hosts*, generalmente se utilizan rutas por defecto, muy efectivo en aquellas redes que poseen un único ruteador para conectarse al resto de redes. En este caso cuando la dirección de destino no pertenece a la red local, el *host* envía el datagrama a un ruteador por defecto. Los ruteadores a diferencia de un *host* no deben utilizar ruteo por defecto. Ellos deben utilizar tablas de ruteo internas lo más completas posibles.

Los ruteadores son capaces de informar a un *host* de origen sobre una decisión errónea de ruteo tomada por el *host*. Si el ruteador conoce de otro ruteador que tenga una mejor ruta hacia el destino, puede hacerle saber mediante un mensaje utilizando el protocolo ICMP.

Cuando un destino no está disponible en la red o no existen registros adecuados en una tabla de ruteo para esa dirección, el ruteador elimina el paquete y envía un mensaje de error usando el protocolo ICMP declarando el destino como inalcanzable.

1.4.3.1 Tipos de Ruteo

Hay dos tipos de ruteo:

- **Estático:** Las rutas son asignadas por el administrador de la red en los ruteadores.
- **Dinámico:** Los ruteadores aprenden dinámicamente las rutas a los distintos destinos empleando protocolos de enrutamiento.

1.4.3.2 Ruteo estático

En el ruteo estático el administrador de red configura: en el ruteador, la dirección a la cual debe enviar los datagramas que no pertenecen a la red de origen, y en los *hosts* de la red, la dirección del ruteador por defecto al cual éstos deben enviar los datagramas que no pertenecen a la red.

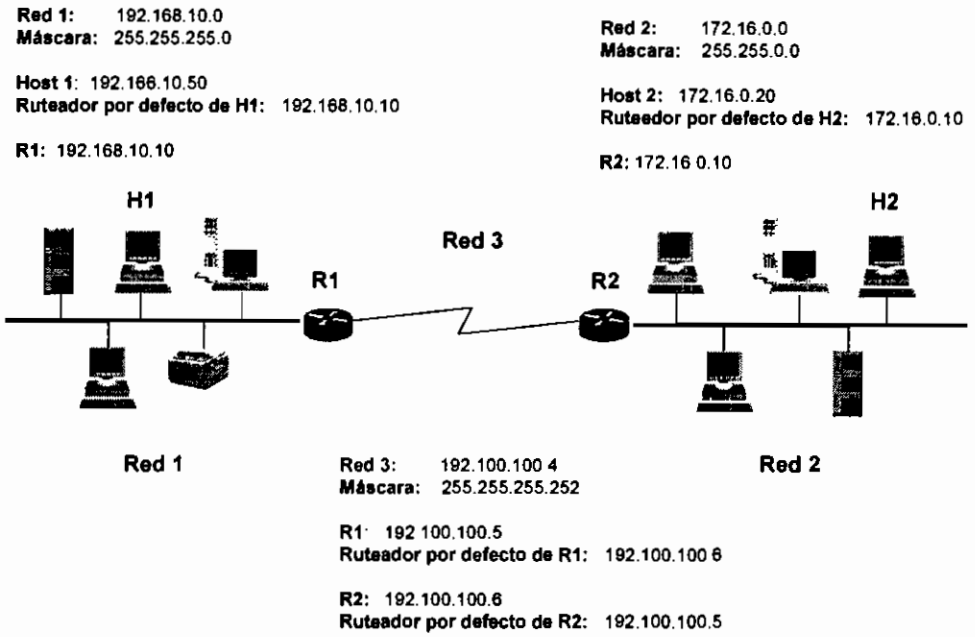


Fig. 1.17. Ejemplo de ruteo estático

La figura 1.17. muestra un ejemplo de ruteo estático, junto con los valores que deberían configurarse en *hosts* y ruteadores ubicados en dos redes que desean comunicarse entre si. Es necesario notar que el enlace entre los ruteadores R1 y R2 constituye una red. Cada ruteador posee una dirección IP por cada red a la que está conectado. El ruteador R1 tiene como ruteador por defecto al ruteador R2 y viceversa.

1.4.3.3 Ruteo Dinámico

El ruteo dinámico se caracteriza por que los ruteadores aprenden dinámicamente las rutas a los distintos destinos empleando protocolos de enrutamiento.

1.4.4 Protocolos de enrutamiento

Un protocolo de enrutamiento no es lo mismo que un protocolo de ruteo. Un protocolo de ruteo, como IP, permite rutear los datagramas. Un protocolo de enrutamiento es simplemente una técnica utilizada por los ruteadores para aprender rutas y mantenerlas actualizadas conforme cambia la red. La función principal de un protocolo de enrutamiento es el intercambio de información de ruteo con otros ruteadores. La información de ruteo se encuentra en las tablas de ruteo. Existen básicamente dos tipos de algoritmos de ruteo en los que se basan los protocolos de enrutamiento:

- Algoritmo vector distancia
- Algoritmo estado de enlace

1.4.4.1 Algoritmo Vector Distancia

En el algoritmo vector distancia cada ruteador mantiene la información de ruteo en su tabla de ruteo, la cual asocia a cada posible destino en la subred de comunicaciones una línea de salida y una distancia hacia ese destino. La distancia se la denomina una métrica y se la mide típicamente en saltos o en retardos de un punto a otro en la subred.

Inicialmente cada ruteador en la subred aprende cuáles son sus vecinos y la distancia a cada uno de ellos. Si la distancia está medida en saltos, la distancia a un vecino será de un salto. Dos ruteadores son vecinos cuando tienen una red en común a la que ambos están conectados.

Cada ruteador debe enviar una copia de su tabla de ruteo a intervalos de tiempo regulares a los ruteadores que se encuentren directamente a su alcance a fin de mantener información de ruteo actualizada. El intercambio de información de ruteo permite a un

ruteador aprender sobre destinos lejanos que son accesibles a través de sus vecinos y la distancia estimada respecto a cada uno de ellos. En base a está información el ruteador puede actualizar su propia tabla de ruteo con los nuevos destinos, la distancia estimada total y la línea de salida para ir a dicho destino. De esta forma el ruteador llega a construir una tabla con todos los destinos posibles en la red y su distancia estimada.

Debido a que el intercambio de información de ruteo es periódica entre vecinos, en el caso de que la información de ruteo recibida indique una ruta más corta a un destino, un destino no existente en la tabla de ruteo o un cambio de distancia a un destino, el ruteador actualizará dinámicamente su tabla de ruteo.

Este algoritmo presenta desventajas, debido a que no converge rápidamente ante cambios en la topología de la red, sea por el aparecimiento de nuevas redes o fallas en las redes existentes, debido a que la información se propaga lentamente provocando que algunos ruteadores mantengan información incorrecta mientras se propaga la información de ruteo. La principal desventaja del algoritmo es que requiere muchos recursos de red ya que debe enviar el contenido completo de sus tablas de ruteo a intervalos regulares de tiempo generalmente cortos. Otra desventaja es que el número de saltos utilizado como métrica no diferencia redes que presenten mejor velocidad de transmisión o confiabilidad. No es lo mismo el paso por un enlace de alta velocidad que a través de un enlace serial de baja velocidad, sin embargo son equivalentes en número de saltos.

El crecimiento de Internet y las desventajas de los protocolos de ruteo basados en el algoritmo vector distancia han hecho que vayan desapareciendo ya que no pueden tener tablas que manejen tantos destinos en la red.

1.4.4.2 Algoritmo Estado de Enlace

El algoritmo de estado de enlace surgió como sucesor del algoritmo vector distancia. El algoritmo funciona de la siguiente forma:

Un conjunto de redes físicas es dividido en un número de áreas. Dentro de cada área un ruteador aprende cuáles son sus vecinos y estima el costo del enlace con cada uno de ellos. El costo de un enlace puede utilizar diferentes métricas: tipo de servicio (retardo, capacidad, confiabilidad), distancia física, ancho de banda, etc. El ruteador construye un paquete que contiene el estado de cada uno de los enlaces con sus vecinos y sus respectivos costos, el cual es distribuido a cada ruteador dentro del área en forma confiable. Cada ruteador dentro del área recibe una copia de un paquete perteneciente a otro ruteador presente dentro de su misma área. De esta forma el ruteador puede construir una representación de toda la topología del área a la que pertenece y calcular la distancia más corta a cada destino dentro de la misma, la cual será la ruta óptima para entregar un datagrama. Conociendo la ruta más corta a cada destino, el ruteador crea la tabla de ruteo para cada tipo de métrica que utilice, en la cual asigna a cada destino con el siguiente salto hacia adelante a lo largo de la ruta más corta.

Cada área posee trayectorias más cortas hacia otras áreas. Un datagrama que deba atravesar varias áreas será manejado en forma independiente dentro de cada área, pero siempre utilizará la ruta más corta a través de ellas.

Los protocolos de estado de enlace, a diferencia de los protocolos vector distancia, únicamente envían actualizaciones del estado de sus enlaces cuando algún evento inesperado ha ocurrido en la red. También pueden enviar actualizaciones regulares como una forma de determinar si una conexión con su vecino permanece activa, el vecino deberá responder con un acuse de recibo. Si la información de ruteo recibida indica cambios en la topología de la red, el algoritmo dinámicamente recalcula las rutas más cortas a los distintos destinos. Los protocolos de estado de enlace, a diferencia de los protocolos basados en el algoritmo vector distancia, usan un menor número de recursos de red ya que únicamente intercambian información de actualización del estado de sus enlaces y no de sus tablas de ruteo completas. Estos algoritmos, sin embargo, requieren mayor cálculo computacional en cada ruteador. En cambio, los usuarios obtienen respuesta mas rápida a eventos de red, convergencia más rápida y acceso a más avanzados servicios de red.

En Internet se define el concepto de sistema autónomo (AS - *Autonomous System*). Un sistema autónomo es un conjunto de redes y ruteadores controlados por una sola autoridad (compañía, institución, etc.). Se puede decir que Internet es un conjunto de sistemas autónomos interconectados. Un sistema autónomo a su vez puede estar dividido en áreas más pequeñas.

Los protocolos de enrutamiento usados dentro de un sistema autónomo y fuera de un sistema autónomo son distintos. Un protocolo de enrutamiento utilizado dentro de un sistema autónomo se denomina Protocolo de Ruteador Interior (IGP - *Interior Gateway Protocol*). Un protocolo de enrutamiento entre sistemas autónomos se denomina Protocolo de Ruteador Exterior (EGP - *Exterior Gateway Protocol*).

Hay algunos protocolos de enrutamiento comúnmente asociados con el protocolo de ruteo IP y el Internet, entre ellos: RIP, IGRP, OSPF y BGP.

1.4.4.3 *Interior Gateway Protocols* (IGPs) : RIP, IGRP, OSPF

RIP, IGRP, y OSPF son básicamente utilizados como protocolos de enrutamiento dentro de un sistema autónomo particular, tal como dentro de una red de una corporación o dentro de la red de un proveedor de servicios de Internet (ISP). Por lo tanto RIP, IGRP, y OSPF son protocolos de ruteadores internos (IGP).

El Protocolo de Información de Enrutamiento (RIP - *Routing Information Protocol*) describe cómo los ruteadores intercambiarán la información de las tablas de ruteo usando un algoritmo vector distancia. Con RIP, los ruteadores vecinos periódicamente intercambian sus tablas de ruteo enteras o partes de sus tablas. RIP utiliza cuenta de saltos como la métrica del costo de una ruta y una ruta está limitada a máximo 16 saltos. Además RIP establece que cada 30 segundos debe enviar a sus vecinos una copia completa de su tabla de ruteo. RIP define otros contadores de tiempo. Uno de ellos permite al ruteador declarar inválida una ruta con un ruteador vecino cuando este último no ha enviado mensajes de actualización de su tabla durante 90 segundos. Otro contador indica el tiempo que el ruteador, después de invalidar una ruta, debe hacer caso omiso a información que le proporcionen sus vecinos acerca de rutas alternas que permitan

llegar al destino declarado por él como inalcanzable. Esto permite que la información de una ruta inválida se propague por toda la red, evitando que se tomen decisiones erróneas de ruteo. Un valor típico para este contador es de 270 segundos. [10] Desafortunadamente, RIP ha llegado a ser ineficiente en el Internet debido a su rápido crecimiento, lo cual hace que el límite de 16 saltos resulte pequeño.

El Protocolo de Enrutamiento de Gateway Interior (IGRP - *Interior Gateway Routing Protocol*) está basado en el algoritmo vector distancia. IGRP utiliza como criterio de decisión para la creación de las tablas de ruteo, la combinación de métricas: retardo de propagación, ancho de banda, confiabilidad y la carga.

IGRP permite asignar múltiples rutas hacia un mismo destino. Si una ruta principal no está disponible se usará alguna de las secundarias. IGRP usa los mismos contadores que RIP, pero con distintos tiempos. IGRP envía la información de su tabla de ruteo a sus vecinos cada 90 segundos. El tiempo para declarar una ruta como inválida se establece en 270 segundos, y el tiempo de espera de un ruteador antes de recibir notificaciones de rutas alternas hacia un destino declarado por él como inalcanzable se establece en 280 segundos. [10]

El protocolo abierto de la primera ruta más corta (OSPF - *Open Shortest Path First*) es un protocolo de enrutamiento basado en el algoritmo de estados de enlace. OSFP es más robusto que RIP, converge más rápido, requiere menos ancho de banda, y es mejor para trabajar con redes más grandes. A OSPF se le denomina protocolo abierto, ya que su especificación está al alcance de cualquier persona que revise el RFC 1247 que lo describe. Como se mencionó anteriormente un sistema autónomo puede dividirse a su vez en un número de áreas más pequeñas. Un área es un grupo de redes contiguas y sus *hosts*. Cada ruteador dentro de un área mantiene una tabla que contiene el estado de los enlaces de los ruteadores dentro del área y que le permite describir la topología completa del área y calcular la ruta más corta hacia cada destino dentro de la misma.

Existen ruteadores que poseen interfaces con varias áreas, a los cuales OSPF los denomina ruteadores de borde de área. Estos ruteadores poseen una tabla que contiene la topología de cada área a la que están conectados. El conjunto de todos los ruteadores

de borde de área forman lo que se denomina el área de *backbone* dentro del AS o *backbone* intra-AS. Cuando un paquete debe ser enviado de una área a otra (ruteo inter-área), debe ser primeramente enviado al *backbone*. El *backbone* se encargará de rutear el paquete del área origen al área de destino. El área de destino ruteará internamente el paquete hacia el *host* de destino final. Cuando el origen y destino están dentro de la misma área se denomina ruteo intra-área.

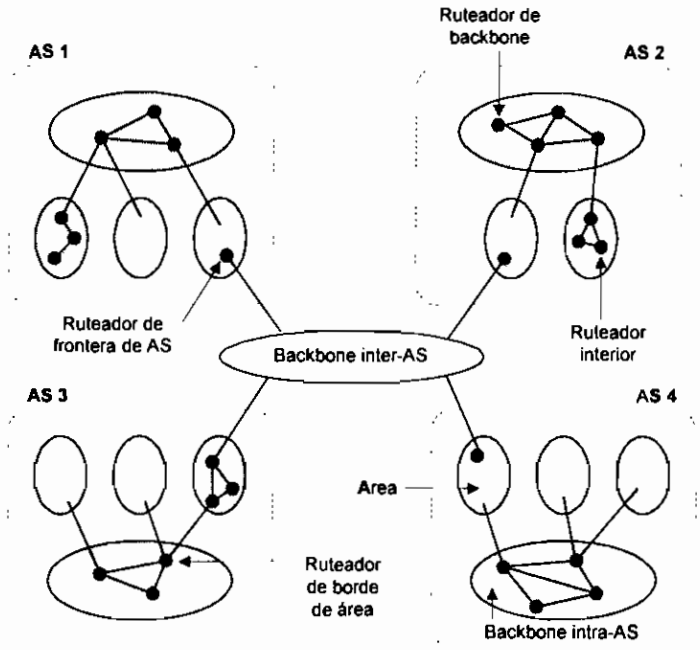


Fig. 1.18. Sistemas autónomos, áreas, *backbones* en OSPF [1]

OSPF es el protocolo de enrutamiento encargado de distribuir información de ruteo dentro de cada área. Se debe recordar que el *backbone* intra-AS es en si un área. OSPF utiliza como métrica el tipo de servicio (retardo de propagación, desempeño, confiabilidad) solicitado en la cabecera del datagrama IP y crea tablas de ruteo dentro de cada área basadas en cada uno de estos criterios. Cada ruteador dentro de un área periódicamente envía en forma de *broadcast* el estado de sus enlaces. El estado de los enlaces es también enviado cuando surgen cambios imprevistos en un enlace. En ambos casos OSPF solamente envía en forma de *broadcast* los cambios en los estados de sus enlaces antes que de toda la tabla de ruteo. OSPF versión 2, está rápidamente reemplazando a RIP e IGRP en el Internet.

OSPF describe mecanismos para descubrir nuevos ruteadores e intercambiar información del estado de sus enlaces. OSPF además permite optimizar el ruteo en redes multiacceso, las cuales son aquellas que poseen varios ruteadores. OSPF en este tipo de redes designa uno solo de todos los posibles ruteadores existentes como ruteador principal y otro como ruteador secundario de respaldo. Unicamente el ruteador principal es el encargado de enviar información del estado de los enlaces dentro de la red multiacceso, evitando que todos los ruteadores de esta red envíen la misma información a otro ruteador. Los ruteadores que intercambian información de ruteo en OSPF se denominan ruteadores adyacentes. En casos de redes multiacceso los ruteadores adyacentes no necesariamente pueden ser vecinos.

1.4.4.4 *Exterior Gateway Protocols* (EGPs) : BGP

OSPF denomina a los ruteadores que permiten ir hacia otros sistemas autónomos como ruteadores de frontera del sistema autónomo. OSPF no permite intercambio de información de ruteo entre sistemas autónomos. En este caso se utiliza el protocolo BGP. El protocolo BGP-4 (*Border Gateway Protocol version 4*) es un protocolo de ruteadores exteriores utilizado para proveer información de ruteo entre distintos sistemas autónomos en Internet. BGP es un protocolo vector distancia, como RIP, pero a diferencia de casi todos los otros protocolos de este tipo, las tablas BGP almacenan la ruta completa hacia un destino; es decir, contiene todos los ruteadores por los cuales debe circular y no únicamente el ruteador del siguiente salto. BGP- 4 también soporta ruteo basado en políticas, es decir, permite que el ruteo se base en: asuntos políticos, de seguridad, legales o económicos, antes que basarse enteramente en asuntos técnicos.

BGP dependiendo de la política de ruteo, decidirá si pasa por una ruta que contiene ruteadores con políticas opuestas o utiliza rutas alternativas.

Finalmente es necesario aclarar que un mensaje RIP es transportado en un datagrama de usuario UDP el cual a su vez es llevado en un datagrama IP. Un mensaje IGRP en cambio se encapsula en un segmento TCP sobre IP. Un mensaje OSPF, por el otro lado, es llevado directamente en un datagrama IP, e identificado con el código 89 en el campo Protocolo de la cabecera del datagrama. OSPF emplea acuses de recibo y corrección de

errores para brindar una transferencia confiable. Los mensajes BGP, son llevados en forma confiable en segmentos TCP sobre IP. [G]

1.4.5 *Internet Control Message Protocol* (ICMP)

En la capa internet se define a más de IP, el Protocolo de Mensajes de Control Internet (ICMP - *Internet Control Message Protocol*), el cual es un protocolo utilizado por ruteadores para enviar mensajes de error y de control hacia otros ruteadores o *hosts*. ICMP también es utilizado por un *host* para determinar si un destino es accesible o no.
Los mensajes ICMP normalmente actúan en la capa IP o en la capa de protocolos más altos como: TCP o UDP. Es decir un mensaje ICMP es procesado por el *software* de capa internet, informando a procesos de capas superiores en caso de que ellos causaran un problema.

Los mensajes más comunes de ICMP son:

- **Destino inalcanzable:** Indica que un datagrama no pudo ser entregado debido a que el *host* de destino no fue alcanzado: ya sea porque el *host* o red de destino es inalcanzable o desconocido, el protocolo o puerto utilizado es desconocido, se requiere de fragmentación en un datagrama donde el bit DF está activado o finalmente debido a que la red o *host* puede ser inalcanzable por el tipo de servicio especificado en el datagrama IP.
- **Eco y respuesta al eco:** Se utiliza estos dos mensajes para determinar si un destino es accesible o alcanzable en la red. Un *host* envía una solicitud de eco a un destino conteniendo un campo de datos opcional. El *host* de destino debe responder al eco con un paquete conteniendo la misma información enviada en el pedido de eco. Se conoce como comando *Ping* a una solicitud y respuesta de eco.
- **Problema de parámetros:** Indica que un ruteador o algún *host* encontró algún parámetro erróneo en la cabecera de un datagrama.
- **Redireccionamiento:** Es utilizado por un ruteador para informar a un *host* de origen sobre decisiones erróneas de ruteo tomadas por el *host* e informarle sobre mejores rutas para enviar los datagramas.

[G] Ver Anexo C.

- **Tiempo de vida excedido:** Este mensaje indica que un datagrama ha sido eliminado debido a que el tiempo de vida ha terminado, o debido a que un paquete fragmentado no fue recibido completo.
- **Solicitud de sello de tiempo:** Son mensajes que solicitan a un ruteador remoto envíe en una respuesta, su hora local. El mensaje de respuesta contiene información que permite al ruteador de origen determinar el tiempo que le toma al sistema remoto almacenar y procesar datagramas. Además permite al ruteador de origen estimar la diferencia entre la hora local y la hora de un ruteador remoto.
- **Solicitud de máscara de subred:** Permite a un *host* enviar un pedido a un ruteador solicitándole le informe cuál es la máscara de subred utilizada en esa red. Este pedido generalmente se lo envía por difusión.

Los mensajes ICMP son enviados encapsulándolos en un datagrama IP. Los mensajes ICMP se especifican en el datagrama IP con un valor de protocolo igual a 1. El formato de un mensaje ICMP es el siguiente:

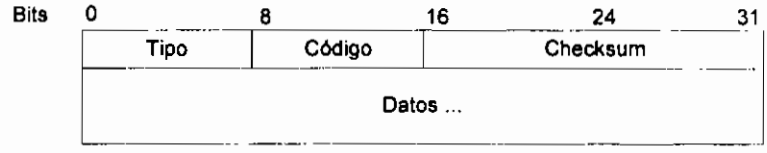


Fig. 1.19. Formato de un paquete ICMP [2]

La cabecera del mensaje ICMP contiene el tipo y código del error que provocó el mensaje ICMP. El procesamiento e interpretación del mensaje depende del tipo de mensaje. Además la cabecera contiene un campo de Suma de Verificación (*checksum*) para detectar errores sobre el mensaje ICMP, y un campo de datos que lleva la información del mensaje ICMP, cuyo contenido varía según el tipo de mensaje ICMP.

Los mensajes ICMP relacionados con errores, siempre contienen la cabecera del datagrama IP y los primeros 8 bytes del campo de datos del datagrama IP que provocó dicho mensaje de error ICMP. En estos 8 bytes está contenida parte de las cabeceras añadidas por los protocolos TCP o UDP. La información anterior permite especificar qué protocolo y proceso de usuario debe procesar el mensaje de error ICMP.

Existen dos aplicaciones que utilizan directamente el protocolo ICMP: *Ping* y *Traceroute*.

Ping permite determinar si un destino es accesible o no. Se basa en la petición y respuesta de eco descrita anteriormente. Si el destino es inaccesible no se podrá intercambiar paquetes con él. *Ping* también permite medir el tiempo que se tarda en alcanzar un *host* y nos da idea de cuan lejos está.

Traceroute permite determinar la ruta que un datagrama sigue desde el origen a un destino. El programa envía un datagrama IP conteniendo un datagrama de usuario UDP en el campo de datos. El programa se encarga de hacer que un ruteador por el que pase el datagrama incremente en una unidad el valor del campo TTL de la cabecera del datagrama IP. Inicialmente se envía un datagrama con el campo TTL igual a 1 y un parámetro no válido en la cabecera de un paquete UDP enviado en el campo de datos. Al llegar al primer ruteador éste decrementa el TTL como es usual y debido a que TTL es cero, envía un mensaje ICMP de tiempo de vida excedido al origen, posteriormente el programa *Traceroute* obliga a incrementar el TTL en una unidad, lo cual hace que el datagrama pueda ser enviando al siguiente ruteador, el que, a su vez, repite el proceso enviando un mensaje de tiempo de vida excedido y así sucesivamente hasta llegar al destino. El destino, a su vez, envía un mensaje ICMP de error debido al parámetro inválido en la cabecera UDP. Ya que cada mensaje ICMP viene encapsulado en un datagrama IP, el cual contiene la dirección del ruteador o *host* que envió el mensaje, es posible determinar la ruta seguida por el datagrama.

1.4.6 *Internet Group Management Protocol* (IGMP)

El *multicasting* es la transmisión de un mismo datagrama IP a un grupo de *hosts*, identificado por una única dirección IP clase D. El datagrama IP es entregado a todos los miembros del grupo en forma no confiable y con el mejor esfuerzo.

El *multicasting* ha permitido el aparecimiento de nuevas aplicaciones tales como: las vídeo conferencias, envío simultáneo de mensajes a varios destinos, etc.

La membresía de un *host* a un grupo *multicast* es dinámica, es decir un *host* puede unirse y dejar un grupo en cualquier instante. No existe restricciones para el número de *hosts* que pueden pertenecer a un grupo de *multicast*.

Un grupo *multicast* puede ser permanente o temporal. Un grupo permanente siempre existe, aun sin miembros, y tiene una dirección IP clase D permanente. Los grupos temporales pueden crearse dinámicamente y poseen direcciones asignadas generalmente en forma manual o especificadas según la aplicación. Los grupos temporales solo existen cuando poseen miembros.

El *multicasting* puede utilizarse en una sola red física o a través de una red de redes. En el último caso se requiere el uso de ruteadores especiales denominados ruteadores *multicast*. Un ruteador *multicast* se encargará de contactar otros ruteadores y establecer rutas que le permitan entregar los datagramas IP de *multicast* a otros miembros del grupo ubicados en otras redes.

Una forma de limitar la difusión de un datagrama de *multicast* únicamente dentro de un red local es emplear el campo TTL de la cabecera del datagrama igual a 1. De esta forma el ruteador *multicast* se verá impedido de propagar el datagrama a otras redes. No se generan mensajes de error ICMP para paquetes IGMP.

El Protocolo de Gestión de Grupo Internet (IGMP - *Internet Group Managenement Protocol*) es un protocolo usado por *hosts* y ruteadores que soportan envíos *multicasting*. El protocolo IGMP permite a un *host* informar a un ruteador *multicast*, el deseo de unirse a un grupo de *multicast* a fin de recibir paquetes enviados a ese grupo. Dependiendo del grupo, el *host* podrá o no enviar mensajes. Un *host* puede unirse y dejar un grupo de *multicast* en forma dinámica. Además un *host* puede tener varios procesos que formen simultáneamente parte de distintos grupos de *multicast* a través de una misma interfaz de red. Además un mismo proceso puede unirse a un mismo grupo *multicast* desde distintas interfaces de red.

El protocolo IGMP permite a su vez a los ruteadores *multicast* saber a qué grupos de *multicast* están afiliados algunos de los *hosts* de la red en la cual el ruteador está

conectado. Esta información permite al ruteador saber si debe difundir un datagrama direccionado a un grupo de *multicast* en su red o simplemente desecharlo si no existe ningún *host* en su red que esté afiliado a ese grupo.

Un mensaje IGMP se transmite encapsulado en un datagrama IP. Los mensajes IGMP se especifican en el datagrama IP con un valor de protocolo igual a 2. El formato del mensaje IGMP es el siguiente:

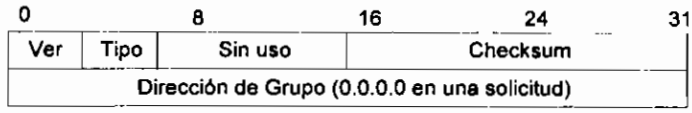


Fig. 1.20. Formato de un datagrama IGMP [2]

Un mensaje IGMP posee un tamaño fijo sin datos opcionales.

- **Ver:** Indica la versión del protocolo IGMP. La versión actual de IGMP es 1.
- **Tipo:** Identifica si es un mensaje IGMP de solicitud (1) o de reporte (2).
- **Suma de Verificación (*checksum*):** Permite detectar errores sobre el mensaje IGMP.
- **Dirección de grupo:** En una solicitud de un ruteador *multicast* a los *hosts*, el campo de dirección de grupo es 0, y en un reporte de un *host* contiene la dirección de grupo a la que su proceso pertenece.

Un ruteador *multicast* periódicamente envía un mensaje de solicitud en cada una de las redes a las que está conectado para determinar si existen *hosts* que en ese momento formen parte de un grupo *multicast*. El mensaje de solicitud IGMP es enviado en un datagrama IP que posee TTL = 1 y dirección de destino 224.0.0.1. Esta dirección IP representa un grupo *multicast* permanente al cual pertenecen por defecto todos los *hosts* de la red local.

Un *host* responde a un mensaje de solicitud enviando un mensaje de reporte por cada uno de sus procesos que en ese momento formen parte de un grupo *multicast*. El mensaje de reporte IGMP contiene la dirección IP del grupo *multicast* en el campo

Dirección de grupo y viaja en un datagrama IP que posee como dirección de destino la dirección IP del grupo de *multicast*.

Puesto que varios *hosts* de la misma red pueden pertenecer a un mismo grupo *multicast*, cuando un *host* ya ha enviado un mensaje de reporte correspondiente a dicho grupo, el resto de *hosts* no envía más.

Cuando un *host* desea unirse a un grupo en una interfaz, envía un mensaje de reporte conteniendo la dirección clase D del grupo de difusión al cual quiere unirse, en un datagrama con dirección de destino igual a la dirección del grupo *multicast*. El ruteador de *multicast* recibe el reporte y activa una bandera para indicar que al menos existe un *host* en esa red que pertenece a ese grupo.

En redes locales en que no existan ruteadores *multicast*, únicamente existirán paquetes IGMP de reporte cuando un *host* se una a un grupo o desee crear uno nuevo.

El Protocolo de Ruteo de Multidifusión Vector Distancia (DVMRP - *Distance Vector Multicast Routing Protocol*), es un complejo protocolo de enrutamiento, no estandarizado, utilizado por ruteadores de multidifusión para intercambiar su información de ruteo. Se basa en el algoritmo vector distancia. Intercambiando esta información, los ruteadores *multicast* pueden establecer un conjunto de rutas para poder entregar los datagramas de multidifusión a todos los miembros de un grupo. El ruteo de multidifusión, a través de redes que no soporten multidifusión por *hardware*, se lo hace mediante la técnica de *tunneling* [2], en la cual un datagrama direccionado a una dirección de multidifusión es encapsulado en un datagrama IP de unidifusión, el cual atraviesa la red hasta el siguiente ruteador fuera de la red. De esta forma se puede atravesar redes que no permiten multidifusión hasta llegar al destino.

El uso de difusión es otra forma de soportar multidifusión en redes que no soportan *multicast* a nivel de capa física.

En redes que soporten *multicast* (Ejm. Ethernet), la transformación de direcciones IP de *multicast* en direcciones de *hardware* tipo *multicast*, depende del tipo de red específica.

1.5 PROTOCOLOS DE CAPA TRANSPORTE

El modelo de referencia TCP/IP, como se mostró en la figura 1.2, define tres tipos de servicios relacionados entre sí. El nivel inferior ofrece un servicio de entrega de paquetes sin conexión, no confiable. Sobre este nivel ofrece el servicio de transporte extremo a extremo o *host-host,* que es la base para los servicios de aplicación.

A nivel de cada transporte se define la abstracción denominada puerto de protocolo. Un puerto de protocolo es un punto abstracto entre capa aplicación y capa transporte, a través del cual un programa de aplicación recibe y envía mensajes a capa transporte. Un puerto de protocolo está asociado con un número de puerto, el cual es un número de 16 bits, utilizado por el protocolo de capa transporte para identificar a qué protocolo de capa superior o programa de aplicación (proceso de usuario) debe ser entregado un mensaje.

El número de puerto y la dirección IP juntas forman lo que se denomina un *socket* o dirección de capa transporte. La comunicación extremo a extremo entre dos *hosts* es identificada de forma única por el conjunto:

<protocolo, dirección IP de origen, puerto de origen, dirección IP de destino, puerto de destino>

Un programa de aplicación debe negociar con el sistema operativo la asignación de un puerto de protocolo y un número de puerto, a través del cual podrá enviar y recibir sus mensajes.

El *software* de transporte en el *host* destino comprobará el número del puerto de destino del mensaje para ver si coincide con alguno de los puertos que están en uso. Si no corresponde a ningún puerto existente en ese momento, envía un mensaje de error al *host* de origen utilizando el protocolo ICMP, reportando el destino como inalcanzable ya que el puerto es desconocido. Caso contrario pone el mensaje en una cola en memoria intermedia asignada al puerto, donde podrá ser accedido por el programa de aplicación. Si el puerto está lleno, el mensaje se descarta.

Existen dos formas de asignar un número de puerto a un programa de aplicación (proceso). Esta asignación de número de puerto debe ser conocida por ambos extremos de la comunicación.

1. Usar una autoridad central que se encargue de asignar los puertos y establecer una lista de puertos asignados. En este caso todo el *software* de capa aplicación se elabora de acuerdo con esta lista. A este tipo de puertos se los conoce como puertos bien conocidos y son utilizados por servicios o aplicaciones de red estandarizadas. Los puertos bien conocidos son controlados y asignados por la *Internet Assigned Numbers Authority* (IANA)

2. Usar una asignación dinámica, en la cual los puertos no son conocidos globalmente. Cuando un programa necesita un puerto, el *software* de red, generalmente en el sistema operativo, le asigna uno. Para conocer la asignación de puertos en otra máquina es necesario enviar una petición sobre cómo acceder a un determinado programa de aplicación.

El modelo TCP/IP utiliza ambas formas de asignación de número de puerto. Los puertos bien conocidos utilizados generalmente por programas servidores tales como FTP, TELNET, SMTP, DNS, etc. poseen números de puerto fijos comprendidos entre 0 y 1023. Los puertos entre 1024 y 65535 no son controlados por IANA y pueden ser asignados dinámicamente a los programas clientes o programas de aplicación realizados por un usuario. [2]

Un mismo *socket* puede ser utilizado por múltiples conexiones de red simultáneamente. Generalmente esto ocurre en un modelo cliente - servidor, donde varios clientes hacen uso de una misma aplicación ejecutándose en un servidor.

1.5.1 *User Datagram Protocol* (UDP)

El Protocolo de Datagramas de Usuario (UDP - *User Datagram Protocol*) es un protocolo de transporte no orientado a conexión, no confiable. UDP envía los datos de un programa de aplicación de un *host* a otro programa de aplicación de otro *host*, sin

garantizar la correcta recepción de los mismos. UDP añade una cabecera a los datos de la aplicación para formar su mensaje, denominado datagrama de usuario, el cual a su vez es encapsulado en el campo de datos de un datagrama IP, para su entrega. Debido a que UDP es un protocolo no orientado a conexión y no confiable, los datagramas de usuario enviados pueden perderse, duplicarse o llegar en desorden al destino. Por esta razón, la capa aplicación se deberá encargar de los detalles correspondientes para solucionar estos problemas.

Un datagrama de usuario UDP está formado por una cabecera y un área de datos. Su formato se presenta a continuación:

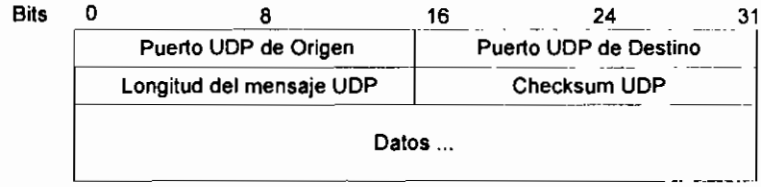


Fig. 1.21. Formato de un datagrama de usuario UDP [2]

- **Puerto de Origen:** Valor del puerto UDP de origen.
- **Puerto de Destino:** Valor del puerto UDP de destino.
- **Longitud del mensaje UDP:** Contiene la longitud total en octetos del datagrama de usuario UDP.
- **Suma de verificación (*checksum*):** Es un campo opcional utilizado para determinar errores en los datos y capaz de reconocer si un datagrama de usuario llegó a su destino correcto y puerto correcto, ya que en el cómputo del *checksum* a más de los campos del datagrama de usuario UDP se utiliza un pseudoencabezado formado por: la dirección IP de origen, dirección IP de destino, tipo de protocolo y longitud del mensaje UDP sin pseudoencabezado.

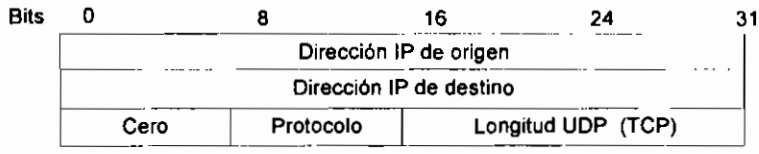


Fig. 1.22. Pseudoencabezado UDP (TCP)

Los programas de aplicación que utilicen UDP como protocolo de transporte utilizan como puerto de protocolo un puerto UDP. Cada puerto UDP tiene asociado un número de puerto según la aplicación que lo utilice. La cabecera contiene el número de puerto UDP de origen y el número de puerto UDP de destino. Esto permite a UDP identificar los posibles múltiples programas de aplicación que pueden estarse ejecutando en una máquina en un determinado instante. Debido a que cada programa de aplicación puede ser completamente identificado por su número de puerto, UDP es capaz de multiplexar varias comunicaciones de transporte utilizadas por procesos de aplicación entre un *host* de origen y un *host* de destino que se estén ejecutando simultáneamente. En el *host* destino se demultiplexan las comunicaciones de transporte en función del número de puerto asociado a una aplicación a la que corresponde el datagrama de usuario.

1.5.2 *Transmission Control Protocol* (TCP)

Internet utiliza a nivel de capa internet, un servicio de entrega de datagramas no confiable, implementado por el protocolo IP. Un servicio no confiable implica que los datagramas enviados pueden perderse, llegar dañados, duplicados, o en desorden al destino, debiendo las capas superiores implementar mecanismos de control a fin de solucionar estos problemas.

El Protocolo de Control de Transmisión (TCP - *Transmission Control Protocol*) es un protocolo de transporte orientado a conexión, encargado de garantizar una comunicación extremo a extremo confiable, libre de errores y en una secuencia correcta, para que sea utilizada por los programas de capa aplicación que requieran este servicio. Es decir TCP se encarga de ocultar las imperfecciones de la red. TCP no asume confiabilidad en los protocolos de bajo nivel como IP. La mayoría de aplicaciones en Internet operan sobre el servicio de transporte confiable dado por TCP.

TCP se encarga de proporcionar a los procesos de capa aplicación:

- **Transferencia continua del flujo de datos:** Los programas de aplicación asumen que TCP transfiere sus datos en forma de un flujo de datos continuo a través de la red. Sin embargo, TCP se encarga de dividir la información entregada por los

procesos de capa aplicación en paquetes denominados segmentos, los cuales son entregados al protocolo IP para que éste los transmita individualmente en forma de datagramas hacia el destino. A su vez, un datagrama puede ser dividido en fragmentos en un ruteador a lo largo del viaje para ajustarse a la MTU de la red física. En el destino, IP reconstruye el datagrama original a partir de los fragmentos y pasa el segmento contenido en su campo de datos a la capa transporte. A continuación TCP se encarga de reconstruir el flujo de datos original a partir de los segmentos recibidos antes de entregar al programa de aplicación respectivo para su procesamiento.

TCP se encarga de negociar el tamaño máximo de segmento a transferirse entre los extremos.

TCP puede almacenar los datos enviados por capa aplicación hasta juntar un tamaño considerable a fin de incrementar la eficiencia de transmisión. También es capaz de aceptar pedidos de envío inmediato, los cuales no deben ser almacenados, mediante una opción de empuje (*push*). TCP también permite enviar datos urgentes, deteniendo la acumulación de datos y transmitiendo inmediatamente todo lo que tenía almacenado junto con el mensaje urgente.

- **Confiabilidad:** TCP es un protocolo confiable, es decir, se encarga de asegurar que los datos transmitidos por capa aplicación sean recibidos en el destino en forma correcta. TCP para brindar confiabilidad asigna números de secuencia a cada byte de datos en los segmentos enviados, utiliza sumas de verificación (*checksum*) para realizar control de errores, emplea acuses de recibo para reconocer que segmentos han sido recibidos y reenvía información en caso de pérdida o daño de los segmentos enviados.

  TCP asigna un número de secuencia a cada byte que transmite un segmento. La cabecera de un segmento TCP contiene el número de secuencia del primer byte de los datos enviados en el segmento. El número de secuencia permite al protocolo TCP en el destino, reordenar los segmentos recibidos en el caso en que lleguen en desorden y eliminar los segmentos duplicados. El protocolo TCP en el destino

utiliza acuses de recibo para informar al origen el siguiente número de secuencia que espera recibir e indicar que todos los números de secuencia inferiores a éste han sido recibidos correctamente.

TCP utiliza el mecanismo de ventana deslizante para aprovechar el ancho de banda de una red. El mecanismo de ventana deslizante permite al *host* de origen enviar un determinado número de segmentos consecutivamente, lo que se denomina una ventana, sin necesidad de esperar un acuse de recibo entre cada envío de un segmento. Por cada segmento enviado comprendido en la ventana, el *host* de origen inicia un contador de tiempo. El *host* de destino también utiliza una ventana deslizante para determinar los segmentos que espera recibir y enviar el respectivo acuse.

Para el envío de acuses de recibo, TCP utiliza un esquema acumulativo, es decir, tras enviar los segmentos comprendidos en una ventana, el destino no enviará un acuse por cada segmento, sino únicamente un acuse que represente a todos los segmentos que llegaron correctamente y en orden, el cual contendrá el número de secuencia del siguiente byte que espera recibir. Es decir un único acuse de recibo es el acuse de recibo de uno, varios o todos los segmentos de la ventana.

Si el acuse de recibo de un segmento llega antes de que el tiempo de su contador expire, TCP cancela el contador asociado a ese segmento y procede a desplazar la ventana al segmento determinado por el número de secuencia que el receptor espera recibir, indicado en el acuse de recibo. De esta forma la ventana se va desplazando con cada acuse de recibo que llega, pudiendo enviar otros segmentos que estaban almacenados y que ahora están dentro de la ventana.

Si después de un tiempo dado no se ha recibido el acuse de recibo de un segmento de la ventana, se reenvía el segmento, pues TCP asume que se perdió, se dañó o el acuse de recibo se perdió.

El reenviar un segmento debido a un acuse de recibo perdido puede hacer que al destino lleguen segmentos duplicados. Si el número de secuencia de un segmento se

repite en el destino se elimina el duplicado.

- **Control de flujo:** TCP permite al *host* de destino enviar en el mismo acuse de recibo, en el cual indica el número de secuencia del siguiente byte que espera recibir, el número de bytes que en ese momento puede almacenar en sus *buffers* internos. A este tamaño máximo de almacenamiento se denomina ventana de recepción. Esta información permite al *host* de origen ajustar el tamaño de su ventana dinámicamente a fin de que no vaya a saturar al destino con el envío de más segmentos de los que pueda procesar.

  Además TCP define una ventana de congestionamiento, la cual estima el máximo tamaño de la ventana de transmisión que se puede enviar por la red sin que se produzca congestionamiento. Esta ventana permite evitar que un tamaño excesivo de la ventana de transmisión en una red congestionada, incremente la demora en la entrega de los segmentos y provoque que los tiempos de los contadores terminen causando retransmisión de los segmentos e incrementando aun más la congestión.

  El tamaño máximo de la ventana de transmisión, es decir el número de segmentos que pueden enviarse consecutivamente, vendrá dado por el mínimo tamaño entre las ventanas de recepción y congestión.

- **Circuitos virtuales:** TCP es un protocolo orientado a conexión, esto significa que se encarga de establecer, mantener y finalizar una conexión utilizada por una comunicación entre procesos de usuario (programas de aplicación). TCP para brindar el servicio orientado a conexión define la abstracción de circuito virtual. Un circuito virtual se denomina así, ya que los procesos de usuario que utilizan TCP como transporte asumen que se comunican directamente entre sí, como si estuviesen conectados mediante un único medio físico (circuito), el cual actúa como un tubo, en el que el origen coloca sus datos y el receptor los recoge en el mismo orden.

  A nivel de red, IP no establece circuitos por los cuales enviar sus datagramas. No hay fase de establecimiento ni liberación de una conexión; cada uno de los datagramas es tratado de forma independiente por la red y pueden ser encaminados

de forma distinta, pero TCP se encarga de ordenarlos en el nodo destino. Así, TCP logra que la aplicación crea que utiliza un circuito físico (semejante a un tubo) entre ambos extremos.

TCP identifica una comunicación entre procesos identificando el circuito virtual que utiliza. Un circuito virtual se identifica en base a los puntos extremos de la conexión. El punto extremo de una conexión es un *socket*, es decir, el conjunto de la dirección IP de la interfaz de red y el número de puerto utilizado por el proceso. Por lo tanto un circuito virtual se identifica mediante el conjunto: <Protocolo, Dirección IP de origen, número de puerto de origen, Dirección IP de destino, número de puerto de destino>. En este caso los puertos utilizados por los procesos son puertos TCP, cada uno de los cuales tendrá un número de proceso asociado según la aplicación.

TCP define números de puertos bien conocidos utilizados por procesos de aplicación estandarizados y números de puertos que se pueden asignar dinámicamente.

- **Multiplexaje:** TCP permite multiplexar varias conexiones de transporte utilizadas por procesos de capa aplicación ejecutándose simultáneamente en una misma máquina. La multiplexación es posible ya que dos procesos se comunican utilizando un circuito virtual, el cual está completamente identificado por los *sockets* en cada punto extremo. Un mismo *socket* puede ser utilizado por varios circuitos virtuales simultáneamente. Por ejemplo los procesos de servidores son capaces de manejar múltiples comunicaciones a través de un mismo *socket*.

- **Comunicación *Full Duplex*:** TCP proporciona comunicación bidireccional mediante circuitos virtuales.

Un segmento TCP está formado por una cabecera y un área de datos. Su formato se presenta a continuación:

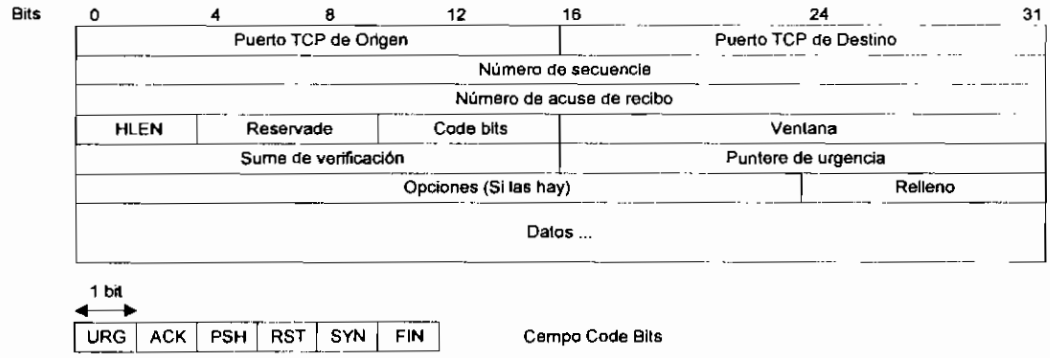


Fig. 1.23. Formato de un segmento TCP [1]

La cabecera se compone de los siguientes campos:

- **Puerto Origen:** Número del puerto TCP que identifica al programa de aplicación origen.
- **Puerto Destino:** Número del puerto TCP que identifica al programa de aplicación destino.
- **Número de secuencia:** Indica el número de secuencia del primer byte de datos contenido en el campo de datos. Es utilizado por el destino para ubicar el segmento en su lugar adecuado dentro del flujo de datos enviado.
- **Número de acuse de recibo:** Es el número de secuencia del siguiente byte de datos que espera recibir el *host* de destino.
- **HLEN:** Determina la longitud de la cabecera en unidades de 32 bits, ya que la cabecera no es de tamaño fijo debido a la existencia del campo de opciones de tamaño variable.
- **Reservado:** Es un campo reservado para uso futuro.
- ***Code Bits*:** Define 5 bits cuyo significado es el siguiente:

- **URG:** Un valor URG =1 indica que el segmento lleva información urgente en el campo de datos.
- **SYN:** Es utilizado para establecer conexiones. En una petición de establecimiento de conexión el primer segmento enviado por el origen contiene los bits: SYN = 1 y ACK = 0. Un acuse de recibo con: SYN = 1 y ACK = 1 indica una confirmación de establecimiento de conexión.
- **FIN:** Es utilizado para terminar conexiones. Un segmento con los bits: FIN = 1 y ACK = 0 indica el deseo de liberar una conexión. Un segmento con los bits: FIN =

1 y ACK = 1 en un acuse de recibo indica una confirmación de liberación de conexión al solicitante.

- **ACK:** Un valor ACK = 1 indica que el campo de acuse de recibo es válido. Es usado con SYN y FIN, para controlar el establecimiento y liberación de la conexión.
- **RST:** Un valor RST = 1 termina la comunicación en forma abrupta y unilateral de una conexión
- **PSH:** Solicita la entrega inmediata de la información (PUSH), sin que sea almacenada hasta tener un tamaño adecuado de datos para enviar.

- **Ventana:** Este campo indica el número de bytes que en ese momento el destino puede almacenar en sus *buffers* internos. Sólo se aceptarán octetos hasta llenar este *buffer*, no más.
- **Suma de verificacion: (*checksum*):** Permite determinar si han ocurrido errores en el segmento y reconocer si el segmento llegó a su destino y puerto correcto, ya que en el cómputo del *checksum* a más de los campos propios del segmento, TCP utiliza un pseudoencabezado, idéntico al mostrado en la figura 1.22, formado por: las direcciones IP de origen y destino del datagrama IP, tipo de protocolo y longitud del segmento TCP.
- **Puntero de Urgencia:** Es un puntero que permite indicar la posición del último byte de datos urgentes enviados en un segmento junto con otros datos que se encontraban previamente almacenados. El proceso de aplicación deberá determinar donde inicia el mensaje de datos urgentes dentro del campo de datos del segmento TCP.
- **Opciones:** Es utilizado para proporcionar una serie de opciones adicionales. Por ejemplo, permite a los extremos acordar el tamaño máximo de los segmentos que se transferirán. Este campo puede ser de longitud variable. El tamaño por defecto de un segmento es de 536 bytes de datos. Este tamaño implica que un datagrama IP por defecto mide 576 bytes, de los cuales 20 bytes corresponden a la cabecera IP y 20 bytes de cabecera TCP, en ambos casos sin opciones adicionales.[11]

Otras opciones permiten: incrementar el tamaño de la ventana de recepción para líneas de alta velocidad, alto retardo o ambas, emplear retransmisión selectiva de segmentos erróneos, etc.

1.5.2.1 Establecimiento y liberación de una conexión TCP

TCP es un protocolo orientado a conexión por lo cual debe establecer la conexión previa a la comunicación de los procesos de capa aplicación y liberarla una vez que se haya finalizado.

Una conexión se establece de la siguiente forma:

- El *host* de destino, por lo general un servidor que implementa un programa de aplicación, envía una petición a su sistema operativo indicándole que está dispuesto a aceptar cualquier petición de conexión entrante.
- El sistema operativo asigna un número de puerto TCP al proceso en el servidor. El proceso pasa a esperar indefinidamente peticiones de conexión en el puerto asignado. La mayoría de aplicaciones estandarizadas en Internet poseen asignados números de puertos bien conocidos.
- El *host* de origen, denominado cliente, que desea establecer una comunicación con el proceso en el *host* de destino (servidor), solicita a su sistema operativo la asignación de un puerto arbitrario no utilizado y no reservado para su comunicación. El número de puerto en el cliente generalmente no utiliza un número de puerto bien conocido. Sólo los servidores, utilizan números de puerto bien conocidos.
- Una vez asignado un número de puerto libre al *host* de origen, TCP envía un segmento conteniendo una petición de conexión al destino. TCP emplea los bits: SYN=1 y ACK=0 en la cabecera para indicar que el segmento contiene una petición de conexión. El segmento contiene además un número de secuencia aleatorio enviado por el origen y los números de puerto de origen y destino. En la mayoría de aplicaciones el origen conoce el puerto de destino ya que la aplicación es estandarizada. Existen aplicaciones que se usan muy poco, las cuales no siempre se requieren que estén activas monitoreando el puerto asignado, esperando por una petición de conexión, ya que se desperdiciaría recursos. En este caso cuando el origen envía una petición a un proceso en un destino, la petición llega a un servidor de procesos, encargado de monitorear un conjunto de puertos simultáneamente, esperando por una petición de conexión. El servidor de procesos al analizar la petición, inicia el servidor de aplicación al cual estuvo dirigida la petición del

usuario, para que éste continúe con la comunicación con el respectivo cliente.

- El *host* de destino si desea establecer la conexión, envía al *host* de origen un segmento con los bits SYN=1 y ACK=1. Este segmento además contiene en el campo de acuse de recibo el número de secuencia que envió el origen incrementado en una unidad. El destino envía un número de secuencia aleatorio en el campo respectivo.
- El *host* de origen finalmente envía un acuse de recibo del segmento enviado por el destino. Es decir, el segmento contiene en el campo de acuse de recibo, el número de secuencia enviado por el destino e incrementado en una unidad. De esta forma durante el establecimiento de la conexión, los extremos han establecido los números de secuencia iniciales que utilizarán a lo largo de la comunicación.
- A continuación puede empezar la transferencia bidireccional de información.

Debido a que TCP establece una conexión bidireccional, es necesario finalizar independientemente cada conexión unidireccional. Una conexión se la finaliza de la siguiente forma:

- Cualquier extremo envía un segmento TCP con los bits: FIN=1 y ACK=0 indicando que ya no tiene nada más por transmitir y espera recibir un acuse de recibo del otro extremo para poder cerrar la conexión en ese extremo.
- La comunicación de datos en un sentido se ha cerrado, pero la comunicación en el otro sentido aún está abierta, pudiéndose enviar segmentos de datos para cada uno de los cuales el extremo que cerró la conexión debe continuar enviando acuses de recibo, pero ya no datos.
- El extremo opuesto después de transmitir todos los datos que le faltaban por enviar y esperar los respectivos acuses, envía un segmento con los bits: FIN=1 y ACK=1 y espera por un acuse del otro extremo para finalizar la conexión.
- Para evitar que el acuse de recibo de un segmento con el bit FIN activado se pierda, el extremo inicia un contador. Si pasado cierto tiempo no recibe el acuse de fin de conexión, dicho extremo da por cerrada la conexión.
- Finalizada ambas conexiones, TCP elimina todos los registros asociados a esa conexión (números de puerto, números de secuencia, temporizadores, etc.).

1.6 PROTOCOLOS DE CAPA APLICACION

En esta capa se encuentran los protocolos de alto nivel, que permiten al usuario ejecutar aplicaciones reales. Una aplicación puede ser un programa de aplicación escrito por un usuario o una aplicación estandarizada en el conjunto de protocolos TCP/IP. Algunas de las aplicaciones más conocidas son: TELNET, FTP, TFTP, DNS, SMTP, NNTP, HTTP, IRC, etc.

La aplicación debe interactuar con los protocolos de la capa transporte a fin de que esta capa se encargue de enviar y recibir datos en forma de mensajes o flujos continuos de octetos requeridos según la aplicación. Dependiendo de la aplicación se seleccionará un tipo de transporte. UDP es un protocolo no confiable, no orientado a conexión, que no provee control de errores ni control de flujo, por lo cual la aplicación es la encargada de implementar los mecanismos de control de errores y control de flujo. Las aplicaciones que utilizan UDP, son aquellas que logran mayor eficiencia de transmisión al utilizar un menor tamaño de cabecera y donde una entrega rápida es más importante que una entrega confiable.

La mayoría de aplicaciones prefieren utilizar TCP como protocolo de transporte, por ser un protocolo confiable y orientado a conexión, lo cual facilita la elaboración de las aplicaciones sin preocuparse del manejo de su información en la red.

1.6.1 Modelo Cliente - servidor

La mayoría de aplicaciones utiliza el modelo de interacción cliente - servidor para sus comunicaciones.

Un servidor es un programa de aplicación que ofrece un servicio a los usuarios de una red. Un cliente es quien hace un pedido de un servicio. El servidor recibe un pedido, ejecuta el servicio solicitado y envía de regreso los resultados como respuesta al cliente. Un servidor puede usualmente tratar con múltiples solicitudes (múltiples clientes) al mismo tiempo, creando procesos esclavos para cada conexión.

Una aplicación está formada por dos partes: la parte del cliente y la parte del servidor. La parte del cliente de la aplicación es la que un usuario utiliza para enviar los pedidos para un servicio particular a la parte del servidor de la aplicación, la cual se encarga de procesarlos enviando las respuestas necesarias.

La mayoría de servidores de aplicaciones estandarizadas esperan por los pedidos de los clientes en puertos con números de puerto bien conocidos. Esto permite que los clientes sepan exactamente a cual *socket* IP ellos deben enviar sus solicitudes. El cliente utiliza un número de puerto arbitrario, no utilizado y no reservado para su comunicación.

Existen aplicaciones que no utilizan números de puerto bien conocidos. En estos casos el cliente se comunica inicialmente con un servidor de procesos que posee un número de puerto bien conocido, encargado de monitorear los pedidos enviados por varios clientes y asignar un puerto libre al servidor de la aplicación que el cliente desea ejecutar. El servidor de procesos informa al cliente el puerto asignado al servidor y al cual deberá enviar sus posteriores pedidos.

A continuación se detallan algunas de las aplicaciones más utilizadas en el Internet.

1.6.2 DNS

Cada *host* conectado a Internet tiene asignado una dirección IP numérica única. Sin embargo, un usuario a nivel de capa aplicación prefiere utilizar nombres literales fáciles de recordar en vez de direcciones numéricas, para referirse a una máquina en Internet.

El sistema de nombres de dominio (DNS - *Domain Name System*) es una base de datos distribuida de forma jerárquica por toda la red. DNS se encarga de traducir el nombre literal, empleado por las aplicaciones de alto nivel, en la dirección IP utilizada por los protocolos de red antes de proceder con la comunicación con dicha máquina.

Para asignar e identificar en forma única a un *host* en la red mediante un nombre literal, DNS utiliza una estructura jerárquica basada en un árbol invertido, denominado espacio de nombres de dominio mostrado en la figura 1.24.

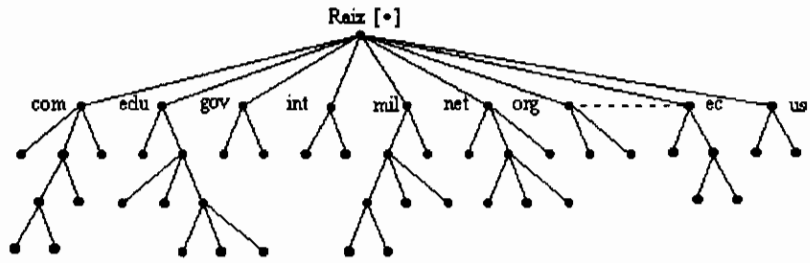


Fig. 1.24. Espacio de Nombres de Dominio DNS [1]

El espacio de nombres de dominio se ramifica a partir de la raíz, creando primeramente nodos de alto nivel. Cada uno de estos nodos, a su vez, puede subdividirse en cualquier forma creando otros nodos a partir de él y así sucesivamente. Cada nodo del árbol tiene asignado un nombre relativo denominado etiqueta, que puede ser de hasta 63 caracteres de largo.[12]

DNS define el concepto de dominio, el cual es un subconjunto del espacio de nombres de dominio, formado por uno o varios nodos consecutivos a lo largo de una ramificación originada en la raíz. Cada dominio tiene asignado un nombre que lo identifica en forma única de los demás, denominado nombre de dominio. El nombre de dominio está formado por el conjunto de etiquetas, separadas entre si por un punto, de todos los nodos del dominio recorridos en forma ascendente a partir del nodo más extremo del dominio hasta llegar a la raíz del árbol.

Cada nodo del espacio de nombres de dominio, es el extremo de algún dominio. Consecuentemente, un nodo tiene asignado, a más de su nombre relativo dado por la etiqueta, un nombre absoluto que lo identifica en forma única de cualquier otro nodo en el espacio de nombres de dominio. El nombre absoluto de un nodo es el nombre del dominio del cual es su nodo extremo. Debido a que cada nodo tiene un nombre absoluto, se puede asignar nombres de dominio, en forma única, a todas las máquinas dentro de su dominio.

DNS es un sistema descentralizado, en el que cada nodo en el espacio de nombres de dominio es administrado localmente por una entidad (organización, empresa o localidad) encargada de asignar nombres a las máquinas bajo su responsabilidad,

garantizando que sean únicos y manteniendo actualizada dicha información. Además la entidad que administra un nodo puede crear otros nodos debajo de él para facilitar la administración. Todos los nodos creados a partir de un nodo deberán tener nombres relativos (etiquetas) distintos entre sí. Se podrán utilizar nombres relativos de otros nodos del árbol sin que se provoque confusión, ya que el nombre absoluto los diferenciará.

Cuando un nodo se divide en otros nodos, se están creando subdominios a partir de un dominio.

El DNS no impone reglas para la forma de escritura y significado de las etiquetas de los nombres de dominio en un nivel particular. Sin embargo, por tradición, el espacio de nombres de dominio utilizado en Internet agrupa las etiquetas de los nodos de alto nivel, que forman los dominios de alto nivel, en dos categorías:

- Etiquetas asociadas a los distintos países, como por ejemplo: "ec" para Ecuador, "us" Estados Unidos, etc.
- Etiquetas especiales que representan a distintos tipos de organizaciones conectadas a Internet, entre las cuales están: [1]

com	Organizaciones comerciales
edu	Instituciones educativas
gov	Instituciones gubernamentales
int	Instituciones internacionales
mil	Instituciones militares
net	Proveedores de servicios de red
org	Organizaciones sin fines de lucro

Tabla. 1.8. Nombres de etiquetas especiales de dominios de alto nivel

Las anteriores son las etiquetas originales, pero actualmente con el crecimiento de Internet se han definido nuevas etiquetas especiales para cubrir en forma más detallada nuevas organizaciones y facilitar la administración del espacio de nombres de dominio. Entre éstas se encuentran: [13]

firm	Negocios o compañías
store	Negocios que ofrecen bienes para comprar
web	Entidades involucradas con el WWW
arts	Entidades relacionadas con asuntos culturales y arte
rec	Entidades relacionadas con recreación y entretenimiento
info	Entidades que proveen servicios de información
nom	Para nomenclatura individual o personal

Tabla. 1.9. Nuevos nombres de etiquetas especiales de dominios de alto nivel

El esquema de nombres permite por ejemplo crear dominios correspondientes a distintas organizaciones dentro de cada país; cada organización a su vez puede crear nuevos dominios bajo su control, asignando un nombre de dominio a cada uno de sus departamentos, etc. La creación de un nuevo dominio, requerirá el permiso del dominio en el cual desea estar incluido.

Ejemplo 1.3.

Las computadoras de la EPN conectadas a Internet, la cual es una institución educativa ubicada en Ecuador, están agrupadas bajo el nombre de dominio epn.edu.ec. En este caso el nombre de dominio está formado por las etiquetas de tres nodos: epn, edu, ec. El dominio epn.edu.ec es un subdominio del dominio edu.ec, el cual representa las instituciones educativas de Ecuador. Este dominio a su vez es un subdominio del dominio de alto nivel ec, el cual representa a todas las instituciones, organizaciones, etc. ubicadas en Ecuador.

Una máquina bajo el dominio epn.edu.ec puede ser un servidor de WWW, llamado por ejemplo: www.epn.edu.ec. DNS se encargaría de traducir este nombre literal en su dirección IP (Ejm. 199.188.57.38) cuando un *host* desee establecer una comunicación con este destino.

Para incluir el dominio fie199.epn.edu.ec, el cual podría administrar los nombres de las máquinas pertenecientes a la Facultad de Ingeniería Eléctrica de la EPN, se debe pedir permiso al dominio superior en el cual va a incluirse, en este caso epn.edu.ec y solamente a éste. No hay necesidad de pedir permiso a dominios superiores como

edu.ec ó ec para la creación del nuevo subdominio. Una vez creado el dominio fie199.epn.edu.ec, éste a su vez podría dividirse en otros subdominios para los departamentos de: Telecomunicaciones, Control, Potencia, sin necesidad de pedir permiso a dominios de niveles superiores. La figura 1.25 aclara la creación del dominio epn.edu.ec dentro del dominio edu.ec.■

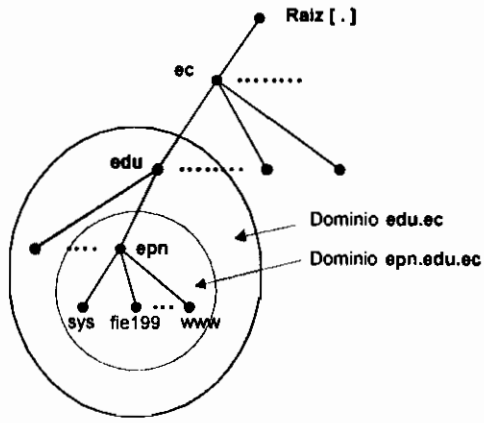


Fig. 1.25. Representación de dominios epn.edu.ec y edu.ec

Las máquinas agrupadas bajo un nombre de dominio no necesariamente están agrupadas en forma física, por ejemplo formando parte de una misma red. Pueden estar separadas geográficamente, sin una interconexión física común. La organización en dominios es una agrupación conceptual. Por ejemplo, bajo el dominio *com* pueden estar ubicadas organizaciones comerciales ubicadas en distintos países del mundo.

Para poder traducir nombres literales de *hosts* o de destinos de correo en una dirección numérica, DNS utiliza una base de datos físicamente distribuida en toda la red en sistemas denominados servidores de nombres. Un servidor de nombres es un programa servidor encargado de asociar los nombres de dominio con direcciones IP como respuesta a un pedido de un cliente.

En realidad la respuesta a un pedido de DNS, no es únicamente la dirección IP, sino un registro de recursos asociado a ese nombre de dominio. Un registro de recursos proporciona información adicional sobre un nombre de dominio como se verá posteriormente.

Una entidad que administra un dominio, cuando crea subdominios delega a otra entidad para que se haga responsable de mantener y actualizar la información de su subdominio. Esta nueva entidad podrá libremente dividir su subdominio en más subdominios y delegarlos a otras entidades, sin permiso de dominios de nivel superior. Un dominio puede tener algunos subdominios y también contener *hosts* que no pertenezcan a ningún subdominio.

Cada dominio posee una base de datos local que asocia nombres de dominio de los *hosts* y subdominios administrados por él, con direcciones IP (registros de recursos) y punteros hacia servidores de nombres, que tengan información de cada subdominio, respectivamente. Un mismo servidor físico puede contener información de varios subdominios e incluso de varios dominios. Esto implica que no necesariamente se requiere un servidor de nombres físico por cada subdominio. Cada dominio (subdominio) es responsable de mantener actualizada su información en el servidor físico.

Para facilitar la resolución de un nombre de dominio en su respectivo registro de recurso, el espacio de nombres de dominios se divide en unidades administrativas denominadas zonas. Una zona es un subárbol del árbol del espacio de nombres de dominio, el cual es administrado por una entidad separada. Una zona está formada por un único dominio o por un dominio con sus subdominios tal como se muestra en la figura 1.26. Ninguna zona debe sobrelaparse con otra.

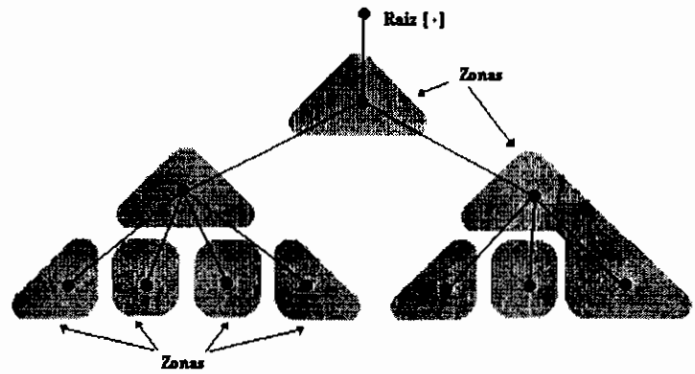


Fig. 1.26. División del espacio de nombres de dominios en zonas

Cada zona contiene servidores de nombres que tienen total información sobre una zona. Son estos servidores los que almacenan la información de varios subdominios o dominios, evitando que se tenga un servidor de nombres por cada subdominio. Se dice que estos servidores tienen autoridad sobre una zona. Un mismo servidor de nombres puede tener autoridad sobre varias zonas. Existen dos tipos de servidores de nombres dentro de una zona: servidores primarios y servidores secundarios.

Un servidor primario es aquel que obtiene la información de la zona sobre la que tiene autoridad a partir de un archivo en el mismo servidor. Un servidor secundario también tiene autoridad sobre la misma zona del primario, pero obtiene su información a partir de un servidor primario, mediante un proceso de transferencia de archivos realizado en forma periódica. El uso de servidores de nombres secundarios permite: brindar redundancia, reducir la carga en el servidor primario y asegurar que todos los *hosts* en la zona tengan un servidor de nombres cercano.

1.6.2.1 Funcionamiento de DNS

Cuando un programa de aplicación (HTTP, FTP, TELNET, correo electrónico, etc.) requiere información de una máquina remota, de la cual sólo conoce su nombre de dominio, emplea una subrutina denominada resolvedor, la cual utiliza el nombre de dominio de la máquina remota como parámetro. El resolvedor (cliente) primeramente verifica su memoria local, a fin de buscar si existe una dirección IP asociada a ese destino. Si encuentra, envía el paquete a esa dirección IP de destino. Caso contrario, envía un paquete UDP o TCP, conteniendo un mensaje de pedido DNS al servidor de nombres más cercano, para que se encargue de buscar el nombre y entregar la dirección IP correspondiente como respuesta al cliente.

Un servidor DNS utiliza generalmente el número de puerto bien conocido UDP 53 para atender pedidos de sus clientes, pero un servidor DNS también puede aceptar conexiones TCP en el mismo puerto 53. La ventaja de UDP es la mayor velocidad respecto a TCP.

Cuando un servidor de nombres recibe una petición de un cliente, determina si tiene autoridad sobre el nombre de dominio pedido. Si es así, envía al cliente una respuesta conteniendo el registro de recursos asociado a dicho nombre de dominio. Cuando el servidor no tiene autoridad sobre el nombre de dominio pedido, puede realizar la búsqueda en dos formas: recursiva o iterativa.

En una búsqueda recursiva, el servidor que no posee autoridad, se encarga por si mismo de contactar otros servidores hasta encontrar la respuesta y entregarla al cliente. Un servidor contiene las direcciones de todos los servidores de nombres que contienen nombres de dominios superiores ubicados en otras zonas hasta llegar a la raíz, y punteros hacia servidores de nombres en subdominios. Los servidores de la raíz tienen direcciones de los servidores de dominios de alto nivel. De esta forma un servidor siempre puede alcanzar otro servidor de nombres en el árbol.

En una búsqueda iterativa, en cambio, el servidor devuelve al cliente la información que el tiene disponible y una lista de otros servidores que el cliente podría, por su cuenta, contactar a fin de continuar su búsqueda volviendo a realizar otra petición.

El tipo de búsqueda que deba ser realizada por un servidor que no tenga autoridad sobre un nombre de dominio se especifica en el paquete de petición hecha por el cliente.

A fin de mejorar la eficiencia de la búsqueda, todo servidor almacena las respuestas de las búsquedas sobre las cuales no tiene autoridad en un *cache*. De esta forma cuando un cliente realiza un pedido para el cual el servidor no tiene autoridad, primeramente busca en el *cache* antes de proceder con los métodos anteriores. Si la respuesta está en el *cache*, es enviada inmediatamente al cliente. Esta respuesta corre el riesgo de no estar actualizada. Para evitar este problema, se limita la duración de la permanencia de un dato en el *cache* a un tiempo especificado por el servidor que proporcionó la respuesta, basándose en la estabilidad o no de cambio de ese nombre de dominio.

La respuesta entregada por un servidor de nombres a un cliente puede ser de dos tipos: autorizada o no autorizada, según el servidor que devuelve la respuesta tenga o no autoridad sobre la zona en la que se encuentra el nombre de dominio pedido. Una

respuesta leída desde un *cache* es no autorizada. Una respuesta no autorizada no implica que no sea correcta, sino que corre el riesgo de no estar actualizada. El cliente almacena la respuesta enviada por un servidor en un *cache* para uso futuro; al igual que con los servidores sin autoridad, solo tendrá permanecia temporal.

DNS también puede ser utilizado con menor frecuencia para realizar asociaciones inversas. Una asociación inversa permite que un cliente envíe una dirección IP de destino como parámetro y que el servidor le envíe el nombre dominio correspondiente a esa dirección.

1.6.2.2 Registros de recursos

Se mencionó que la respuesta entregada por un servidor a un cliente no es únicamente la dirección IP asociada a un nombre de dominio, sino un registro de recursos.

DNS define los siguientes tipos de recursos en una base de datos: [1,2]

Tipo	Significado	Contenido
A	Dirección de red de un *host*	Contiene la dirección IP de un *host*
MX	Intercambio de correo	Contiene el nombre del *host* encargado de recibir correo electrónico en un dominio especifico
NS	Nombre de servidor	Contiene el nombre de un servidor de nombres autorizado para ese dominio. Además contiene nombres de servidores de nombres en dominios superiores.
PTR	Puntero	Contiene el nombre de dominio correspondiente a una dirección IP de un *host*. Usado para búsquedas inversas.
CNAME	Nombre canónico	Permite asociar sobrenombres (alias) al nombre de dominio de una máquina.
HINFO	Información de *host*	Permite especificar algunas características respecto al tipo de *hardware* y sistema operativo que utiliza un *host*.
SOA	Inicio de autoridad	Contiene el nombre de dominio del servidor de nombres primario de una zona, la dirección de correo del administrador de esa zona, números seriales e intervalos de tiempo de referencia.

Tabla. 1.10. Tipos de registros de recursos

Una base de datos en un servidor de nombres es un conjunto de registros de recursos que permite administrar información acerca de: direcciones IP de *hosts* de una zona, direcciones de servidores de correo, punteros hacia otros servidores de nombres en dominios superiores y en subdominios, información del *hardware*, sistema operativo

que utilizan esas máquinas, etc.

Un registro de recursos contiene a más de la información asociada con el nombre de dominio, información que permite a un servidor que recibe el registro de recurso determinar el tiempo de vida que lo puede almacenar en su *cache*, tipo de registro de recurso, etc.

1.6.2.3 Formato de un mensaje DNS

El formato de un mensaje DNS se presenta en la figura 1.27.

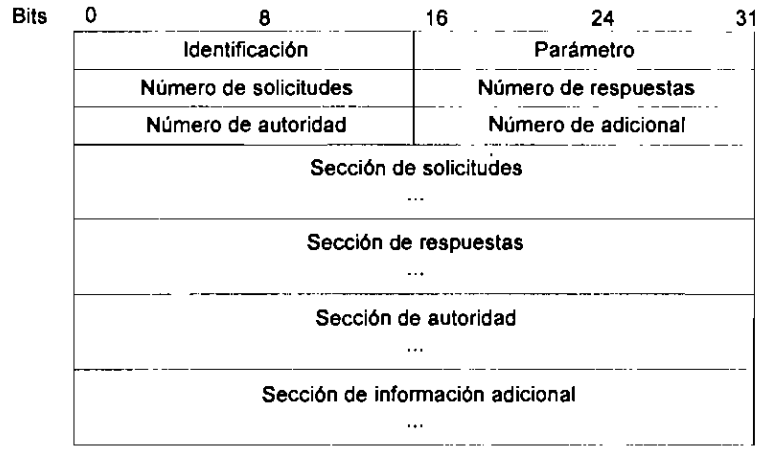


Fig. 1.27. Formato de un mensaje DNS [14]

- **Identificación:** Cuando un cliente envía un pedido, coloca en el campo Identificación un número aleatorio. La respuesta desde algún servidor contendrá el mismo número de identificación. Esto permite reconocer cuál es la respuesta a un pedido.
- **Parámetro:** Especifica: el tipo de operación solicitada (pedido o respuesta), el tipo de solicitud, el tipo de búsqueda si es un servidor no autorizado (recursiva o no recursiva), y el código de respuesta.

A continuación se especifica el significado de los bits del campo Parámetro. [14]

Bit	Campo	Valor	Significado
0	Operación	0	Solicitud
		1	Respuesta
1-4	Tipo de solicitud	0	Estándar
		1	Inversa
		2	Pedido de estado del servidor
		3-15	Reservado
5	Indicadores AA		Activado si tiene respuesta autorizada
6	TC		Activado si el mensaje fue truncado debido a que su tamaño era mayor al permitido por el canal de transmisión.
7	RD		Activado si se desea recursión
8	RA		Activado si la recursión está disponible
9-11	Reservado		Reservado
12-15	Tipo de respuesta	0	Sin error
		1	Error de formato en la solicitud
		2	Falla en el servidor
		3	El nombre no existe
		4	El servidor no puede hacer este tipo de búsqueda

Tabla. 1.11. Significado de los bits del campo Parámetro del mensaje DNS.

- **Número de Solicitudes:** Un mensaje DNS puede contener varios pedidos simultáneamente. Una respuesta de un servidor puede contener todas o algunas de las respuestas a un mensaje de pedido DNS. El campo Número de Solicitudes contiene el número de pedidos que contiene un mensaje de pedido DNS.
- **Número de Respuestas:** Contiene el número de registros de recursos (respuestas) que contiene la Sección de Respuestas de un mensaje de respuesta DNS.
- **Número de Autoridad:** Especifica el número de registros de recursos de servidores de nombres, que tienen autoridad sobre un nombre de dominio, contenidos en el campo Sección de Autoridad. Estos registros de recursos son parte de la información que un servidor no autorizado devuelve como respuesta a un cliente que solicitó una búsqueda iterativa. El cliente podrá contactarse con cualquiera de estos otros servidores de nombres volviendo a realizar otra petición.
- **Número de Adicional:** Contiene el número de registros de recursos contenidos en la Sección de Información Adicional. La Sección de Información Adicional puede contener registros de recursos asociados a registros de recursos en la Sección de Respuestas y que podrían ser de utilidad para el cliente.
- **Sección de Solicitudes:** Contiene uno o más pedidos de resolución de nombres de dominio. Un pedido, contenido en este campo, tiene el siguiente formato:

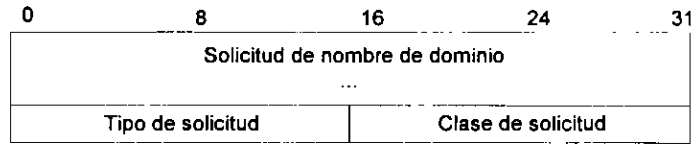


Fig. 1.28. Formato de un pedido DNS [14]

- **Solicitud de Nombre de Dominio:** Contiene las etiquetas que forman el nombre de dominio. Cada etiqueta está precedida por un octeto que indica el tamaño de la etiqueta.
- **Tipo de Solicitud:** Especifica si el nombre de dominio, es un nombre de un máquina, una dirección de correo, etc. DNS puede ser utilizado sobre algunas otras familias de protocolos que no sean los de Internet.
- **Clase de Solicitud:** Identifica la familia de protocolos sobre la cual se ejecuta DNS. En el caso de Internet, la clase se la denomina IN.

- **Sección de Respuestas, Sección de Autoridad y Sección de Información Adicional:** contienen una o más respuestas a los pedidos de resolución de direcciones en un mensaje de pedido DNS. La respuesta a un pedido como se mencionó es un registro de recurso. Una respuesta en estos campos tiene el siguiente formato:

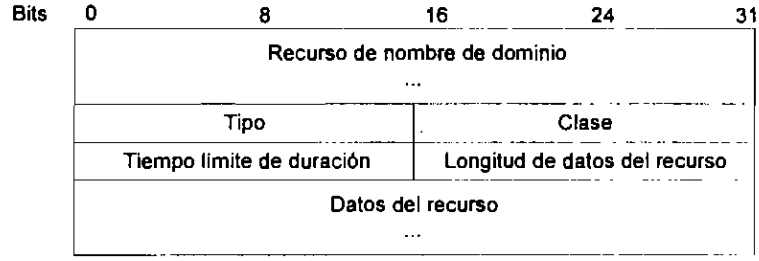


Fig. 1.29. Formato de una respuesta DNS [14]

- **Recurso de Nombre de Dominio:** Contiene el nombre de dominio asociado a ese registro de recurso. El nombre está formado por etiquetas precedidas por un octeto que determina la longitud de cada etiqueta.
- **Tipo:** Especifica el tipo de registro de recurso que contiene la respuesta.
- **Clase:** Identifica la familia de protocolos sobre las que se ejecuta DNS.
- **Tiempo Limite de Duración**: Contiene el tiempo máximo que el recurso contenido en la respuesta puede mantenerse en memoria *cache* en un servidor o un *host*.

- **Longitud de datos de recurso:** Especifica el tamaño en octetos de los datos del recurso contenidos en el campo de Datos de Recurso.

1.6.3 TELNET

El Protocolo de Terminal de Red (TELNET) permite a un usuario, el cliente TELNET, acceder remotamente a recursos en cualquier otra máquina en la red, conocida como servidor TELNET, convirtiendo el terminal del cliente en un terminal local conectado a ese servidor, es decir como si el usuario estuviese directamente operando en la máquina remota. Con TELNET un usuario remotamente puede acceder al disco duro, manejar archivos, ejecutar aplicaciones, acceder a datos, imprimir un documento en el destino remoto, modificar configuraciones, etc.

TELNET utiliza TCP como protocolo de transporte. Un servidor TELNET utiliza el puerto TCP número 23 para atender peticiones de sus clientes. [15]

El usuario inicia una sesión remota especificando la dirección IP o el nombre de dominio de la computadora a la cual se desea conectar. Una vez establecida la conexión, el servidor TELNET solicita el nombre de usuario y el *password* para determinar si el cliente puede tener acceso o no a la información en la máquina remota. TELNET a continuación transfiere cada caracter tecleado por el cliente directamente al otro sistema y transporta la salida de la máquina remota a la pantalla del usuario.

El protocolo TELNET se encarga de:

- Definir el concepto de Terminal Virtual de Red (NVT - *Network Virtual Terminal*) que es un dispositivo imaginario el cual provee una interfaz estándar para todos los sistemas remotos, permitiendo la interoperabilidad entre computadores reales basados en distintos sistemas operativos, cada uno de los cuales maneja un formato y significado distinto de los caracteres, especialmente los de control. El NVT oculta a un cliente estos detalles en los posibles sistemas remotos al definir una interfaz estándar. Los caracteres enviados por el cliente son traducidos al formato del NVT y enviados al servidor, el cual traduce los datos del NVT al formato que dicho

servidor utilice. De forma similar sucede en la comunicación del servidor al cliente.

- Negociar un conjunto de opciones de terminal entre el cliente y el servidor a fin de proveer servicios adicionales a más de los disponibles en el NVT.
- Permitir que un cliente pueda ser un usuario con un teclado o un programa que envía los caracteres al servidor.

1.6.4 FTP

El Protocolo de Transferencia de Archivos (FTP - *File Transfer Protocol*), permite a un usuario en cualquier computadora (cliente) traer archivos desde cualquier otra computadora (servidor), o enviar archivos a cualquier otra computadora en una red TCP/IP como Internet. Además también permite a un usuario borrar, renombrar, mover y copiar archivos en un servidor. FTP requiere que un usuario se identifique mediante un nombre de usuario y un *password*, previo a la realización de una transferencia de archivos, a fin de brindar seguridad de la información almacenada restringida. Generalmente un usuario común puede utilizar: "*Anonymous*" como nombre de usuario y tendrá acceso a un conjunto de archivos de uso público en un servidor FTP.

FTP utiliza TCP como protocolo de transporte para brindar una conexión extremo a extremo confiable. FTP establece dos tipos de conexiones TCP: una conexión de control y una conexión de transferencia de datos. La conexión de control utiliza el protocolo TELNET, el cual utiliza el concepto de terminal virtual de red (NTV) para poder enviar datos de control entre los terminales remotos sin preocuparse por el formato de los mismos en cada sistema. La conexión de control permite intercambiar comandos de control sobre la transferencia de archivos como por ejemplo: el número de puertos y procesos que serán usados por la conexión de transferencia de datos, el nombre del archivo a transferir, etc. La conexión de control permanece activa durante toda la sesión FTP. La conexión de transferencia de archivos es creada dinámicamente cuando se lo requiere y es utilizada para transferir un archivo entre un extremo y otro.

La conexión de control utiliza en el servidor el puerto TCP número 21, en el cliente un puerto libre. Para la conexión de transferencia de datos el servidor utiliza el puerto TCP número 20; el número de puerto en el cliente se lo asigna dinámicamente mediante un

acuerdo entre los extremos, utilizando la conexión de control para asegurar la transferencia correcta con el cliente. [16]

FTP dispone de una interfaz de usuario, que permite emplear un conjunto de comandos para: iniciar la sesión FTP con una máquina remota utilizando su nombre de dominio o su dirección IP, seleccionar directorios y archivos a transferirse, especificar el formato de representación (tipo de caracter, convención de fin de línea, etc.) en que un archivo debe ser transferido, llevar a cabo la transferencia de un archivo y finalizar la sesión.

1.6.5 TFTP

El Protocolo de Transferencia de Archivos Trivial (TFTP - *Trivial File Transfer Protocol*) es un protocolo muy simple que permite transferir archivos, utilizando el protocolo de transporte no confiable UDP. A diferencia de FTP, únicamente permite enviar archivos hacia un servidor o copiarlos desde él, sin proporcionar ningún mecanismo de autenticación en base a nombres de usuario o *passwords*.

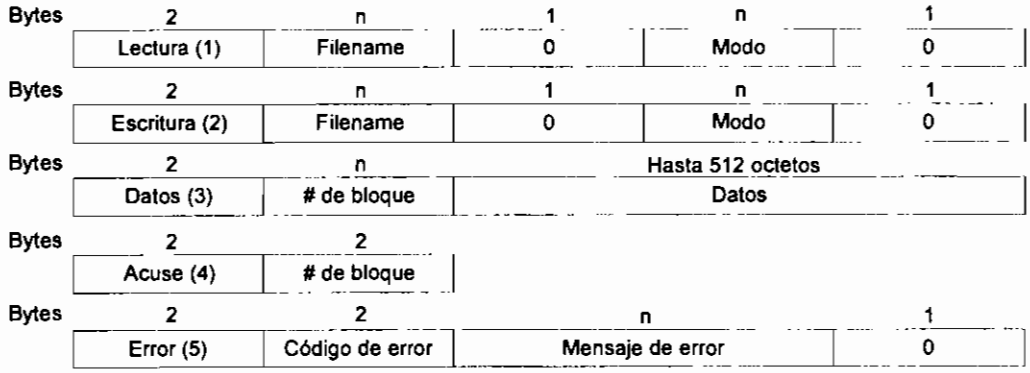


Fig. 1.30. Tipos de paquetes TFTP [17]

TFTP define cinco tipos de paquetes mostrados en la figura 1.30. En todos ellos el primer campo especifica el tipo de paquete: un pedido de lectura (1), un pedido de escritura (2), un paquete de datos (3), un acuse de recibo (4) o un paquete de error (5).

El cliente para iniciar la transferencia con el servidor envía un paquete de pedido de lectura o escritura al puerto UDP número 69, asignado al servidor de TFTP. El cliente

utiliza un puerto libre. El paquete de petición indica si la transferencia será una lectura o una escritura, además contiene el nombre del archivo y el modo (formato) de los datos en el archivo que el protocolo transferirá (caracteres ASCII estándar, datos binarios en forma de octetos de ocho bits, etc.).

El servidor al aceptar el pedido abre la conexión y continua con la transferencia de los paquetes de datos. TFTP divide el archivo que se transferirá en bloques de tamaño fijo de 512 bytes cada uno, asignando a cada uno de ellos un número de bloque. Cada paquete de datos contiene el número de bloque y los 512 bytes de datos del respectivo bloque. [17]

Puesto que TFTP utiliza los protocolos no confiables: UDP en capa transporte e IP en capa internet, el *host* de destino debe enviar un paquete de acuse de recibo por cada paquete recibido antes de que el *host* de origen pueda volver a enviar un nuevo paquete, para asegurar que fue recibido correctamente. El acuse de recibo enviado por el destino indica el número del último bloque que fue recibido correctamente.

En caso de no recibir un acuse de recibo después de un cierto tiempo de haber enviado un paquete, el origen vuelve a retransmitirlo, pues asume que el paquete de datos se perdió o el acuse se perdió.

La transferencia se termina cuando se transfiere un paquete de datos que contiene en el campo de datos menos de 512 bytes. La transferencia también puede terminar abruptamente al recibirse un paquete de error. Un error puede darse, por ejemplo: debido a archivos no encontrados, falta de espacio en el disco de almacenamiento, nombre de archivo ya existente, etc. El paquete de error TFTP, contiene el código del error que originó el envio del paquete, utilizado por el *software* TFTP y un mensaje de error mostrado al usuario.

La interfaz de usuario proporciona un conjunto de comandos que permiten: conectarse con el *host* de destino utilizando su dirección IP, especificar el formato de los datos, realizar la transferencia de lectura o escritura y terminarla.

1.6.6 Correo electrónico

El correo electrónico o *e-mail* es una de las aplicaciones mas populares en el Internet, la cual permite a un usuario enviar mensajes a otros usuarios en otras computadoras en cualquier lugar del mundo en forma casi inmediata. El éxito del correo electrónico, a diferencia de otros medios como el teléfono o el fax, radica en que permite la transferencia de mensajes sin necesidad de que el usuario origen deba interrelacionar simultáneamente con el usuario destino a fin de transferir el mensaje. Además el correo electrónico permite enviar un mismo mensaje simultáneamente a varios usuarios. Aunque no es el medio adecuado, el correo electrónico, se ha convertido en la aplicación más utilizada para intercambiar, a más de los mensajes de texto: documentos con formato, archivos de programas, imágenes, audio, etc., llegando a ser incluso más popular que el mismo FTP, el cual poco a poco está quedando en desuso.

El correo electrónico ha facilitado la interrelación de personas a nivel mundial a través de la red, por las ventajas de rapidez y bajo costo que ofrece. Por otra parte, el correo electrónico ha sido una de las piezas claves para el crecimiento de muchas empresas, las cuales han visto en este medio de comunicación, el sistema óptimo para el intercambio de su información a nivel interno y externo.

El funcionamiento del correo electrónico es muy similar al funcionamiento del correo postal normal, por lo que es importante asociarlos. En el correo postal normal, un usuario después de escribir su carta, especifica claramente la dirección de correo postal del destinatario en el sobre y posteriormente la deposita en su buzón de correo o la entrega a la oficina de correo. La oficina de correo enviará la carta hacia su destino, pudiendo pasar por varias otras oficinas de correo en su camino, si fuese el caso. Si la dirección es correcta, entregará la carta colocándola en el buzón de correo del destinatario. El destinatario podrá acercarse a su buzón, recoger la carta y leerla. Si la dirección fue incorrecta o no pudo enviar la carta, ésta será devuelta al usuario.

Un usuario de correo electrónico para poder utilizar este sistema requiere poseer un buzón de correo en un servidor de correo, generalmente ubicado en un servidor físico propiedad del proveedor de servicios de Internet, al cual el usuario está afiliado ó en un

servidor de correo propio. Un mismo servidor físico puede ser capaz de almacenar varios buzones de correo correspondientes a distintos usuarios en forma similar a los buzones físicos ubicados en una misma oficina de correo postal.

En el correo electrónico, un usuario edita el contenido de su mensaje utilizando un programa de correo comercial. Especifica la dirección de correo del destinatario y procede a enviar el mensaje. El usuario asume que el mensaje ha sido entregado al destino. Sin embargo, el mensaje no viaja directamente al usuario final desde la máquina del usuario origen, sino que inicialmente se almacena una copia del mensaje enviado en el buzón del usuario en el servidor de correo, cuyo nombre de dominio está especificado en la configuración del programa de correo. El servidor de correo posteriormente se encargará de enviar el mensaje al buzón correspondiente en el *host* de destino, especificado en la dirección de correo del destinatario. Para poder enviar el mensaje previamente deberá transformar la dirección de correo del destinatario en la dirección IP de la máquina destino, utilizando el sistema de nombres de dominio DNS. La ventaja de que el servidor almacene una copia del mensaje enviado es la de permitir que el computador del usuario origen no tenga que preocuparse por problemas tales como fallas en la red, ó que el servidor de destino se encuentre apagado u ocupado en ese momento como para poder recibir el mensaje, dejando al computador del usuario libre para poder ejecutar otras aplicaciones o incluso apagarlo. El servidor por si solo se encarga de volver a intentar más tarde hasta poder transmitir el mensaje. Una vez enviado el mensaje, el servidor elimina la copia local. El usuario final podrá acceder a su buzón y recuperar todos los mensajes que le hayan sido enviados.

En caso de que haya pasado un tiempo límite (algunas horas o días) y el servidor no hubiese podido enviar aún el mensaje, enviará un mensaje de error al usuario de origen.

Todo usuario que utiliza correo electrónico requiere de una dirección de correo electrónico. La dirección de correo electrónico es similar a la dirección postal del buzón del correo de un usuario en una oficina de correo. La dirección de correo electrónico permite identificar en forma única la computadora de destino utilizada por un usuario para recibir su correo. La dirección de correo está formada por dos partes: un identificador de usuario, que define un buzón en la máquina de destino (oficina de

correo) y el nombre de dominio de la máquina destino que contiene dicho buzón, separadas entre sí por el signo @. Es decir: identificador de usuario@nombre de dominio.

Por ejemplo: alexba@inter.net es una dirección hipotética de correo electrónico, en la cual alexba es el identificador de usuario o buzón, e inter.net es el nombre de dominio de la máquina en la cual se encuentra el buzón de correo, en el cual el usuario puede enviar y recibir su correo.

Técnicamente hablando, el correo electrónico debe encargarse de los siguientes aspectos:

- El formato de los datos que se van a transmitir.
- La forma como transportar un mensaje desde el *host* de origen hasta el *host* de destino, a través de la red, asegurando que éste llegue completo.
- La forma en que un usuario puede acceder a su buzón en un servidor de correo y recuperar los mensajes a él enviados.
- La forma como rutear los mensajes en base a direcciones de correo utilizando el DNS.

A continuación se detallan cada uno de estos puntos.

1.6.6.1 Formato de un mensaje de correo electrónico

Al igual que una carta normal, un mensaje de correo electrónico está formado por dos partes: un encabezado y un cuerpo separados entre si por una línea en blanco. El encabezado contiene información de control tal como: la dirección del destinatario, la dirección del origen, etc. El cuerpo contiene el contenido del mensaje.

1.6.6.2 RFC 822

El encabezado está formado por un conjunto de campos, denominados cabeceras, cada uno de ellos en una línea separada, los cuales poseen un nombre de campo de la forma

"Nombre: ", seguidos por un valor. El RFC 822 es el estándar que define el formato e interpretación de cada campo en un encabezado.

Los principales campos RFC 822 presentes en un encabezado junto con el valor que los acompaña se presentan a continuación:

Cabecera	Valor
To:	Dirección(es) de correo de destinatario(s) principal(es)
Cc:	Dirección(es) de correo de destinatario(s) secundario(s)
Bcc:	Dirección(es) de correo de destinatario(s) oculto(s) a los de To: y Cc:
From:	Dirección de correo del usuario que creó el mensaje
Sender:	Dirección del usuario que envío el mensaje
Message-Id:	Contiene un número que identifica un mensaje
Reply to:	Dirección de correo a la cual se deben enviar las respuestas
Subject:	Breve descripción acerca de lo que trata el mensaje

Tabla. 1.12. Principales campos de cabecera RFC 822 [1]

Adicionalmente existen otros campos, pero que no son muy utilizados en programas de correo comerciales.

A continuación del encabezado se encuentra el cuerpo del mensaje. En el cual un usuario puede poner lo que desee.

1.6.6.3 *Simple Mail Transfer Protocol* (SMTP)

El Protocolo de Transferencia de Correo Simple (SMTP - *Simple Mail Transfer Protocol*) es el protocolo encargado de transferir el mensaje en forma confiable entre servidores de correo.

SMTP inicialmente transfiere el mensaje desde la computadora del usuario al buzón de correo del usuario en el servidor de correo. Posteriormente, el servidor de origen utiliza SMTP para transferir el mensaje desde el buzón de origen hasta el buzón de destino en el servidor de correo final

El correo electrónico se basa en el modelo cliente-servidor. El cliente es el *host* quien envía el mensaje y el servidor es quien recibe el mensaje. Cuando el usuario conectado

a un servidor de correo desea enviar un mensaje a un destino, el programa de correo establece una conexión TCP con el puerto TCP 25 en el servidor de correo local a fin de enviar su mensaje. [3] El servidor almacena una copia temporal del mensaje enviado por el usuario y termina la conexión con él. A continuación el servidor utiliza el sistema de nombres de dominio a fin de transformar la parte del nombre de dominio de la dirección de correo del servidor que contiene el buzón de destino, en la dirección IP de dicha máquina. La respuesta del DNS al pedido del servidor de correo son registros de recursos, del tipo: MX y A. El tipo MX contiene el nombre del *host* encargado de recibir el correo electrónico y el recurso tipo A contiene la dirección IP de dicho *host*. Una vez hecho esto, el servidor de origen se convierte en cliente. Establece una conexión TCP con el puerto 25 en el servidor de destino y transfiere el mensaje en forma confiable, utilizando un conjunto de primitivas especificadas en el protocolo SMTP. Finaliza la conexión una vez que el mensaje se ha transferido completamente.

SMTP utiliza un conjunto de comandos para el diálogo entre cliente-servidor. El cliente envía comandos de cuatro letras, que tienen un significado especial para el servidor. El servidor responde con mensajes formados por dos partes: un código numérico, utilizado por el proceso del cliente y una parte literal para informar al usuario, la cual no es procesada. En SMTP todos los comandos, respuestas y datos intercambiados son líneas de texto ASCII, terminadas en el caracter CR-LF (*Carriage Return - Line Feed*). [3]
SMTP utiliza la siguiente secuencia de pasos a fin de transferir un mensaje entre un cliente y un servidor: [3]

1. El cliente establece la conexión TCP con el servidor y espera a que el servidor le envíe el mensaje 220 *READY FOR MAIL*, indicando que está listo para iniciar la transferencia de correo, ó el mensaje 421 *Service not available* cuando el servidor por alguna razón no puede continuar con la transferencia.
2. Recibido el mensaje 220, el cliente envía el comando de saludo HELO, con el cual el cliente se identifica ante el servidor con su nombre de dominio.
3. El servidor responde identificándose con su nombre de dominio ante al cliente.
4. El cliente envía el comando MAIL el cual contiene el campo de la cabecera FROM:, el cual indica la dirección de correo de quien creó el mensaje que se va a enviar.
5. El servidor indica que está listo para recibir, enviando el mensaje 250 OK.

6. El cliente envía el comando RCPT seguido por el campo To: de la cabecera, el cual especifica la dirección de correo del destinatario. El servidor, si contiene el buzón especificado en la dirección, envía el mensaje 250 OK. Si el buzón no está presente envía el mensaje de error 550 *No such user here.*
7. El cliente debe enviar un mensaje RCPT por cada destinatario que supuestamente tenga un buzón en el servidor con el cual se estableció la conexión. El servidor debe responder individualmente a cada comando RCPT. Un mensaje de usuario no encontrado no detiene la conexión, únicamente no envía a ese usuario el mensaje. Al resto si.
8. Una vez que el servidor ha respondido a todos los mensajes RCPT, el cliente envía el comando DATA para indicar que está listo para enviar el mensaje.
9. El servidor responde enviando el mensaje 354 *Start mail input*, seguidos por una secuencia de caracteres que el cliente deberá utilizar para indicar el fin del mensaje. Generalmente se usa la secuencia: <CR-LF> . <CR-LF>, es decir una línea al final del mensaje que únicamente contenga un punto.
10. El cliente envía el mensaje de correo (encabezado y cuerpo) línea por línea
11. El servidor detecta la secuencia de fin de mensaje y envía el mensaje 250 OK.
12. El cliente cierra su conexión con el comando QUIT
13. El servidor cierra su conexión con el mensaje 221.

Los anteriores son los comandos más utilizados. Sin embargo existen muchos más que permiten realizar funciones más complejas implementadas en SMTP.

Ejemplo 1.4.

A continuación se muestra un ejemplo hipotético que aclara como se transmite un mensaje utilizando el protocolo SMTP. Supongamos que el usuario alexba@inter.net envía un mensaje a su amigo en andres@web.net. La secuencia mostrada son los comandos y respuestas intercambiadas entre los servidores de origen y destino, después de que el mensaje fue inicialmente almacenado en el buzón del usuario de origen en el respectivo servidor para su transmisión. Los pasos para este proceso inicial son similares a los del ejemplo, únicamente cambia el nombre del servidor de destino.

S: 220 web.net SMTP service ready
C: HELO inter.net
S: 250 web.net
C: MAIL FROM:<alexba@inter.net>
S: 250 OK
C: RCPT TO:<andres@web.net>
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CR-LF>.<CR-LF>
C: ....Envía el cuerpo del mensaje
C: ....Envía el cuerpo del mensaje
C: <CR-LF>.<CR-LF>
S: 250 OK
C: QUIT
S: 221 web.net Service closing transmission channel.

1.6.6.4 *Mutipurpose Internet Mail Extensions* (MIME)

En sus inicios, el estándar del SMTP únicamente permitía el envío de mensajes escritos en alfabeto Inglés utilizando el código ASCII (*American Standard Code for Information Interchange*) no extendido de 7 bits. Además SMTP imponía otras restricciones tales como: [1,3]

- El mensaje solo podía contener caracteres de texto en código ASCII.
- El mensaje no podía exceder una determinada longitud (64 KB).
- La longitud máxima de una línea era de 1.000 caracteres.
- El mensaje no podía contener partes con tipos distintos

Sin embargo, el crecimiento del correo electrónico y el desarrollo de muchas aplicaciones especialmente de multimedia obligaron a la creación de un nuevo estándar que permitiese al usuario enviar mensajes formados por diferentes partes, no necesariamente texto, las cuales pudiesen estar en un formato que no fuese ASCII.

Las Extensiones Multipropósito para Correo en Internet (MIME - *Multipurpose Internet Mail Extensions*) es el protocolo actualmente más utilizado para enviar mensajes con formato no ASCII utilizando el correo electrónico. MIME permite a un usuario por ejemplo: enviar mensajes de longitud ilimitada que contengan documentos con formato (Ejm. con tildes, caracteres especiales, negritas, cursivas, otros idiomas), archivos de programas ejecutables, imágenes, audio, etc.

MIME se desarrolló para ser totalmente compatible con el formato de la cabecera definido por el RFC 822 y con el protocolo de transporte SMTP sin modificarlos en absoluto. MIME únicamente se encarga de añadir una estructura al cuerpo del mensaje y definir reglas para codificar en ASCII mensajes no ASCII, para luego enviarlos utilizando los mismos protocolos empleados por el correo estándar.

MIME define cinco tipos de campos los cuales los añade al encabezado utilizado por RFC 822 antes del inicio del cuerpo del mensaje y también algunos dentro de él. MIME utiliza estos campos para especificar el tipo de datos que contiene el cuerpo del mensaje y la técnica de codificación que ha sido utilizada para representar esos datos en formato ASCII. Los cinco tipos de campos definidos por MIME son: [1]

- ***MIME-Version***: Especifica la versión del protocolo MIME utilizado. Actualmente la versión es 1.0. Cuando este campo no está presente en el encabezado, se asume que el cuerpo del mensaje es únicamente texto en formato ASCII.

- ***Content-Type***: Especifica el tipo y subtipo de los datos contenidos en el cuerpo del mensaje. MIME define los siguientes tipos de datos: [2]

    - *Text:* Para representar información tipo texto.
    - *Image:* Para transmitir imágenes fijas, fotos.
    - *Audio:* Para transmitir audio o voz.
    - *Video:* Para transmitir vídeo o imágenes en movimiento.
    - *Aplication:* Para transmitir aplicaciones ejecutables o datos binarios.
    - *Message:* Para encapsular un mensaje de correo.
    - *Multipart:* Para combinar varios tipos en el cuerpo del mensaje.

  Cada tipo de dato puede tener subtipos, los cuales especifican el formato del archivo de datos o una característica especial de los datos. El subtipo se indica a continuación del tipo, separados entre si por el signo " / ". En la tabla 1.13 se presentan los principales tipos y subtipos definidos en MIME: [1,3]

Tipo	Subtipo	Significado
Text	*Plain*	Contiene texto sin formato en ASCII
	Richtext	Contiene texto con formato
Image	*Gif*	Contiene una imagen con formato GIF(*Graphics Interchange Format*)
	Jpeg	Contiene una imagen con formato JPEG (*Joint Photographic Experts Group*)
Audio	*Basic*	Contiene un archivo de sonido codificado con 8 bits y muestreado a 8 kHz
Video	*Mpeg*	Contiene un archivo de vídeo con formato MPEG (*Motion Picture Experts Group*)
Application	*Octet Stream*	Indica que los datos del mensaje son una secuencia sin interpretación de bytes (8 bits), tal como un programa ejecutable. Este es el subtipo utilizado para datos que no se encuentren en ningún otro subtipo.
	PostScript	Indica que contiene datos en formato Post Script, utilizado para describir páginas impresas. PostScript es un lenguaje de programación.
Message	*RFC 822*	Contiene un mensaje RFC 822 contenido completamente dentro del mensaje
	Partial	Indica que un mensaje ha sido dividido en partes y cada parte es enviada separadamente. En el destino se debe reensamblar el mensaje.
	External Body	Indica una dirección (ftp, tftp, de correo) en la red donde un usuario destino puede descargar un mensaje muy largo si está interesado en él. Esto evita cargar la red con mensajes muy grandes y que pueden no interesar a un destino.
Multipart	*Mixed*	Indica que las diferentes partes contenidas en un mensaje son independientes pero que deben ser transmitidas simultáneamente y ser mostradas al usuario final en el orden en el que aparecen en el mensaje.
	Alternative	Indica que todas las partes de un mensaje son las mismas, pero que cada una está representada o codificada en forma distinta. La aplicación de correo del destinatario escogerá la mejor forma para mostrar la información, dependiendo del formato que pueda soportar. En el mensaje las partes se ordenan según la mejor opción que el origen considera para mostrar el mensaje.
	Parallel	Indica que las distintas partes de un mensaje deben ser presentadas al destinatario en forma simultánea. Por ejemplo un mensaje con audio y vídeo, ambas partes se deben ejecutar al mismo tiempo.
	Digest	Indica que múltiples mensajes están siendo enviados en un único mensaje.

Tabla. I.13. Principales tipos y subtipos definidos en MIME

- ***Content-Transfer-Encoding*:** Especifica la técnica que se usó para codificar en ASCII los datos enviados, para poderlos transmitir utilizando el sistema de correo. La misma técnica deberá ser utilizada en recepción en forma inversa para decodificar los datos a su formato original. Algunos de los tipos de codificación más usados son: *base 64, quoted-printable*, etc.

Un mensaje multiparte utiliza una secuencia de caracteres ASCII como frontera o separador de partes. Cada parte contiene los campos *Content-Type:* y *Content-*

Transfer-Encoding: para indicar el tipo de datos y la forma en que está codificada cada parte del mensaje.

- ***Content-Description*:** Contiene una breve descripción del contenido de un mensaje. Permite al destinatario saber si el contenido le es o no de interés y proceder a abrirlo o no. Es útil por ejemplo para describir en pocas palabras el contenido de un archivo de audio o de vídeo.
- ***Content-ID*:** Especifica un identificador único para el mensaje. Es un campo opcional.

Ejemplo 1.5.

A continuación se presenta un ejemplo en el cual se ve claramente como el protocolo MIME permite enviar, en un mismo mensaje, varios tipos de contenido cuyo formato no necesariamente es ASCII. MIME no modifica los encabezados definidos en RFC 822 sino que añade los suyos propios.

From: alexba@inter.net
To: andres@web.net
Reply to: fer@meca.com
MIME-Versión: 1.0
Content-Type: Multipart/ Mixed; Boundary= xxxxxxxx
Subject: Por favor revisa este documento

--xxxxxxxx
Content-Type: text/richtext

Hola Andrés. Te envío el documento del que te comenté necesito le des una revisada y el programa que me pediste. Cuando hayas leído y modificado el documento se lo envías a mi secretaria en la dirección adjunta en *Reply-To:* Además te envío la foto del paseo.
Gracias. Nos vemos.

.... datos del documento con formato

--xxxxxxxx
Content-Type: image/jpeg
Content-Transfer-Encoding: base 64
....datos de la imagen codificada en ASCII usando la técnica base 64

--xxxxxxxx
Content-Type: application/octet stream
Content-Transfer-Encoding: base 64
.... datos del archivo ejecutable

Se puede ver que el mensaje está formado por el encabezado, una línea en blanco y el cuerpo del mensaje. En el encabezado están presentes los campos definidos por RFC 822 y los campos añadidos por MIME. No necesariamente deben estar presentes todos los campos revisados. La cabecera indica que es un mensaje multiparte y establece una cadena de caracteres para separar las distintas partes. Cada parte, a excepción de aquellas que empleen únicamente caracteres ASCII (texto en Inglés sin formato), especificará el tipo de contenido y técnica de codificación utilizada. Este mensaje completo es transmitido utilizando un protocolo de transporte de correo como SMTP o POP3.

1.6.6.5 *Post Office Protocol* (POP3)

Generalmente un usuario utiliza el correo electrónico desde un PC o desde una estación de trabajo en una red local. Sin embargo estas máquinas por falta de recursos no pueden mantener un programa servidor de SMTP residente, continuamente ejecutándose, en espera de recibir correo, ya que generalmente deben ejecutar otras tareas del usuario ó están apagadas. Además mantener una conexión permanente a Internet en espera de correo resultaría cara o imposible

Para resolver este problema, el sistema de correo electrónico utiliza servidores físicos conectados permanentemente a Internet y especialmente dedicados para ejecutar continuamente el servidor de correo, los cuales reciben el correo de distintos usuarios afiliados a él, almacenándolos en su respectivos buzones.

El Protocolo de Oficina Postal versión 3 (POP3 - *Post Office Protocol*) es el protocolo encargado de permitir a un usuario acceder desde un PC o una estación de trabajo, a su buzón en un servidor de correo y recuperar los mensajes a él enviados.

Es necesario diferenciar entre SMTP y POP3 ya que ambos son protocolos para transferir correo de un punto a otro. POP3 provee los mecanismos para que el programa de correo del usuario final interactúe con su buzón de correo virtual en un servidor, en el cual sus mensajes esperan ser enviados o retirados, mientras que SMTP es utilizado principalmente para transferir mensajes entre servidores de correo definiendo el formato

y significado de los comandos utilizados en el diálogo entre servidores.

Cuando un usuario desea retirar mensajes de su buzón, el programa de correo establece una conexión TCP con el puerto TCP 110 utilizado por el servidor de POP3. Cuando la conexión es establecida, el servidor de POP3 envía un saludo. [18] El cliente y el servidor POP3 a continuación intercambian un conjunto de comandos y respuestas respectivamente hasta que la conexión se cierra o es abortada.

Los comandos en POP3 consisten de una palabra clave posiblemente seguida por un argumento. Todos los comandos terminan con los caracteres CR-LF. Las respuestas en POP3 consisten de un indicador de suceso y una palabra clave posiblemente seguida de información adicional. Todas las respuestas terminan en un par CR-LF. Hay dos tipos de respuestas o indicadores de suceso: positivo "+OK" y negativo "-ERR". Para cada comando enviado por el cliente existe una respuesta del servidor.

Una sesión POP3 tiene tres estados. Una vez que se establece la conexión TCP y el servidor envía el saludo, la sesión entra al estado de Autorización, en el cual el cliente debe identificarse a si mismo ante el servidor POP3 con un nombre de usuario y una clave. Si el nombre y clave son correctos, el servidor adquiere los recursos asociados con el buzón del cliente y se pasa al estado de Transacción. Los recursos son los mensajes que están presentes en el buzón. El servidor envía una respuesta positiva en la cual indica el número y tamaño total de los mensajes contenidos en el buzón. En este estado el cliente solicita acciones por parte del servidor POP3, tales como: traer el mensaje o eliminarlo del buzón. Cuando el cliente termina sus transacciones, la sesión pasa al estado de Actualización, en la cual, el servidor libera cualquier recurso adquirido durante el estado de transacción, elimina definitivamente los mensajes borrados durante las sesión, se despide y termina la conexión TCP.

Ejemplo.1.6.

A continuación se presenta una secuencia típica de comandos y respuestas intercambiadas entre un cliente y un servidor POP3.

S: +OK mail1 POP3 server ready (Comments to: mail1@inter.net)
C: USER alexba
S: +OK alexba is a real user
C: PASS clave secreta
S: +OK alexba's maildrop has 2 messages (280 octets)
C: STAT
S: +OK 2 280
C: LIST
S: +OK 2 messages (280 octets)
S: 1 100
S: 2 180
C: RETR 1
S: +OK 100 octets
S: < El servidor POP3 envía el mensaje completo>
S: .
C: DELE 2
S: +OK message 2 deleted
C: QUIT
S: +OK mail1 POP3 server signing off

La transferencia inicia con un saludo enviado por el servidor POP3. El cliente se identifica utilizando el comando USER para indicar el nombre de usuario (buzón) y el comando PASS para indicar su clave secreta. El servidor responde enviado el número de mensajes y el tamaño medido en octetos que el buzón tiene en ese instante. A continuación, el usuario puede ejecutar el comando STAT para saber cuantos mensajes tiene y el tamaño total de su buzón o puede usar el comando LIST para saber el tamaño individual de cada uno de los mensajes en el buzón. Cada mensaje tiene asignado un número que lo identifica. En base a este número el usuario podrá traerlo hasta su máquina con el comando RETR o eliminarlo del buzón con el comando DELE. Cuando se trae un mensaje, el fin del mensaje se indica con la secuencia <CR-LF>.<CR-LF>, es decir con una línea enviada por el servidor que únicamente contenga un punto. Finalmente el usuario termina la sesión con el comando QUIT, ante el cual el servidor se despide y termina la conexión.

Si surge algún problema, por ejemplo: nombre de usuario inexistente, clave incorrecta, número de mensaje no existente, el servidor envía el adecuado mensaje de error. Existen otros comandos a más de los indicados que realizan funciones más especificas, pero los anteriores son los más utilizados.

1.6.7 *Usenet News*

Usenet News es una aplicación que permite a un usuario intercambiar artículos relacionados con un determinado tema, en el cual tenga interés (ciencia, política, deportes, social, etc.), con otras personas a nivel mundial que comparten sus mismos intereses. Todas las personas interesadas en el tema se suscriben a lo que se denomina un grupo de noticias o foro de discusión.

Un grupo de noticias puede ser implementado en un servidor físico de una compañía, de un proveedor de servicios, etc., encargado de almacenar temporalmente todos los artículos relacionados al tema, enviados por varios usuarios, para que sean puestos a discusión de otros usuarios del grupo de noticias. Un mismo servidor puede contener varios grupos de noticias. Cada servidor de noticias está conectado directamente con uno u otros servidores de noticias, denominados *newsfeed(s)*, con los cuales se comunica periódicamente para intercambiar aquellos nuevos artículos que recientemente hayan sido recibidos (publicados) en cada grupo que estos servidores administren. De esta forma se difunde un nuevo artículo a través de todos los servidores en la red.

Cada grupo de noticias posee un nombre que lo distingue de los demás. Para facilitar la asignación de nombres de los grupos de discusión, se los ha dividido en jerarquías de alto nivel las cuales abarcan en forma general todos los campos en que un usuario pueda estar interesado. Los nombres de las jerarquías de alto nivel se presentan a continuación: [1]

Nombre	**Tópico del Tema**
Comp	Computadores, ciencia e industria de la computación
Sci	Ciencias físicas en ingeniería
Humanities	Literatura y humanidades
News	Discusión sobre el mismo grupo de noticias
Rec	Actividades recreacionales, deporte, música
Misc	Misceláneos
Soc	Aspectos sociales
Talk	Debates, polémicas, discusiones
Alt	Cualquier otra actividad alternativa

Tabla. 1.14. Principales jerarquías de USENET

Cada una de estas jerarquías posee subjerarquías a fin de especificar más detalladamente acerca de qué trata un determinado grupo de noticias. El administrador del servidor físico que implementa uno o varios grupos de noticias es el encargado de administrar la creación de nuevas subjerarquías según pedidos de los usuarios.

El nombre del grupo de noticias está formado por los nombres de las jerarquías y subjeraquías separadas entre sí por un punto. Así por ejemplo un grupo de noticias dedicado a música pop puede estar bajo el nombre Rec.music.pop, donde Rec es el nombre de la jerarquía principal, *music* y *pop* son nombres de subjerarquías.

Un usuario utiliza un programa especial denominado un lector de noticias (*news reader*) para poder leer los artículos relacionados al tema publicados en el grupo de noticias respectivo. Un *news reader* también permite al usuario: enviar artículos, suscribirse, anular la suscripción a un grupo de noticias, etc. Esta aplicación está muy relacionada con el correo electrónico. Inicialmente se implementaba un grupo de noticias en base a una lista de correo. Cuando un usuario deseaba publicar un artículo, utilizaba una lista de correo conteniendo las direcciones de todos los destinatarios a quienes debía ser entregado el artículo.

Debido al crecimiento de Internet, ya no es posible utilizar listas de correo para publicar un artículo. Ahora, un usuario que desea publicar un artículo, utiliza el programa lector de noticias (*news reader*) para enviar el artículo a un servidor de correo de una compañía, de un proveedor de servicios, etc., en forma similar a como se envía correo electrónico. El servidor recibe los mensajes de correo entrantes direccionados a grupos de noticias y los almacena generalmente en un directorio denominado *news*, el cual posee subdirectorios tales como *comp, sci, rec*, etc. Dependiendo a qué grupo de noticias esté direccionado el artículo enviado es almacenado en el respectivo subdirectorio.

Un artículo enviado a un grupo de noticias utiliza el mismo formato utilizado por los mensajes de correo descritos en el RFC 822, junto con algunas cabeceras adicionales para transferir su artículo. Las cabeceras adicionales utilizadas para enviar un artículo a un grupo de noticias usando correo electrónico son: [1]

Cabecera	Valor
Newsgroups:	Especifica el grupo de noticias, o los grupos de noticias, a los cuales va dirigido el artículo enviado.
Followup-to:	Especifica el grupo de noticias al cual se debe enviar los comentarios y reacciones de un determinado artículo que ha sido publicado.
Distribution:	Especifica el alcance geográfico que debe tener la publicación del artículo.
Nntp-Posting-Host:	Especifica el nombre de dominio de la máquina que públicó el artículo.
References:	Especifica si un artículo es respuesta a otro artículo. Toda respuesta contiene la identificación del artículo al cual hace referencia. En un envío de un nuevo artículo no se utiliza este campo.
Organization:	Permite especificar la organización con la cual el usuario que envía un artículo está ligado.
Lines:	Especifica la longitud del cuerpo del artículo.
Summary:	Contiene una pequeña descripción de lo que trata un artículo.

Tabla. 1.15. Cabeceras utilizadas en artículos de USENET

Además de estos campos, la cabecera del artículo contiene algunos de los campos definidos en RFC 822 tales como: *From:*, *Subject:* y *Message-Id:* los cuales identifican: la dirección de correo de quien envía el artículo, una breve descripción de lo que trata el cuerpo del mensaje y una identificación del mensaje respectivamente. *Message-Id:*, es utilizado como el nombre del archivo. Tal como en el correo electrónico, la cabecera del artículo está separada del cuerpo del artículo mediante una línea en blanco.

1.6.7.1 *Network News Transfer Protocol* (NNTP)

El Protocolo de Transferencia de Noticias de Red (NNTP - *Network News Transfer Protocol*) es el encargado de transferir los artículos desde un servidor de noticias a otro y también de permitir a un *host* que no puede recibir noticias, leerlas remotamente.

NNTP utiliza TCP como protocolo de transporte. El cliente, un servidor de noticias, puede establecer la conexión con su servidor *newsfeed*, o el servidor *newsfeeed* puede solicitar al servidor de noticias que acepte una conexión TCP con él. En ambos casos el servidor *newsfeed* emplea el puerto TCP 119 para ejecutar el programa servidor NNTP.[19]

NNTP es equivalente a SMTP. SMTP permite transferir archivos entre servidores de correo. NNTP permite transferir artículos de noticias entre servidores de noticias,

permitiendo que un artículo publicado en él se difunda en forma rápida a todos los usuarios suscritos a ese grupo en cualquier lugar del mundo donde se encuentren. NNTP define las primitivas necesarias para que un servidor de noticias pueda pedir a su servidor *newsfeed* una lista de nuevos grupos y artículos que este último posea, una lista de los artículos en cada grupo, una lista de nuevos artículos en cada grupo, solicitar la transferencia de un articulo determinado, permitir la transferencia de artículos desde un servidor *newsfeed* de un servidor hacia otro servidor *newsfeed*, y para terminar la sesión TCP.

1.6.8 *Gopher*

Gopher es una aplicación antigua que permite a un usuario buscar y recuperar documentos que residen en muchos servidores distribuidos en el Internet. Ya que los documentos en Internet están distribuidos en varias máquinas resultaría muy difícil para un usuario saber en donde se encuentra un documento que le interese. *Gopher* se encarga de facilitar la tarea de búsqueda al usuario, presentando un menú con un listado de documentos a los cuales el usuario puede tener acceso tan solo seleccionando el nombre del documento que desea revisar. Además *Gopher* permite realizar búsquedas de documentos relacionados con un tema especifico.

Ante el usuario, *Gopher* se asemeja a un sistema de archivos que agrupa varios archivos bajo directorios. En el caso de *Gopher*, los directorios agrupan documentos que pueden estar distribuidos en cualquier lugar de la red y no necesariamente en una misma máquina.

El protocolo y el *software Gopher* se basan en el modelo cliente-servidor. *Gopher* utiliza el protocolo de transporte TCP, pues requiere de un flujo de datos confiable. Un cliente que desee utilizar *Gopher*, utiliza un *software* de cliente, el cual establece una conexión TCP con el puerto TCP 70 en un servidor remoto, [20] en el cual se ejecuta el servidor *Gopher*. Una vez establecida la conexión, el servidor *Gopher*, se encarga de enviar un menú conteniendo una lista de directorios o una lista de documentos que tiene acceso. Toda lista termina en una línea que contiene únicamente un punto. Una vez transmitido el menú desde el servidor al cliente, se finaliza la conexión TCP.

El cliente al recibir el menú conteniendo la lista de directorios puede seleccionar uno de ellos e inmediatamente se desplegarán los subdirectorios o documentos contenidos en ese directorio. Al seleccionar el nombre de un documento, el documento será automáticamente transferido desde cualquier punto donde se encuentre almacenado y presentado al cliente. Para cada directorio o documento que el cliente ingresa, *Gopher* debe establecer una nueva conexión TCP con el destino que contiene el ítem seleccionado, lo transmite al cliente y termina la conexión.

Un listado presentado en un menú *Gopher* puede hacer referencia a distintos objetos ubicados en distintos servidores en la red. Cada ítem en el listado del menú es identificado por un tipo, que especifica la clase de objeto que es el ítem (Ejm. directorio, archivo de texto, archivo binario, ítem de búsqueda, etc.). Ante el usuario únicamente se presenta un nombre literal al que hace referencia el titulo de un documento o un directorio. Sin embargo, *Gopher* envía para cada ítem la siguiente información, que será utilizada por el *software* del cliente para localizar un documento especifico en un servidor remoto: [20]

- Tipo de ítem, que identifica la clase de objeto que el ítem es.
- Nombre literal visible para uso del usuario, le permite leer y seleccionar un ítem.
- Una cadena de caracteres que contiene la ruta o *pathname* utilizada por el cliente para localizar el objeto deseado en el servidor remoto.
- Un nombre de dominio del *host* donde se encuentra el ítem
- Un número de puerto en el cual el servidor acepta conexiones

El *software* de cliente utilizará el *pathname*, el nombre de *host* y el número de puerto para establecer la conexión TCP con el servidor que contiene el ítem que le interese y traerlo hasta su máquina.

El administrador de cada servidor será encargado de crear y mantener las páginas que contienen las listas de directorios, documentos y enlaces a otros servidores *Gopher*.

Una desventaja de *Gopher* es que únicamente soporta texto y no imágenes. *Gopher* en la actualidad está quedando en desuso, ante el aparecimiento del WWW el cual presenta

una interfaz de usuario mucho mas amigable combinando éste y muchos otros servicios en uno solo.

1.6.9 *Veronica, Archie* y *Wais*

Son tres aplicaciones que permiten realizar búsquedas en la red.

- **Veronica** (*Very Easy Rodent-Oriented Net-wide Index to Computarized Archives*). Veronica es un sistema que permite tener acceso a recursos de información almacenados en la mayoría de servidores *Gopher* en cualquier lugar del mundo de forma rápida.

  Veronica se encarga de visitar los sitios *Gopher*, leer los directorios y nombres de archivos contenidos en los menús de estos servidores *Gopher*, y luego crear un gran índice conteniendo todos ellos. Cuando un usuario desea realizar una búsqueda, a través de Veronica, escribirá algunas palabras claves que hacen referencia a un tema de interés y Veronica a través del mismo cliente *Gopher* se encargará de buscar. El resultado de la búsqueda de Veronica será un menú *Gopher* conteniendo un listado con ítems en los cuales las palabras claves introducidas por el usuario estén presentes. El usuario podrá seleccionar cualquiera de los ítems del menú *Gopher* para revisar el documento o directorio.

  Veronica elimina la necesidad de que el usuario deba estar pasando de servidor en servidor en busca de información hasta llegar al servidor que contiene la información de interés.

- **Archie.** Archie es el equivalente de Veronica para servidores FTP. Es un servicio que permite a un usuario buscar todos los sitios FTP que contienen archivos relacionados a un determinado tópico. Archie utilizará el nombre de un archivo como criterio de búsqueda y revisará las listas de contenidos mantenidas en servidores FTP a fin de encontrarlo y permitir que el usuario pueda realizar la transferencia.

- **WAIS** (*Wide Area Information Servers*). WAIS es un sistema encargado de crear una base de datos de asuntos especializados en múltiples servidores, y permitir a un usuario acceder a esta base de datos para realizar una búsqueda mediante programas clientes WAIS. El usuario ingresa un argumento de búsqueda en la base de datos y WAIS se encarga de entregar como resultado una descripción de cada texto en los que se encuentre el argumento de búsqueda. El usuario puede luego recuperar el texto completo.

En la actualidad tanto Veronica, Archie y WAIS pueden ser ejecutados en forma mucho más fácil a través de un programa navegador de Web. El uso de estos buscadores es apropiado cuando los motores de búsqueda existentes en el Web no proveen la información requerida por un usuario o cuando el usuario desea realizar una búsqueda más especializada que no está disponible en el Web, sino en servidores *Gopher*, FTP o WAIS respectivamente.

1.6.10 *World Wide Web* (WWW)

El WWW (*World Wide Web*) es la aplicación más importante del Internet, la cual permite a un usuario acceder a documentos, conocidos como páginas Web, dispersos en miles de máquinas en todo el Internet. Una página Web es un documento que puede incluir texto, audio, imágenes o vídeo y a su vez enlaces hacia otras páginas en la red.

El WWW se basa en los conceptos de hipertexto e hipermedia. El hipertexto es una cadena de texto resaltada en una página Web, la cual posee un enlace hacia otro documento en la red. Una cadena de hipertexto con enlaces a otras páginas también se conoce como un hipervínculo. La hipermedia extiende el concepto de hipertexto. Un documento de hipermedia combina cadenas de texto resaltado, imágenes, iconos, etc. que pueden poseer enlaces hacia otros documentos en la red. Un usuario puede seleccionar uno de estos elementos generalmente haciendo *click* sobre él, para ir hacia otra página Web que pueda contener mas información sobre algún tema y la cual puede estar ubicada en cualquier lugar del mundo. Las páginas Web están almacenadas en servidores denominados servidores Web o servidores WWW.

Generalmente se utilizan los términos Web e Internet como sinónimos, pero en verdad no son la misma cosa. El Web es un subconjunto de la red Internet, una colección de documentos de hipertexto/hipermedia enlazados entre si. El Internet es la colección de máquinas y enlaces que permiten el almacenamiento y el intercambio de información.

Para usar el World Wide Web, un usuario necesita utilizar un programa conocido como Web *browser* o navegador del Web, que permite a la computadora del usuario comunicarse con otras computadoras en el Internet. Algunos de los programas navegadores (*browsers*) mas conocidos son: *Mosaic*, *Netscape Navigator*, *Internet Explorer*, *Lynx*, etc.

Podría pensarse que *Gopher* y WWW realizan la mismas tareas, pero sin embargo no es así. *Gopher* únicamente permite al usuario tener acceso a documentos de texto disponibles en varios servidores alrededor del mundo. *Gopher* también puede permitir al usuario transferir programas ejecutables, imágenes, sonido y archivos de vídeo desde un servidor hasta su computador, pero no mostrarlos. Una vez que el archivo haya sido transferido, el usuario puede ejecutar el *software* adecuado para ver o utilizar un determinado archivo.

El WWW a diferencia de *Gopher* posee una interfaz gráfica, añade las capacidades de hipertexto e hipermedia, presentación automática de gráficos y más facilidades de uso. Cuando el usuario selecciona una frase de hipertexto o un icono en el documento Web, el *software* de su navegador busca la información en el servidor de Internet que contiene el documento al cual hace referencia el hipervínculo, lo transfiere al usuario desde ese servidor hasta la máquina del cliente y lo presenta al cliente. Si el documento contiene imágenes, el navegador se encarga automáticamente de mostrarlas dentro del cuerpo del documento, sin necesidad de utilizar aplicaciones separadas como en el caso de *Gopher*. Algunos *browsers* requieren ejecutar aplicaciones conocidas como *plug-in(s)*, para poder ejecutar o mostrar algunos de los contenidos de una página Web que el *browser* por si solo no pueda ejecutarlos. Un *browser* puede ejecutar estos programas suplementarios en forma automática, sin intervención del usuario, para ejecutar archivos de audio, vídeo u otras funciones. A estos programas *plug-in* también se les conoce como visores externos o aplicaciones de ayuda. El *browser* se encarga de integrar todas

estas aplicaciones para presentar una página Web en forma automática.

En un documento Web pueden incorporarse gráficos, distintas fuentes de letras, texto con formato, audio, vídeo, etc. Un documento Web dista mucho de un sencillo menú de solo texto entregado por *Gopher*. Un navegador de Web puede conectar servidores *Gopher*, pero los navegadores de *Gopher* no pueden accesar a sitios Web.

Un navegador de Web permite a un usuario utilizar muchos otros protocolos de Internet tales como: FTP, *Gopher*, NNTP, WAIS, Archie, etc. De esta forma un usuario no necesita de un *software* de cliente particular para tener acceso a cada uno de estos otros recursos disponibles en la red, sino que con el mismo navegador con el cual accede a páginas Web en el WWW puede utilizar estos otros servicios, incluso de una manera más fácil que con los mismos programas especialmente dedicados para cada aplicación.

El WWW utiliza tres componentes fundamentales:

- *Universal Resource Locator* (URL): Permiten identificar en forma única un documento en el Web.
- *Hypertext Transfer Protocol* (HTTP): Es el protocolo encargado de la comunicación entre servidores y navegadores de Web.
- *Hypertext Markup Language* (HTML): Define el formato de los documentos Web y la forma en que un documento debe ser presentado ante un usuario.

1.6.10.1 *Universal Resource Locator* (URL)

En el Web, a todo documento, imagen, audio o vídeo se lo denomina un recurso. Para poder tener acceso a estos recursos, el Web utiliza los localizadores de recursos universales (URLs - *Universal Resource Locators*). Un URL permite: identificar un recurso en la red de forma única, saber donde está localizado y cómo se puede acceder a él.

Con un navegador de Web, se puede escribir el URL correspondiente a un recurso y directamente tener acceso a él, ó al hacer *click* sobre un hipertexto o un icono, el

navegador utilizará el URL al cual hace referencia ese enlace para ubicarlo y transferirlo.

En forma general un URL está constituido por las siguientes partes: [21]

<protocolo>:<información especifica del protocolo>

La parte Protocolo se refiere al tipo de protocolo utilizado para acceder a ese recurso. El protocolo utilizado para transferir documentos Web se denomina HTTP. Sin embargo, como se mencionó anteriormente un navegador de Web puede ser utilizado para acceder a varios otros recursos tales como: FTP, *Gopher*, News, Telnet, etc. Cada protocolo tiene asignado un nombre que permite al navegador identificar qué tipo de protocolo deberá usar para transferir un recurso. Los nombres URL más utilizados son:

Nombre	Usado por:
http	HTTP
ftp	FTP
gopher	*Gopher*
mailto	Dirección de correo electrónico
news	Grupo de noticias o artículo de noticias
nntp	Artículo de noticias
telnet	Acceso remoto
wais	Wais
file	Nombre de archivo en un *host* local

Tabla. 1.16. Nombres de URL de protocolos más utilizados [1]

La parte de información específica del protocolo está separada del nombre de protocolo por dos puntos. La sintaxis de esta parte depende del tipo de protocolo utilizado, pero en general posee la estructura mostrada a continuación: [21]

//<user>:<password>@<host>:<port>/<url-path>

Algunas de las partes tales como: "//", "<user>:<password>@", ":<password>", ":<port>", y "/<url-path>" pueden no estar presentes, dependiendo del protocolo. [H]

[H] **Nota:** Los signos < > en realidad no están presentes en el URL, solo se han utilizado para distinguir las distintas partes que forman la parte específica del protocolo.

El signo "//" puede estar o no presente dependiendo del protocolo y actúa como un delimitador. La parte "*user*" especifica un nombre de usuario y "*password*" especifica una clave que el usuario debe utilizar para acceder a un recurso. La parte "*host*" especifica el nombre de dominio del *host* que contiene un recurso. La parte "*port*" es el puerto con el cual el navegador deberá establecer la conexión a fin de transferir un recurso. Si el número de puerto se omite, se utilizará el número de puerto por defecto asignado a ese servidor. El "*url-path*" especifica la ruta o *path* que permite acceder a un determinado recurso en el *host* de destino que lo contiene. Esta parte varía dependiendo del tipo de protocolo.

Uno de los URL más utilizados es el correspondiente al protocolo HTTP. La sintaxis específica de este URL se presenta a continuación: [21]

http://<host>:<port>/<path>?<searchpart>

Al igual que en el formato general, "*host*" es el nombre de dominio de la máquina que contiene un documento Web al cual se desea acceder, "*port*" es el número de puerto con el cual se debe realizar la conexión. En caso de que se omita el valor de un puerto se utilizará el puerto por defecto número 80, que es un puerto TCP bien conocido. Los campos *path* y *searchpart* son opcionales. El "*path*" especifica la ruta o *pathname* en la máquina donde se encuentra ubicado el documento Web (directorio/subdirectorio/ nombre del archivo). El "*searchpart*" especifica una cadena de búsqueda en una base de datos, la cual siempre va precedida por el signo "?". Cuando no hay un "*path*" o un "*searchpart*" el usuario automáticamente accede a una página Web por defecto o de bienvenida en el servidor de destino, la cual puede tener enlaces a otras páginas en ese y otros servidores.

Ejemplo 1.7.

A continuación se presenta unos URL que pueden ser familiares para estudiantes de la EPN.

http://www.epn.edu.ec
http://fie199.epn.edu.ec/
http://fie199.epn.edu.ec/info.htm

El primer URL muestra la página de bienvenida de la E.P.N, el segundo muestra la página de bienvenida de la Facultad de Ingeniería Eléctrica de la EPN. El último accede directamente a un recurso (página Web) denominado info.htm en el servidor fie199.epn.edu.ec, el cual brinda información más detallada sobre las tareas que efectúa la Facultad de Ingeniería Eléctrica de la EPN.■

Cada protocolo posee su sintaxis específica, en la cual se omiten dependiendo del protocolo una u otra de las partes del formato general. Además según el protocolo se detalla la sintaxis de su parte "*url-path*", en el caso de que exista. La información de la sintaxis del URL de cada protocolo se encuentra totalmente detallada en el RFC 1738.

A continuación se muestra la sintaxis general de algunos de los URL usados por distintos protocolos que pueden utilizarse a través de un navegador de Web, sin entrar en detalle del significado de cada parte que lo conforma. Para una análisis completo de cada URL es necesario revisar el RFC 1738 y el RFC 1630.

ftp://<user>:<password>@<host>:<port>/<url-path>
gopher://<host>:<port>/<gopher-path>
mailto:<dirección de correo>
nntp://<host>:<port>/<nombre de grupo de noticias>/<número de artículo>
telnet://<user>:<password>@<host>:<port>/
wais://<host>:<port>/<nombre de base de datos>
file://<host>/<path>

1.6.10.2 URIs (*Uniform Resource Identifiers*): URL, URN, URC

Un URL es parte de un concepto mucho más amplio conocido como URI (*Uniform Resource Identifier*). Un URI en si es un concepto que permite identificar en forma única un recurso en el Internet, sea éste una página de texto, un vídeo, audio, una imagen fija o animada, un programa ejecutable, etc.

Un URI puede ser de varias clases tales como: URLs (*Uniform Resource Locators*), URNs (*Uniform Resource Names*), y URCs (*Uniform Resource Characteristics*). Estos se suponen trabajan juntos en un esquema general para nombrar, describir y recuperar recursos en el Internet.

El URI más común y más utilizado es el URL (*Uniform Resource Locator*), el cual se lo puede imaginar como una dirección de un recurso, tal es así que a los URL más se los conoce como direcciones de páginas Web. Un URL típicamente describe:

- El mecanismo utilizado para accesar al recurso
- La computadora específica donde el recurso está contenido
- El nombre específico del recurso (un nombre de archivo) en una computadora.

La mayoría de usuarios que utilizan el Internet a través de un navegador están familiarizados con los URLs y probablemente conocen sus problemas: enlaces muertos, sitios sobrecargados, bastante tráfico en ciertos enlaces, etc. Todos estos problemas son causados debido a que los URLs confunden el nombre de un recurso con su localización. Es decir, un recurso actualmente está asociado a un localidad (sitio Web) permanente, lo cual ocasiona los problemas mencionados cuando muchos usuarios desean acceder a un mismo recurso. Para evitar eso actualmente se están desarrollando los conceptos de URN y URC.

Un URN es una clase de URI que permite nombrar a un recurso con un nombre persistente e independiente de su localización, es decir sin necesidad de asignarle una dirección. Un URN permite a un usuario tener acceso a un mismo recurso el cual puede estar distribuido en varios puntos en la red, sin necesidad de que el usuario deba conocer exactamente donde el recurso está localizado. La ubicación del recurso puede variar con el tiempo pero el objetivo, aún en desarrollo, es que el navegador del usuario o alguien más deberá encontrarlo usando su URN.

A fin de usar un URN será necesario agrupar los recursos en lo que se denomina nombres de espacios. Un nombre de espacio se asemeja a una categoría. Cada nombre de espacio contendrá recursos posiblemente basados en distintos tipos de protocolos. El

usuario solo deberá conocer el nombre del espacio en el que un recurso se encuentre y el nombre del objeto o recurso dentro de él. El programa se debería encargar de encontrar y recuperar el recurso asociado a ese URN, en forma independiente del formato específico de protocolo que se use para acceder a él.

Un URN está asociado con otro concepto denominado URC (*Uniform Resource Characteristics*), el cual es otra clase de URI que brinda información descriptiva asociada a un URN, tal como: autor, título, asunto, fecha, longitud, y más. Un URC también contendrá un conjunto de URLs de localidades donde se puede encontrar el recurso. El navegador del usuario tomará uno de los URLs y tratará de encontrar y recuperar el archivo. Si no puede, tratará con otro URL, luego otro y otro hasta lograr transferir el recurso o hasta agotar los URLs. [22]

Todos los URNs tienen la siguiente sintaxis: [23]

urn: <NID> : <NSS>

La parte <NID> es el identificador de un nombre de espacio, y NSS es el nombre o identificador de recurso en ese nombre de espacio. La frase "urn:" puede escribirse en minúsculas o mayúsculas. Existen un sinnúmero de reglas y convenciones para las partes <NID> y <NSS> descritas en el RFC 2141.

Ejemplo 1.8.

Por ejemplo un usuario que desee buscar información sobre un libro o una publicación escrita podría utilizar un URN de la forma: [1] urn:ISBN:0-395-36341-1, para acceder a una publicación, libro en línea, o información relacionada a él, que se encuentre bajo el nombre de espacio ISBN, cuyo nombre de recurso sea 0-395-36341-1.

La ISBN (*International Standard Book Number*) es la organización encargada de identificar la edición de un libro. El nombre del recurso, en este caso 0-395-36341-1 es

[1] Ejemplo basado en el RFC 2288

un código de la ISBN que representa la editorial, el título del libro, el autor, etc. de una obra especifica que el usuario desea tener información.

El programa del usuario se encargará de buscar y presentar al usuario información acerca de este recurso desde cualquier lugar donde se encuentre sin necesidad de que el usuario deba contactarse con un lugar específico, por ejemplo, el sitio hipotético: http:\\www.isbn.com\books\art\johnson\395.html.

1.6.10.3 *Hypertext Transfer Protocol* (HTTP)

El WWW se basa en el modelo cliente-servidor. En el WWW, el cliente es el programa navegador de Web (*browser*), y el servidor, denominado servidor Web, es la máquina remota en la cual se encuentra un recurso al cual el cliente desea acceder. Para poder intercambiar información entre un cliente y un servidor, éstos deben comunicarse entre si.

El protocolo estándar de comunicación entre servidores Web y los clientes es el Protocolo de Transferencia de Hipertexto (HTTP - *Hypertext Transfer Protocol*). El HTTP es un protocolo de capa aplicación, orientado a conexión que utiliza TCP como protocolo de transporte. HTTP define un conjunto de reglas para poder intercambiar recursos tales como: archivos de texto, imágenes, sonido, vídeo y otros archivos de multimedia a través del WWW.

Cada vez que un usuario desea acceder a un recurso, ya sea escribiendo su URL o haciendo *click* sobre un enlace de hipertexto o en algún icono, el programa navegador de Web (cliente) debe encargarse de determinar el URL correspondiente al recurso que se desea acceder. Luego el navegador solicita al DNS se encargue de transformar el nombre de dominio, presente en el URL, en la dirección IP de destino. Hecho esto, el navegador procede a establecer una conexión TCP con el puerto bien conocido TCP 80 en el cual se ejecuta el servidor Web, en la máquina remota que contiene el recurso al cual se quiere acceder.

Una vez establecida la conexión TCP, se inicia la interacción entre cliente y servidor. El cliente envía un pedido al servidor solicitando se le transfiera el recurso especificado en la parte "*path*" del URL. La respuesta del servidor transfiere el recurso en cuestión. Una vez transferido el recurso, HTTP finaliza la conexión TCP y el navegador de Web se encarga de presentar el documento Web al usuario. Para cada recurso que se vaya a transferir HTTP debe utilizar una conexión independiente.

A más de pedidos de transferencia de recursos, HTTP también permite enviar pedidos que requieren que el servidor ejecute alguna acción con el recurso especificado en el URL como se verá posteriormente. Para cada pedido HTTP siempre existe una respuesta desde el servidor.

HTTP es un protocolo que permite la transferencia de recursos entre un servidor y un cliente. Estos recursos pueden ser de varios tipos: texto, imágenes, audio, vídeo, etc. HTTP requiere especificar tanto en pedidos como en respuestas el tipo de recurso y algunas características asociadas a él, que el otro extremo necesitará conocer para poder procesar dicho recurso una vez que sea transferido. HTTP hace uso del formato de cabeceras RFC 822, definido para el correo electrónico, y también emplea MIME para añadir estructura a un mensaje a fin de permitir transferir recursos de varios tipos como se verá posteriormente.

a) Formato de un pedido HTTP

Como se ha mencionado, el cliente una vez establecida la conexión TCP, envía un pedido al servidor, el servidor responde a ese pedido y termina la conexión. Tanto un pedido como una respuesta tiene una estructura definida por el protocolo HTTP.

El pedido típico de un cliente tiene la siguiente estructura: [24]

Pedido = <Método>< URL>< Versión del Protocolo> CR-LF
[Cabeceras]
CR-LF
Datos del pedido

El pedido enviado por el cliente solicita el establecimiento de una conexión con el servidor. Un pedido contiene: un método que debe ser aplicado a un recurso, el identificador URL de un recurso, la versión del protocolo HTTP en uso e información adicional opcional en forma de cabeceras que permiten incluir modificadores de petición. Según el tipo de método solicitado en el pedido, éste podrá o no llevar información en su parte de datos. CR-LF significa *Carriage Return-Line Feed*, es decir son los caracteres de control utilizados para indicar fin de línea y desplazamiento a una nueva.

Un método indica la acción que se debe ejecutar sobre el objeto identificado por el URL. En la tabla 1.17 se presentan un listado de los métodos más utilizados en HTTP y su respectiva función. [25]

HTTP define algunos tipos de cabeceras que se puede utilizar en el mensaje de pedido. Estas cabeceras utilizan el formato definido en el RFC 822, es decir "Nombre:", seguidos por un valor. El uso de las cabeceras en un pedido permite al cliente describir: tipos de recursos que puede recibir, características con las que un recurso puede ser transferido, modificadores a un método, etc. Un pedido puede contener varias cabeceras con su respectivo argumento, separadas entre si por un caracter CR (*Carriage Return*). El listado de cabeceras siempre termina en una línea en blanco (CR-LF) que lo separa de la parte de datos del pedido. Las cabeceras más utilizadas se presentan a la tabla 1.18.[26]

Método	Función
GET	Solicita al servidor le envíe el recurso especificado en el URL.
HEAD	Solicita se envíe la cabecera del documento solicitado en el URL y no el cuerpo del documento en sí.
PUT	Indica que la información enviada en la parte de datos del pedido debe ser almacenada en el servidor bajo el URL especificado. Si el URL al cual hace referencia el recurso ya existe en el servidor, éste será actualizado con el nuevo recurso enviado en el pedido. En definitiva PUT permite crear nuevos recursos en un servidor.
POST	Es utilizado para pedir al servidor de destino que acepte el recurso contenido en el campo de datos del pedido como un nuevo recurso subordinado al recurso identificado por el URL en el pedido. A diferencia de PUT, POST añade información a un recurso especificado, no crea uno nuevo. POST es usado por ejemplo para añadir un mensaje a un grupo de noticias, aumentar una lista de correo, añadir información a bases de datos, etc.
DELETE	Elimina del servidor la información correspondiente al URL especificado.
LINK	Establece un enlace de un recurso existente asociado al URL con otro recurso existente.
UNLINK	Remueve uno o varios enlaces que un recurso especificado en el URL pueda tener.

Tabla. 1.17. Métodos más utilizados en HTTP

Cabecera	Función
From:	Permite identificar un usuario con su dirección de correo y hacer que la respuesta enviada por el servidor al pedido sea en función de la identificación del usuario.
Accept:	Permiten especificar una lista de esquemas de representación que el cliente puede aceptar en la respuesta. Esta cabecera es muy similar a la cabecera *Content-Type*: utilizada en un mensaje MIME. Al igual que en MIME, tiene como argumento un tipo de archivo y un subtipo, Ejm. text/html, image/gif, audio/basic, video/mpeg, etc. Además en esta cabecera el cliente puede solicitar algunos parámetros asociados con el recurso tales como resolución, color, etc., con los cuales el servidor si es posible debería transmitir el recurso. El cliente aunque no lo especifique siempre será capaz de aceptar recursos del tipo/subtipo: text/plain y text/html. Ejm. *Accept: text/plain, text/html, image/gif.*
Accept-Encoding:	Similar a Accept, pero en este caso, el cliente especifica una lista de los tipos de codificación con que puede ser enviado un recurso desde el servidor. Los tipos de codificación más utilizados son: x-compress y x-gzip. Ejm. *Accept-Encoding: x-compress; x-gzip.*
Accept-Language:	Permite al usuario especificar una lista de idiomas que son preferibles en la respuesta. Una respuesta en un idioma no especificado es ilegal. Cada idioma tiene un código asignado según los estándares ISO 3316 y el ISO 639
User-Agent:	Si está presente especifica el programa y versión del *software* utilizado por el cliente.
Referer:	Es un campo opcional que permite especificar el URL del documento o el elemento dentro del documento desde el cual el URL al que hace referencia el pedido fue obtenido. De esta forma el servidor puede fácilmente regresar a documentos que generaron un enlace a otro documento.
Authorization:	Si está presente, contiene información de autenticación (nombre de usuario, clave, parámetros) requerida para que un usuario pueda acceder a un recurso.
If-Modified-Since:	Es una de las cabeceras más utilizadas junto con el método GET. Como argumento contiene una fecha y hora internacional referida al meridiano de Greenwich (Hora GMT). Permite que un recurso sea transferido desde el servidor al cliente únicamente si el recurso ha sido modificado después de la fecha y hora indicada. Si el recurso no ha sido modificado simplemente no se transferirá. Esto permite a un programa navegador utilizar recursos que se encuentran almacenados en su misma máquina y presentarlos en forma más rápida al usuario. Ejm. *If-Modified-Since: Fri, 24 May 1999 21:00:00 GMT.*

Tabla. 1.18. Principales cabeceras utilizadas en un pedido HTTP

b) Formato de una respuesta HTTP

La respuesta típica de un servidor tiene la siguiente estructura: [24]

Respuesta = <Versión del Protocolo><Código de estado><Frase de razón> CR-LF
[Cabeceras]
CR-LF
Datos de la respuesta

La respuesta enviada por el servidor contiene la misma versión de protocolo HTTP que el cliente solicitó en su pedido. El código de estado es un número decimal ASCII que permite al programa cliente saber el significado de una respuesta. La frase de estado

permite al usuario del programa cliente leer y entender qué significa una respuesta. Algunos de los códigos de estado y frases de estado más comunes se presentan a continuación: [25]

Código	Frase	Significado
1xx	*Reserved*	Reservado
2xx	*Success*	Resultado exitoso
201	*OK*	Exito, el contenido del recurso solicitado está en el campo de datos de la respuesta
202	*Accepted*	Aceptado para procesamiento posterior
204	*No Content*	Hecho, pero no hubo ningún contenido
3xx	*Redirection*	Redireccionamiento
300	*Multiple Choices*	Hay múltiples elecciones para un documento
304	*Not Modified*	Documento sin modificación
4xx	*Client error*	Error del cliente
401	*Unauthorized*	Necesita autorización para acceder a un documento
403	*Forbidden*	El documento existe pero está prohibido transferirlo
404	*Not Found*	El documento solicitado no existe
5xx	*Server error*	Error del servidor

Tabla. 1.19. Códigos y Frases de estado comunes en una respuesta HTTP

Cabecera	Función
Allowed:	Lista el conjunto de métodos que un usuario específico está permitido efectuar con ese servidor. Si se omite, el cliente por defecto puede utilizar únicamente los métodos GET y HEAD. Ejm. *Allow: GET HEAD PUT*
Public:	Lista los métodos que cualquier usuario puede utilizar. El valor por defecto, cuando se omite esta cabeeera, permite al usuario únicamente utilizar el método GET. Ejm. *Public: GET HEAD*
Content-Length:	HTTP a diferencia de SMTP permite transferir archivos en formato binario sin necesidad de convertirlos a ASCII. La cabecera *Content-Length* especifica con un número decimal ASCII la longitud de un archivo binario en octetos.
Content-Type:	Esta cabecera es la misma cabecera *Content-Type* definida en MIME. Permite especificar el tipo y subtipo de recurso contenido en el campo de datos de la respuesta. Para HTTP existen algunos nuevos subtipos no utilizados en el correo electrónico. Ejm. *Content-Type: text/html*
Content-Encoding:	Especifica el mecanismo de codificación utilizado. Los mecanismos más usados son: x-compress y x-gzip.
Date:	Indica la fecha y hora GMT en que un recurso fue creado. Ejm: *Date: Fri, 24 May 1999 22:30:20 GMT*
Expires:	Indica la fecha después de la cual el recurso deja de ser válido. Esto permite que un cliente pueda almacenar un recurso localmente por un período de tiempo durante el cual el rccurso no necesitará ser transferido desde el servidor remoto ya que se asegura no se modificará.
Last-Modified:	Indica la fecha de la última modificación de un recurso.
Version:	Define la versión de un recurso que varíe con el tiempo.
Content-Language:	Contiene el código ISO del idioma en que un documento está escrito
Cost:	Especifica el costo del acceso a este recurso, tal vez por derechos de autor.
Title:	Contiene el título del documento.

Tabla. 1.20. Principales cabeceras utilizadas en una respuesta HTTP

HTTP define un conjunto de cabeceras que utilizan el formato RFC 822, las cuales se pueden utilizar en forma opcional en el mensaje de respuesta. Estas cabeceras pueden ser cabeceras MIME tales como: *MIME-Version:, Content-Type:*, etc., y otras cabeceras propias de HTTP que permiten especificar características del recurso contenido en el campo de datos de la respuesta. El listado de cabeceras en la respuesta está separado del campo de datos por una línea en blanco (CR-LF) y separadas entre sí por el caracter (CR - *Carriage Return*). Cada cabecera posee un argumento. En la tabla 1.20 se presentan algunas de las cabeceras más utilizadas en una respuesta del servidor. [27]

Ejemplo 1.9.

Cuando un usuario haciendo uso de su navegador escribe o hace *click* sobre un hipertexto que posee un hipervínculo con la direccción URL hipotética: http://www.fie.epn.edu.ec/menu/info.html, DNS transforma el nombre de dominio www.fie.epn.edu.ec en la dirección IP correspondiente. A continuación HTTP envía un mensaje de pedido solicitando al servidor www.fie.epn.edu establezca una conexión con la máquina del usuario y proceda a transferir el recurso /menu/info.html. El mensaje de pedido sería semejante a:

```
GET /menu/info.html HTTP/1.0
<CR-LF>
```

La respuesta enviada por el servidor podría ser:

```
HTTP/1.0 200 OK
MIME -Version: 1.0
Date: Fri, 24 May 1999 22:30:20 GMT
Server: Apache/t.2.4
Content-Length: 192
Content-Type: text/html
<CR-LF>
    ....Datos del recurso
    ....Datos del recurso
```

Si el navegador detecta que posee el mismo nombre de recurso en el *cache* de la máquina local, el pedido enviado por HTTP podría tener la siguiente forma:

```
GET /menu/info.html HTTP/1.0
If-Modified-Since: Wed, 22 May 1999 19:23:00 GMT <CR>
<CR-LF>
```

Este pedido como se ha visto permitiría solicitar al servidor que el recurso indicado sea transferido únicamente si ha sido actualizado después de la fecha y hora indicada. Si el servidor posee una versión más actualizada del recurso lo enviará como se ha visto. Caso contrario enviará una respuesta indicando que el recurso no ha sido modificado semejante a:

```
HTTP/1.0 304 Not Modified
<CR-LF>
```

El navegador al recibir esta respuesta procederá a utilizar el recurso almacenado en el *cache* y presentarlo al usuario

c) HTTP 0.9, HTTP 1.0 y HTTP 1.1

En la actualidad existen tres versiones de protocolos HTTP: HTTP 0.9, HTTP 1.0 y HTTP 1.1. HTTP 0.9 fue la primera versión aparecida en 1991 y que actualmente ya no se la usa. HTTP 1.0 es hoy en día el protocolo más utilizado en el Web, sin embargo posee algunos inconvenientes. Uno de ellos, tal vez el más grave, es que requiere abrir y cerrar una conexión por cada recurso que desea transferir. Así por ejemplo, para transferir varios recursos que forman una misma página Web, HTTP 1.0 debe emplear una conexión individual para transferir cada uno de ellos, a pesar de que posiblemente todos los recursos requeridos se encuentren en el mismo servidor remoto. Esto resulta en una falta de eficiencia y en una demora en la presentación de la página completa al usuario. Además el alto número de pedidos causa que el tráfico en Internet sea alto en determinados enlaces.

La nueva versión HTTP 1.1, compatible con las anteriores, pero con muchas más opciones, cabeceras, características, etc. permitirá traer las páginas Web un poco más rápido al *browser* del usuario y reducir el tráfico existente en el Web. HTTP 1.1 aún no es utilizado ampliamente pero lo será cuando los servidores Web y los programas navegadores de Web de los usuarios lo incorporen.

HTTP 1.1 en vez de abrir y cerrar una conexión para transferir cada recurso individualmente provee una conexión persistente que permite realizar múltiples pedidos a un servidor, los cuales son almacenados o puestos en fila de espera en un *buffer* de salida. La capa TCP subyacente puede poner múltiples pedidos (y respuestas a pedidos) dentro de un segmento TCP que es enviado a la capa IP para su transmisión en datagramas. Debido a que el número de pedidos disminuye, el número de paquetes que fluye a través de Internet es menor, lo cual como consecuencia reduce el tráfico y mejora el desempeño.

HTTP 1.1 también permite utilizar compresión y descompresión de archivos HTML, lo cual reduce en forma sustancial la cantidad de datos que debe transmitirse en el Internet.

Además de las conexiones persistentes y otras mejoras de desempeño, HTTP 1.1 también provee la habilidad de que un mismo servidor pueda tener múltiples nombres de dominios compartiendo la misma dirección IP. Esto simplificará el procesamiento de servidores Web que administran simultáneamente varios sitios Web, lo cual a menudo se llama *virtual hosting*.

1.6.10.4 *HyperText Markup Language* (HTML)

Una página Web está escrita en un lenguaje denominado HTML (*HyperText Markup Language*). HTML permite a un creador de páginas Web:

- Publicar documentos con títulos, texto, tablas, listas, fotos, etc.
- Recuperar información a través de enlaces de hipertexto, haciendo *click* sobre él.
- Diseñar formularios para ser utilizados en transacciones con servicios remotos, para uso en búsquedas de información, hacer reservaciones, ordenar productos, etc. Incluir hojas de cálculo, vídeo clips, audio y otras aplicaciones directamente en sus documentos.

HTML ha sido diseñado con la visión de que todo tipo de dispositivo sea capaz de utilizar información en el Web: PCs con monitores de distinta resolución y cantidad de

colores, teléfonos celulares, dispositivos de mano (*hand-held*), computadores con gran y pequeño ancho de banda, etc.

HTML es un lenguaje basado en un conjunto de etiquetas delimitadoras que indican al programa navegador que recibe una página cómo debe formatearla para presentarla al usuario. Existen muchas etiquetas, algunas de las cuales permiten insertar encabezados, títulos, añadir textos, dar formato al texto, añadir imágenes, añadir hipervínculos a otras páginas, etc. Algunas de las etiquetas más usadas se presentan a continuación: [1]

Tag	Descripción
<HTML>...</HTML>	Declara que el contenido es una página Web
<HEAD>...</HEAD>	Delimita el encabezado principal de una página
<TITLE>...</TITLE>	Define el título de una página, el cual no se lo muestra en la página
<BODY>...</BODY>	Delimita el cuerpo de la página
<Hn>...</Hn>	Delimita un encabezado secundario
<B>...</B>	Pone en **negritas** el contenido (...)
<I>...</I>	Pone en *cursivas* el contenido (...)
<UL>...</UL>	Delimita una lista no numerada
<OL>...</OL>	Delimita una lista numerada
<MENU>...</MENU>	Delimita un menú
<LI>	Indica el inicio de un ítem de una lista
 	Indica el inicio de una línea
<P>	Inicio de un párrafo
<HR>	Traza una línea horizontal
<PRE>...</PRE>	Texto preformateado
<IMG SRC="...">	Inserta en la página la imagen (...)
<A HREF="...">...</A>	Define un hipervínculo con el URL (...)

Tabla. 1.21. Algunas de las principales etiquetas usadas en HTML

En la actualidad no es necesario conocer estas etiquetas pues existen programas muy fáciles de usar que permiten diseñar fácil y rápidamente páginas Web a través de una interfaz muy simple de utilizar para el usuario. El programa se encarga de compilar automáticamente el contenido de la página al formato HTML. Algunos de estos programas son: *Microsoft Front Page 98*, *HotMetal 4.0*, etc. Mediante estos programas el usuario únicamente requiere de imaginación para poder crear una página Web.

Además en la actualidad, se puede añadir mejor presentación a las páginas Web realizadas con HTML haciendo uso de otro lenguaje denominado *Java*. *Java* trabaja conjuntamente con HTML y permite añadir más recursos de multimedia a una página Web: imágenes dinámicas, textos en movimiento, audio, vídeo, etc.

1.6.11 *Internet Relay Chat* (IRC)

El *Internet Relay Chat* (IRC) es una parte del Internet donde un usuario puede comunicarse con otra persona o con un grupo de personas simultáneamente, conectadas en ese momento a la red, escribiendo mensajes en su computador, los cuales son inmediatamente presentados a estas otras personas en sus respectivas pantallas de computador en cualquier lugar del mundo. Esta nueva forma de comunicación se la ha denominado "*chatear*" y se está difundiendo en forma rápida en la red.

EL IRC en sí es un protocolo basado en el modelo cliente-servidor. Cuando un usuario desea "chatear" con uno u otros usuarios conectados en ese momento a la red, necesita conectarse a un servidor IRC, utilizando un programa cliente de IRC. No es necesario que un usuario se conecte directamente al mismo servidor al cual la persona con que desea "chatear" se encuentra conectada. Simplemente, debe conectarse a cualquier servidor IRC, generalmente el más cercano al usuario. Una vez conectado a un servidor IRC, mediante el programa cliente, el usuario puede unirse a un canal.

Un canal es parecido a un grupo de discusión, en el cual el usuario puede comunicarse con gente conectada a ese mismo canal y empezar a "chatear" con ellos acerca de algún tema que se esté tratando. Un canal es creado implícitamente cuando el primer usuario se conecta a él y finaliza cuando el último cliente lo abandona. Mientras el canal existe, cualquier cliente puede unirse a él utilizando el nombre del canal. El nombre de un canal está precedido por el símbolo #. Ejm. #chat. Una vez que el cliente se ha unido a un canal, puede iniciar el intercambio de mensajes en tiempo real. El servidor IRC al cual se encuentra conectado el cliente, recibe sus mensajes y se encarga de enviarlos a otros servidores de la red IRC. Los servidores de destino se encargan de entregar el mensaje a cada uno de sus clientes que pertenezcan al canal al cual va direccionado el mensaje recibido. IRC asegura que todos los clientes de un canal recibirán todos los mensajes direccionados a ese canal.

Un servidor IRC provee un punto en el cual los clientes se conectan para conversar con otros clientes y un punto al cual se conectan otros servidores. El conjunto de todos los servidores IRC forman la red IRC, la cual es un subconjunto de Internet. La red IRC

posee una configuración tipo árbol, donde cada servidor actúa como un nodo central para la parte de la red que ve, como se muestra a continuación:

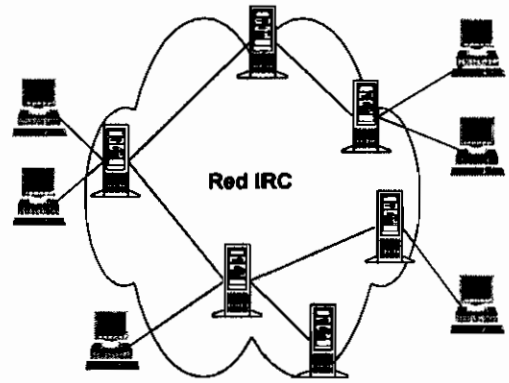


Fig. 1.31. Topología tipo árbol de los servidores en la red IRC

En IRC a cada cliente de un canal se lo identifica en forma única mediante un sobrenombre. A más del sobrenombre, todos los servidores deben tener la siguiente información de todos los clientes: el nombre real del *host* en el cual el cliente se está ejecutando, el nombre de usuario del cliente en ese *host*, y el servidor al cual el cliente está conectado.

En una red IRC se definen clientes especiales, denominados operadores, a los cuales se les permite ejecutar funciones generales de administración de un canal y de mantenimiento de la red IRC. Generalmente el primer cliente que se une a un canal es denominado operador de ese canal. Al operador del canal se lo reconoce porque posee el símbolo @ antepuesto a su sobrenombre. Un operador es el encargado de proponer el tópico sobre el cual los clientes que se unan a ese canal podrán "chatear". El operador tiene la capacidad de remover a un usuario conectado a ese canal, cerrando la conexión entre ese cliente y un servidor. Esto por ejemplo permitiría a un operador por pedido de otros usuarios o por cuenta propia, retirar a un cliente que esté siendo ofensivo o se lo considere no deseado dentro de un canal. Además el administrador puede establecer algunos modos según los cuales un cliente puede unirse a un canal. Por ejemplo solicitándole una clave, etc. A más de estas tareas, un operador podría ser capaz de ejecutar tareas básicas de red, consideradas como peligrosas, tales como: desconectar y reconectar un servidor cuando surjan inconvenientes en la red.

IRC es el protocolo que define como procede la comunicación entre un cliente y un servidor o entre servidores en una red IRC. Esta comunicación, asincrónica por naturaleza, se basa en el intercambio de mensajes. Un mensaje IRC está formado por tres partes: un prefijo, que es opcional, un comando, y un conjunto de parámetros de ese comando separados entre sí por uno o más caracteres ASCII de espacio en blanco (0x20). Todo mensaje está terminado en los caracteres CR-LF y puede tener una longitud máxima de 512 caracteres incluidos los 2 caracteres CR-LF de fin de línea. CR-LF permite delimitar los mensajes dentro del flujo de octetos que se recibe. [28]

Mensaje:: [<Prefijo>] <Comando> <Parámetros> <CR-LF>

El prefijo es utilizado por los servidores para indicar el verdadero origen de un mensaje. Los clientes cuando envían un mensaje no utilizan un prefijo. El servidor se encarga de añadir el prefijo. El prefijo contiene el nombre del servidor IRC, el sobrenombre del usuario, el nombre real del usuario en ese *host* y el nombre del *host* en el cual está ejecutándose el programa cliente del usuario.

Las partes Comando y Parámetros están relacionadas entre sí. La parte Comando contiene el nombre de un comando IRC y la parte Parámetros contiene parámetros asociados a ese comando. IRC define un sinnúmero de comandos que permiten especificar al inicio de una sesión: el nombre de usuario, nombre del *host*, nombre del servidor, nombre real del usuario. Además otros comandos permiten: especificar si el servidor se está comunicando con un cliente o con un servidor, asignar privilegios de operador a un usuario, unirse a un canal, realizar tareas de administración como operador, enviar mensajes entre usuarios, finalizar una sesión, etc. El detalle de todos los comandos, su sintaxis, sus parámetros y respuestas a cada comando se encuentran detallados en el RFC 1459 que describe el protocolo IRC.

IRC permite varios tipos de comunicación: uno a uno, uno a muchos y uno a todos. Todas éstas a su vez basan su funcionamiento en la topología tipo árbol de la red IRC.

Todo cliente envía y recibe mensajes desde el servidor al cual está conectado. No necesita establecer conexiones con servidores remotos a fin de entregar o recibir un mensaje.

En una comunicación uno a uno, el servidor al cual está conectado el cliente de origen se encarga de enviar los mensajes del cliente a través de la ruta más corta a lo largo del árbol IRC hasta llegar al servidor destino, haciendo que el mensaje pase únicamente por aquellos servidores que se encuentren en esta ruta más corta. El servidor destino entregará al cliente destino el mensaje.

Ejemplo 1.10.

En la figura 1.32, un mensaje enviado por el cliente 1 al cliente 2, sólo pasa a través del servidor A, el cual lo entrega directamente al cliente 2. De forma similar, un mensaje entre los clientes 1 y 3 sólo pasa a través de los servidores A y B y el cliente 3. Ningún otro servidor o cliente recibe este mensaje. Finalmente, un mensaje entre los clientes 2 y 4, sólo pasa a través de los servidores A, B, C, D y el cliente 4.

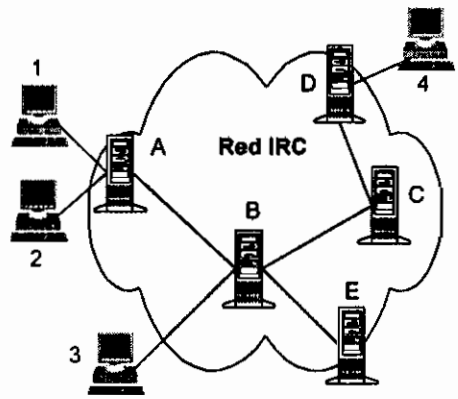


Fig. 1.32. Transmisión de mensajes en IRC

En una comunicación uno a muchos, una forma ineficiente de entrega del mensaje sería enviar el mensaje individualmente a cada destino a través de la ruta más corta en el árbol. La forma más adecuada es el envío de un mismo mensaje a través de la ruta más

corta hacia los servidores de destino, el servidor de destino puede tener varios clientes que se encuentren en un mismo canal, el servidor entregará una copia del mensaje a cada cliente. En una comunicación uno a todos, se utiliza la difusión del mensaje a todos los servidores de la red IRC.

Uno de los programas clientes más utilizados en la actualidad es el mIRC. Este programa se encarga de: permitir escoger el servidor más cercano, solicitar nombres de usuario, sobrenombres, acceder a canales disponibles, "chatear", etc. Otra aplicación, la cual no es un programa cliente de IRC, pero que está relacionada es el ICQ ("I seek you"). Esta aplicación permite determinar si amigos de un usuario se encuentran conectados a la red en el mismo momento que él. Tanto el usuario como las personas que le interesa saber si están conectados deberían tener el mismo programa ejecutándose. Con ICQ se puede luego iniciar una sesión de *chat*, iniciar juegos interactivos a través de la red con estas otras personas, etc.

1.6.12 Otras Aplicaciones

Las aplicaciones anteriores no son las únicas que existen. A más de las mencionadas existen muchas otras a las cuales día a día se suman más y más. A continuación se describen en forma muy corta algunas de estas otras aplicaciones existentes. [3]

***Simple Network Management Protocol* (SNMP):** Es una aplicación muy utilizada que permite, a un administrador de red, administrar y controlar el funcionamiento de una red TCP/IP. SNMP define el protocolo encargado de realizar la administración de distintos dispositivos tales como: *hosts*, ruteadores, puentes, impresoras, etc. y todos aquellos otros que sean susceptibles de ser administrados. SNMP describe el tipo de información que debe mantener cada dispositivo a fin de que pueda ser administrado y la forma como interactúa cada uno de estos dispositivos con la estación administradora.

***Network File Systems* (NFS):** Permite que un sistema pueda acceder a archivos en otra computadora remota de forma más transparente que mediante FTP. NFS crea la ilusión que los discos u otros dispositivos de un sistema estuviesen directamente conectados a

otro sistema. No hay necesidad de usar una aplicación especial para acceder a un archivo en otro sistema. El sistema simplemente cree que posee *drives* adicionales. Esta aplicación, por ejemplo, permite colocar discos duros de gran capacidad en pocas computadoras en una red local, pero dar acceso a otras computadoras a espacio en estos discos. También podría utilizarse en redes que utilizan terminales sin disco, permitiendo que puedan almacenar su información en servidores de archivos, lo cual permite reducir los costos y facilitar el mantenimiento de la red.

Impresión Remota: Permite a un usuario tener acceso a impresoras en otras computadoras como si estuviesen directamente conectadas a la del usuario. Esta aplicación se basa en el protocolo *LinePrinter*.

Ejecución Remota: Permite a un usuario ejecutar un programa de aplicación en un computador diferente. Esto puede ser útil cuando el usuario desea ejecutar una aplicación que requiere de recursos que su computador no dispone, pero que otros computadores más grandes sí. Puede utilizar los protocolos REXEC y RSH. REXEC permite definir la identificación del usuario, un *password*, dirección IP del *host* y el proceso que será iniciado en el *host* remoto. REXEC permite ejecutar comandos en el servidor remoto. RSH al igual que REXEC permite ejecutar comandos en el *host* remoto pero sin necesidad de requerir un *password*.

***Remote Procedure Call* (RPC):** El protocolo de llamada de procedimiento remoto RPC permite a un programa llamar a subrutinas que se ejecutan en un sistema remoto. El programa que llama a la subrutina, el cliente, envía un mensaje de llamada a un servidor de procesos y espera por un mensaje de respuesta. El mensaje de llamada contiene los parámetros del procedimiento. El servidor recibe el mensaje, extrae los parámetros del procedimiento, efectúa el procesamiento solicitado y envía un mensaje de respuesta que contiene los resultados del procedimiento.

1.7 *INTERNET PROTOCOL version 6* (IPv6)

IPv4 ha trabajado muy bien desde su creación en los años 60 y se ha adaptado a grandes cambios tecnológicos que han surgido desde entonces, sin embargo a pesar de su gran flexibilidad IPv4 posee limitaciones que han motivado a reemplazarlo en un futuro cercano, entre las cuales están:

- Inminente agotamiento del espacio de direcciones IP
- Falta de soporte a nuevas aplicaciones tales como audio y vídeo en tiempo real que requieren garantías en los retardos y un mecanismo que reserve ancho de banda para el flujo de datagramas.
- Falta de comunicaciones seguras, no es posible autenticar al emisor de un datagrama.

El nuevo protocolo IPv6 propuesto como reemplazo a IPv4, mantiene la mayoría de características de IPv4, pero introduce nuevos cambios tales como:

- **Direcciones más largas:** IPv6 emplea direcciones de 128 bits en vez de direcciones de 32 bits usadas por IPv4. El espacio de direcciones IP de IPv6 es tan grande que no se prevé su agotamiento futuro.
- **Formato de cabeceras flexible:** IPv6 a diferencia de IPv4 emplea un nuevo formato de datagrama formado por una cabecera fija de 40 bytes y cabeceras opcionales. IPv6 elimina o hace opcionales algunos de los campos fijos de la cabecera de IPv4, tales como los de fragmentación realizada por IPv4, mejorando la velocidad de procesamiento del datagrama.
- **Soporte mejorado para extensiones y opciones:** IPv6 permite incluir información de control en cabeceras opcionales (fragmentación, autenticación, ruteo, opciones salto a salto, opciones extremo a extremo) y tiene la posibilidad de adaptarse a nuevas tecnologías de red subyacente o a nuevas aplicaciones añadiendo cabeceras opcionales.
- **Soporte para asignación de recursos:** IPv6 permite a los paquetes que pertenecen a un flujo de tráfico particular ser etiquetados para recibir un manejo especial. IPv6 permite la preasignación de recursos de red a fin de soportar aplicaciones como

audio y vídeo en tiempo real que requieren una garantía de ancho de banda y retardo.

- **Autenticación y privacidad:** IPv6 soporta cabeceras adicionales para realizar autenticación, integridad de datos, y confidencialidad de datos.

El formato del datagrama IPv6 se presenta a continuación:

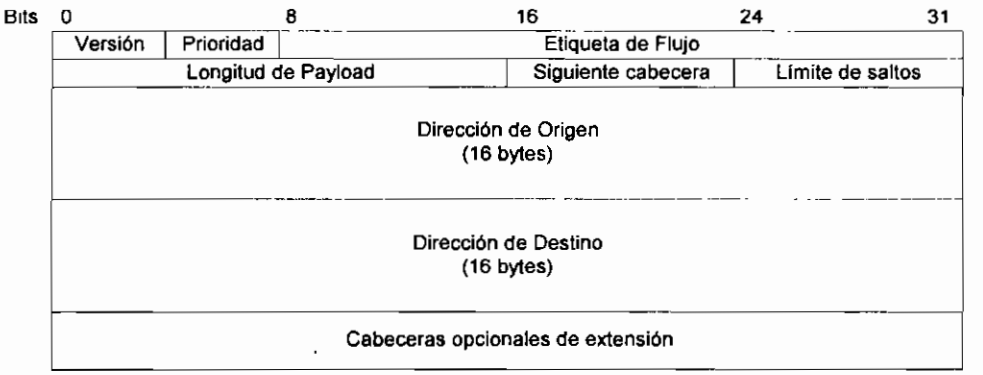


Fig. 1.33. Formato del datagrama IPv6 [3]

- **Versión:** Contiene el valor 6, que indica la versión del protocolo IP.
- **Prioridad:** Permite a una fuente especificar la clase de tráfico para el datagrama. Los valores de prioridad están divididos en dos rangos:

  0-7: Especifica una prioridad para el tráfico que no es en tiempo real a fin de realizar control de congestión (0 menor, 7 mayor)

  8-15: Especifica la sensibilidad al tiempo de retardo del tráfico en tiempo real o, velocidad constante. (8 menor, 15 mayor)
- **Etiqueta de Flujo:** Puede ser usado por un *host* para solicitar a los ruteadores IPv6 manejo especial para ciertos flujos de datagramas. Un flujo de datagramas se lo identifica por la dirección IP de la fuente y una etiqueta de flujo distinta de cero. En caso de que un *host* o ruteador no soporte esta función de flujo, este campo contiene todo cero y es ignorado. Este campo permite a un ruteador determinar cómo rutear y enviar el datagrama sin necesidad de examinar el resto de la cabecera IPv6, lo cual resulta muy útil para aplicaciones en tiempo real.
- **Longitud de *payload*:** Indica la longitud en octetos que siguen a la cabecera fija de 40 bytes (*Payload*).

- **Siguiente Cabecera:** Indica la cabecera (opcional o de protocolo superior) que sigue inmediatamente después a la cabecera fija de IPv6. Ejm. Opciones salto a salto, ruteo, fragmentación, autenticación, ICMP, TCP, UDP, etc. Cada cabecera opcional contiene también un campo de Siguiente Cabecera indicando la siguiente cabecera de opción o protocolo, tal como se muestra a continuación:

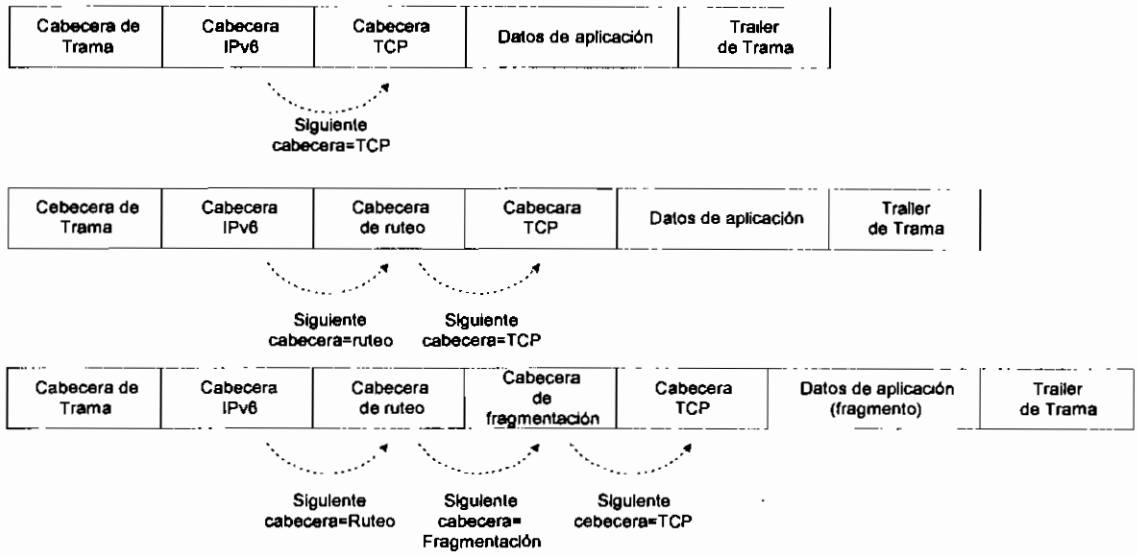


Fig. 1.34. Operación del campo siguiente cabecera de IPv6. [3]

- **Límite de saltos:** Indica el máximo número de saltos que un datagrama puede realizar antes de ser desechado.
- **Dirección de Origen:** Contiene la dirección IP de origen (128 bits).
- **Dirección de Destino:** Contiene la dirección IP de destino (128 bits).
- **Cabeceras opcionales:** Contiene las cabeceras de extensión opcionales. Cada opción posee su formato especifico, pero en general poseen la estructura de la figura 1.35, en la cual se identifica el tipo de cabecera opcional, su longitud total y valores de opciones asociados a ésta.

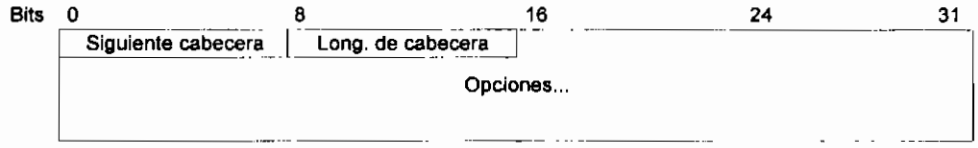


Fig. 1.35 Formato general de una cabecera opcional. [3]

El RFC 1883 recomienda el orden en que deben ser colocadas las cabeceras de extensión, después de la cabecera fija IPv6, tal como se muestra a continuación:

Cabecera de extensión	Descripción
Opciones Salto a Salto	Lleva información que todos los ruteadores a lo largo del destino deben examinar. Solo una opción está definida, la cual permite el transporte de jumbogramas (datagramas de más de 65535 octetos)
Ruteo	Lista uno o más ruteadores por lo cuales debe ser enviado el datagrama
Fragmentación	Permite realizar fragmentación del datagrama al igual que IPv4. En IPv6, el *host* de origen debe descubrir la MTU a lo largo de la ruta hacia el destino y realizar la fragmentación si es necesaria. Los ruteadores no realizan fragmentación en IPv6, a fin de acelerar el ruteo.
Autenticación	Provee un mecanismo por el cual el receptor puede saber quien envió un datagrama y si no fue alterada la integridad del mismo. (Usa algoritmo MD5). El *payload* es enviado sin encriptación.
Datos seguros encriptados	Contiene contenidos encriptados que solo el destino adecuado puede leerlos. (Usa algoritmo DES)
Opciones de destino	Contiene información opcional que solo el destino debe examinarla. Aún no tiene uso.

Tabla. 1.22. Orden recomendado de las cabeceras opcionales en el datagrama IPv6

Las direcciones IPv6 (128 bits-16 bytes) se las representa en ocho grupos de cuatro dígitos hexadecimales, separados entre si mediante dos puntos (:). Ejemplo:

6000:0000:0000:0000:0321:5234:0AB5: FE3A

La nomenclatura de las direcciones IPv6 pueden ser simplificadas de la siguiente forma:

1. Los grupos de ceros consecutivos dentro de la dirección pueden ser omitidos y reemplazados por dos puntos (una sola vez en toda la dirección), y los ceros a la izquierda de un grupo pueden ser omitidos. Ejemplo:

   6000::321:5234:AB5:FE3A

2. Durante la transición de IPv4 a IPv6, las direcciones IPv4 pueden ser escritas como un par de dos puntos seguidas por la notación *dotted decimal* normal de IPv4. Ejemplo:

   ::192.168.10.10

A diferencia de IPv4, el RFC 1884 define tres tipos de direcciones para IPv6:

- ***Unicast***: Identifica una única interfaz de red. Un paquete es únicamente entregado a esa dirección de red.
- ***Anycast***: Identifica un conjunto de interfaces de red. Un paquete es entregado a cualesquiera de esas interfaces de red, no a todas.
- ***Multicast***: Identifica un conjunto de interfaces de red. Un paquete es entregado a todas las interfaces de red identificadas por la dirección. La función de *broadcast* puede ser realizada como una extensión de la definición de *multicast*.

El RFC 1884 ha dividido el espacio de direcciones de IPv6 en varios tipos diferenciados mediante un prefijo, tal como se muestra a continuación:

Fig. 1.36. Formato general de las direcciones IPv6 [3]

Prefijo (binario)	Asignación
00000000	Reservado
00000001	No asignado
0000001	Reservado para asignación NSAP - OSI
0000010	Reservado para direcciones Novell Netware IPX
0000011	No asignado
00001	No asignado
0001	No asignado
001	No asignado
010	Direcciones Unicast basadas según proveedor de servicio
011	No asignado
100	Reservado para asignación basada en ubicación geográfica
101	No asignado
110	No asignado
1110	No asignado
11110	No asignado
111110	No asignado
1111110	No asignado
11111110	No asignado
1111111010	Direcciones para uso de enlace local
1111111011	Direcciones para uso local
11111111	Multicast

Tabla. 1.23 División propuesta para el espacio de direcciones de IPv6 [2]

El detalle de esta clasificación está fuera del alcance de este trabajo y se sugiere revisar el RFC 1884 donde se describe en detalle. Las direcciones IPv6, en forma semejante a las subredes de IPv4, podrán ser divididas en niveles de jerarquía, (Ejm. identificador de proveedor, de red, identificador de subred, etc.).

IPv6 involucrará la realización de cambios, modificaciones, en varios de los protocolos TCP/IP y relacionados con ellos (Ejm. protocolos de LAN, ARP, ICMP, RIP, OSPF, relación con TCP/UDP, DNS, aplicaciones, etc.). No es el objeto de este trabajo analizar cada uno de estos aspectos, más bien se recomienda revisar los RFC que periódicamente aparecerán con nueva información al respecto de IPv6.

Durante varios años más IPv4 será la base del Internet, y las nuevas redes IPv6 que vayan surgiendo deberán mantener total compatibilidad con toda la infraestructura actual de IPv4.

REFERENCIAS (Capítulo I)

[1] Tanenbaum, Andrew S.: "*Computer Networks* ", Third Edition, Prentice Hall, New Jersey, 1996

[2] Comer, Douglas E.: "*Redes globales de información con Internet y TCP/IP*", Prentice Hall, México, 1996.

[3] http://www.redbooks.ibm.com, "*TCP/IP Tutorial and Technical Overview*", IBM

[4] "*Nonstandard for transmission of IP datagrams over serial lines: SLIP* ", RFC 1055, June 1988

[5] "*Compressing TCP/IP headers for low-speed serial links*", RFC 1144, February 1990

[6] "*The Point to Point Protocol (PPP)*", RFC 1661, July 1994

[7] "*The Point-to-Point Protocol (PPP) for the Transmission of Multi-protocol Datagrams over Point-to-Point Links*", RFC 1331, May 1992

[8] "*The PPP Internet Protocol Control Protocol (IPCP)*", RFC 1332, May 1992

[9] "*Address Allocation for Private Internets*", RFC 1597, March 1994.

[10] http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/index.htm,"*Internetworking Technology Overview*"

[11] "*Transmission Control Protocol*", RFC 0793, September 1981

[12] "*Domain Names - Concepts and Facilities*", RFC 1034, November 1987

[13] http://www.hill.com/library/tcpip.html, "*An Overview of TCP/IP Protocols and the Internet*"

[14] "*Domain Names - Implementation and Specification*", RFC 1035, November 1987

[15] "*Telnet Protocol Specification*", RFC 0854, May 1983

[16] "*File Transfer Protocol*", RFC 0959, October 1985

[17] "*Trivial File Transfer Protocol.*", RFC 1350, July 1992

[18] "*Post Office Protocol - Version 3*", RFC 1939, May 1996

[19] "*Network News Transfer Protocol*", RFC 977, February 1986

[20] "*The Internet Gopher Protocol*", RFC 1436, March 1993

[21] "*Uniform Resource Locators (URL)*", RFC 1738, December 1994

[22] http://www.acl.lanl.gov/URI/uri.html, "*Uniform Resource Identifiers (URIs)*"

[23] "URN Syntax", RFC 2141, May 1997

[24] Vass J. Harwell J.: "*The World Wide Web*", IEEE Potencials, October/November 1998

[25] "*Hypertext Transfer Protocol -- HTTP/1.0*", RFC 1945, May 1996

[26] http://www.w3.org/Protocols/HTTP/HTRQ_Headers.html, "*HTTP Request fields*"

[27] http://www.w3.org/Protocols/HTTP/Object_Headers.html, "*Object Metainformation*"

[28] "*Internet Relay Chat Protocol*", RFC 1459, May 1993

INDICE - CAPITULO II

CAPITULO II

PROTOCOLOS RELACIONADOS CON EL ACCESO A INTERNET

En este capítulo se revisarán ciertos protocolos adicionales que servirán de base para el análisis de las distintas tecnologías de acceso a Internet, detalladas en el capítulo IV, y para el entendimiento de mecanismos de autenticación y tarifación de usuarios en un Proveedor de Servicios de Internet (ISP).

2.1 PROTOCOLOS DE AUTENTICACION PPP (PAP y CHAP)

PPP define un método para transportar datagramas multiprotocolo sobre enlaces seriales punto a punto. Para establecer una comunicación sobre un enlace PPP, cada extremo del enlace debe enviar paquetes LCP a fin de configurar el enlace. Una vez que el enlace ha sido establecido y configurado, PPP provee una fase opcional de autenticación antes de proceder a la fase de configuración del protocolo de capa red.

Durante la fase de configuración del enlace, el protocolo LCP de PPP permite negociar un protocolo de autenticación en caso de que se desee realizar la fase opcional de autenticación.

Se han definido dos posibles protocolos de autenticación que pueden ser transportados en el campo de datos de la trama PPP durante la fase opcional de autenticación. Estos son: [1]

1. PAP (*Password Authentication Protocol*)
2. CHAP (*Challenge Handshake Authentication Protocol*)

Se define como *peer* al extremo del enlace PPP que debe autenticarse. El extremo que se encarga de comprobar si la información de autenticación es verdadera se denomina autenticador.

2.1.1 *Password Authentication Protocol* (PAP)

El protocolo de autenticación de *password* (PAP - *Password Authentication Protocol*) provee un método simple para que el *peer* pueda identificarse ante el autenticador, mediante el intercambio de dos mensajes de *handshake*. La autenticación mediante PAP se la realiza una sola vez durante la fase de autenticación PPP y nunca más a lo largo de la vida de la conexión.

Durante esta fase de autenticación, el *peer* PAP envía repetidamente un par identificador/*password* hasta que el autenticador le envíe un acuse de recibo positivo autenticando al *peer*, o hasta que la conexión sea terminada.

PAP no es un método seguro de autenticación ya que los *passwords* son enviados sobre el circuito sin ser de ninguna forma encriptados.

El paquete PAP viaja encapsulado en el campo de datos de una trama PPP durante la fase opcional de autenticación. El campo Protocolo de la trama PPP contiene el valor hexadecimal c0123 para identificar que en su campo de datos existe un paquete PAP. El formato de un paquete PAP se presenta a continuación:

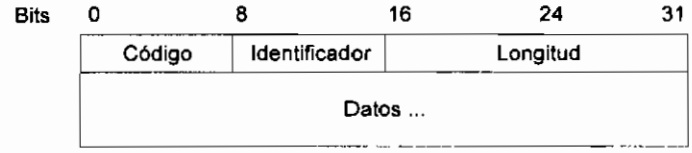


Fig. 2.1. Formato de un paquete PAP [1]

- **Código:** Indica el tipo de paquete PAP. Se definen los siguientes tipos de paquetes:

Código	Tipo
1	*Authenticate-Request*
2	*Authenticate-Ack*
3	*Authenticate-Nak*

Tabla. 2.1. Tipos de paquetes PAP [1]

- **Identificador:** Contiene un valor que permite asociar pedidos con respuestas.
- **Longitud:** Indica la longitud en octetos del paquete PAP incluyendo la cabecera.
- **Datos:** Contiene cero o más octetos cuyo formato depende del código del paquete.

2.1.1.1 Operación de PAP

Terminada la configuración del enlace y escogido PAP como protocolo de autenticación, el *peer* envía un paquete *Authenticate-Request* durante la fase de autenticación. El paquete *Authenticate-Request* es reenviado tantas veces hasta que un paquete de respuesta sea enviado o hasta que un contador expire.

El paquete *Authenticate-Request* contiene en el campo de datos la siguiente información:

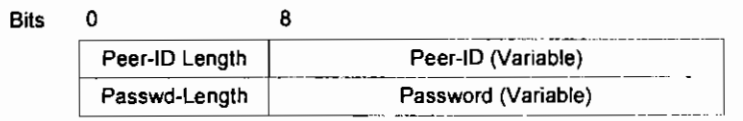


Fig. 2.2. Campo de datos de un paquete *Authenticate-Request* [1]

- ***Peer-ID-Length*:** Indica la longitud en octetos del campo Peer-ID.
- ***Peer-ID*:** Contiene en cero o más octetos el nombre del *peer* que será autenticado.
- ***Passwd-Length*:** Indica la longitud en octetos del campo *Password.*
- ***Password*:** Contiene en cero o más octetos el *password* del *peer*, a ser autenticado.

Si el par *Peer-ID/Password* recibidos por el autenticador en el *paquete Authenticate-Request* son ambos reconocibles y válidos, el autenticador envía un paquete *Authenticate-Ack*, caso contrario un paquete *Authenticate-Nak*. Los paquetes *Authenticate-Ack* y *Authenticate-Nak* pueden contener en el campo de datos cero o más octetos correspondientes a un mensaje en formato ASCII que será mostrado al usuario indicándole el éxito o fallo de la autenticación.

2.1.2 *Challenge-Handshake Authentication Protocol* (CHAP)

El protocolo CHAP (*Challenge-Handshake Authentication Protocol*) es utilizado para periódicamente verificar la identidad del *peer* utilizando un intercambio de tres mensajes de *handshake*. Este intercambio es hecho durante el establecimiento del enlace y puede ser repetido a cualquier instante después de que el enlace ha sido establecido. CHAP es un protocolo de autenticación mucho más seguro que PAP, ya que nunca transmite secretos, tales como *passwords*, en forma visible como lo hace PAP.

CHAP depende de un secreto compartido únicamente entre el *peer* y el autenticador, el cual nunca será enviado en forma visible sobre el enlace. El mecanismo de autenticación utilizado por CHAP se basa en el intercambio de desafíos y respuestas. Una vez terminada la configuración del enlace y escogido CHAP como protocolo de autenticación, el autenticador envía al *peer* un desafío conteniendo un valor aleatorio y de longitud variable, el cual debe poseer características de unicidad e impredecibilidad. El *peer* debe responder al desafío enviando un paquete, el cual contiene un valor calculado utilizando una función "*one-way-hash*" sobre una cadena específica de octetos, la cual incluye campos de la cabecera del paquete CHAP, el secreto compartido y el valor generado aleatoriamente enviado en el desafío.

Una función "*one-way-hash*" posee tres propiedades: [2]

1. Dado P, una cadena de longitud variable de octetos, es fácil computar una función MD(P)
2. Dado MD(P), es computacionalmente imposible encontrar P.
3. Es imposible que dos mensajes distintos (P_1 y P_2) generen el mismo resultado MD(P).

Un algoritmo que cumple las tres condiciones anteriores, y el más utilizado, es el denominado MD5 (*Message Digest 5*), descrito en el RFC 1321, el cual crea a partir de una cadena arbitraria de octetos un resultado único, conocido como "*hash*", consistente de 128 bits (16 octetos).

El algoritmo "*one-way-hash*" es utilizado, ya que garantiza que computacionalmente es imposible determinar el secreto a partir del desafío conocido y el valor de la respuesta.

Una vez que el autenticador recibe el mensaje de respuesta desde el *peer*, chequea la respuesta con su propio valor calculado del esperado valor *hash*. Si los valores coinciden, la autenticación es acusada positivamente, caso contrario la conexión es terminada.

Además, para evitar que un intruso que haya interceptado la respuesta correspondiente a un desafío, quiera intentar usarla para infiltrarse, cada nuevo paquete de desafío enviado a un *peer* debe contener un valor aleatorio jamás usado en un paquete anteriormente enviado, a pesar de que haya sido utilizado hace mucho tiempo atrás.

El paquete CHAP viaja encapsulado en el campo de datos de una trama PPP durante la fase opcional de autenticación. El campo Protocolo de la trama PPP contiene el valor hexadecimal C0223 para identificar que en su campo de datos existe un paquete CHAP. El formato de un paquete CHAP se presenta a continuación:

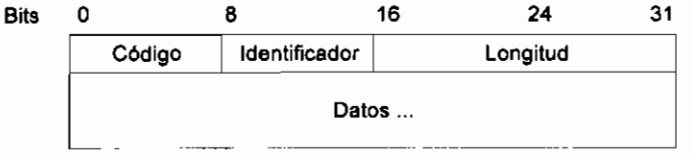


Fig. 2.3. Formato de un paquete CHAP [1]

- **Código:** Indica el tipo de paquete CHAP. Se definen los siguientes tipos de paquetes:

Código	Tipo
1	*Challenge*
2	*Response*
3	*Success*
4	*Failure*

Tabla. 2.2. Tipos de paquetes CHAP [1]

- **Identificador:** Contiene un valor que permite asociar pedidos con respuestas.
- **Longitud:** Indica la longitud en octetos del paquete CHAP incluyendo la cabecera.
- **Datos:** Contiene cero o más octetos cuyo formato depende del código del paquete.

2.1.2.1 Operación de CHAP

Como se describió anteriormente, una vez terminado el establecimiento del enlace y configurado CHAP como protocolo de autenticación, el autenticador debe enviar un paquete *Challenge* al *peer*. Este paquete es reenviado tantas veces hasta que un paquete *Response* sea recibido o hasta que un contador expire. El paquete *Challenge* también puede ser enviado durante el tiempo de vida de la conexión para verificar que ésta no haya sido alterada.

Los paquetes *Challenge/Response* contienen en el campo de datos los siguientes campos:

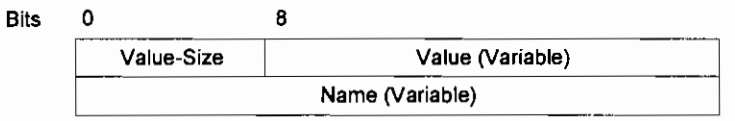


Fig. 2.4. Campo de datos de un paquete *Challenge* [1]

- ***Value-Size*:** Indica la longitud del campo *Value*
- ***Value*:** Un paquete *Challenge* denomina a este campo *Challenge Value*, y contiene una cadena aleatoria y de longitud variable de octetos. Este *Challenge Value* debe ser cambiado cada que un nuevo paquete *Challenge* es enviado. La longitud del *Challenge Value* es independiente del método utilizado para generar los octetos.

  Un paquete *Response* denomina el campo *Value* como *Response Value*, y contiene un "*one-way hash*" calculado sobre la cadena de octetos consistente de la concatenación de valores: campo Identificador, secreto compartido y *Challenge Value*. La longitud del *Response Value* depende del algoritmo utilizado (16 octetos para MD5).[1]

- ***Name*:** Contiene uno o más octetos representando la identificación del sistema que está transmitiendo el paquete. No hay ninguna limitación respecto al tamaño o formato del contenido de este campo. Debido a que CHAP puede ser utilizado para autenticar diferentes sistemas, el campo *Name* permite localizar el apropiado secreto dentro de una base de datos de secretos.

Si el valor recibido en un paquete *Response* es igual al valor esperado, el autenticador transmite un mensaje *Success*, caso contrario un mensaje *Failure*. Estos mensajes pueden contener un mensaje de texto que será mostrado al usuario indicándole el éxito o falla de la autenticación.

2.2 *REMOTE AUTHENTICATION DIAL IN USER SERVICE* (RADIUS)

En el acceso de usuarios a una red (Ejm. Internet, red corporativa, VPN-*Virtual Private Network*), mediante enlaces seriales dispersos, se requiere establecer mecanismos de seguridad, autorización y tarifación. Esto se puede lograr administrando una única base de datos de usuarios, la cual contenga información de autenticación, de configuración, y que detalle el tipo de servicio que se deberá entregar al usuario (Ejm. SLIP, PPP, Telnet, etc.).

RADIUS (*Remote Authentication Dial In User Service)* es un protocolo encargado de llevar información de autenticación, autorización, configuración y tarifación entre un Servidor de Acceso de Red (NAS - *Network Access Server*), el cual desea autenticar sus enlaces, y un servidor de autenticación (servidor RADIUS).

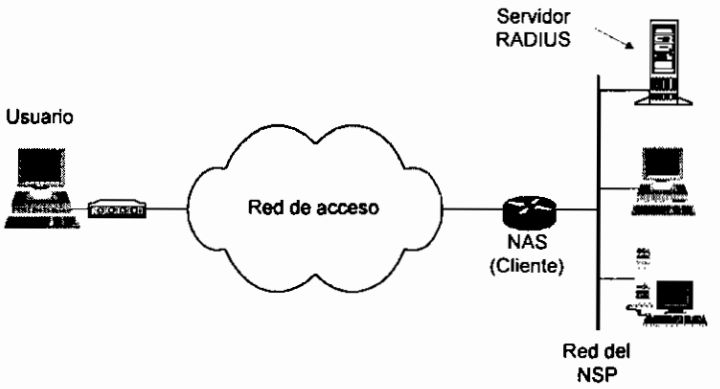


Fig. 2.5. Topología de RADIUS

RADIUS se basa en el modelo cliente-servidor. El cliente (el NAS) es responsable de pasar información de autenticación de los usuarios al o los servidores RADIUS designados, y luego actuar según la respuesta que le sea enviada. Los servidores RADIUS son responsables de recibir pedidos de conexiones de usuarios, autenticar al usuario, y luego retornar toda la información de configuración necesaria para que el

cliente pueda entregar el respectivo servicio al usuario. Un servidor RADIUS puede actuar como un cliente *proxy* ante otros servidores RADIUS u otras clases de servidores de autenticación.

A fin de proporcionar seguridad, las transacciones entre el cliente y el servidor RADIUS son autenticadas a través del uso de un secreto compartido, el cual nunca es enviado a través de la red. Además, todos los *password(s)* de usuario son enviados encriptados entre el cliente y el servidor RADIUS, para eliminar la posibilidad de que sean utilizados en caso de caer en manos no adecuadas. El servidor RADIUS soporta una variedad de métodos de autenticación, tales como: PAP, CHAP, entre otros.

El formato de un paquete RADIUS se presenta a continuación:

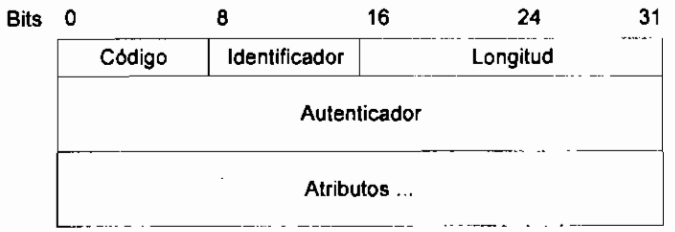


Fig. 2.6. Formato de un paquete RADIUS [3]

Código: Identifica el tipo de paquete RADIUS. Los siguientes códigos y tipos de paquetes están definidos:

Código	Tipo
1	Access-Request
2	Access-Accept
3	Access-Reject
4	Accounting-Request (Start o Stop)
5	Accounting-Response
11	Access-Challenge

Tabla. 2.3. Códigos y tipos de paquetes RADIUS [3]

- **Identificador:** Contiene un identificador que permite relacionar pedidos y respuestas.
- **Longitud:** Indica la longitud medida en octetos del paquete, incluida la cabecera.
- **Autenticador:** Son 16 octetos que permiten autenticar al NAS ante el servidor RADIUS. Según el tipo de paquete, puede ser de dos tipos:

- *Request Authenticator* (En paquetes *Access-Request*)
- *Response Authenticator* (En paquetes *Access-Accept, Access-Reject* y *Access-Challenge*)

- ***Request Authenticator*** : Contiene un valor que debe ser impredecible y el cual nunca se debe repetir a lo largo de toda la vida de un secreto (un *password* compartido entre el cliente (NAS) y el servidor RADIUS). Esto evita que un valor repetido junto con el mismo secreto, permita a un intruso responder con una respuesta interceptada previamente.

  El NAS y el servidor RADIUS comparten entre si un secreto. Sobre la concatenación del secreto compartido seguido del *Request Authenticator* se ejecuta la función "*one-way-hash*" utilizando el algoritmo MD5, para crear una respuesta (*hash*) de 16 octetos. Entre este *hash* y el *password* ingresado por el usuario se efectúa la operación lógica XOR, y el resultado es colocado en un atributo denominado *User-Password*, del campo Atributos. Es decir: [3]

  User Password = [MD5(Secreto + *RequestAuth*)] XOR [*password* de usuario]

- ***Response Authenticator*** : Contiene un *hash* (MD5) calculado sobre una cadena de octetos formada por la concatenación de los campos: Código, Identificador, Longitud, el *Request Authenticator* (del paquete *Access-Request*), y los atributos del respectivo paquete de respuesta, seguido por el secreto compartido. Es decir: [3]

  Response Auth = MD5(Código+ID+Longitud+*RequestAuth*+Atributos+Secreto)

- **Atributos:** Los atributos llevan la información específica de autenticación, autorización, detalles de configuración y tarifación en los paquetes de pedidos o respuestas. Cada atributo posee el formato mostrado a continuación:

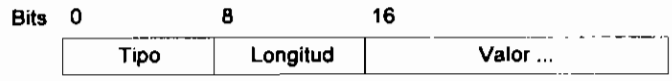


Fig. 2.7. Formato de un atributo de RADIUS [3]

- **Tipo:** Identifica el tipo de atributo. Cada tipo de paquete contendrá uno o varios atributos específicos. Los atributos más utilizados se presentan a continuación:

Tipo	Nombre	Contiene
1	*User-Name*	Nombre del usuario a ser autenticado
2	*User-Password*	*Password* del usuario a ser autenticado, o la respuesta (*hash*) de un usuario correspondiente a un paquete *Access-Challenge*.
3	*CHAP-Password*	Valor provisto por un usuario CHAP en respuesta a un desafío.
4	*NAS-IP-Address*	Dirección IP del NAS que solicitó autenticar a un usuario.
5	*NAS-Port*	Número de puerto físico del NAS que está autenticando a un usuario. No es un puerto TCP/UDP.
6	*Service-Type*	Tipo de servicio que el usuario ha solicitado, o el tipo de servicio que se le brindará. Ejm. *Framed* (PPP, SLIP), Login, solo autenticación, etc.
7	*Framed-Protocol*	Tipo de protocolo de capa enlace a ser usado para un servicio *Framed*. Ejm. PPP, SLIP, etc.
8	*Framed-IP-Address*	Dirección IP a ser configurada por el NAS para el usuario.
9	*Framed-IP-Netmask*	Máscara de subred IP a ser configurada para el usuario, cuando el usuario es un ruteador de una red.
10	*Framed-Routing*	Método de ruteo para el usuario, cuando éste es un ruteador en una red.
11	*Filter-Id*	Nombre de la lista de filtros para el usuario.
12	*Framed-MTU*	MTU a ser configurada para el usuario, cuando no ha sido negociado mediante PPP.
13	*Framed-Compression*	Protocolo de compresión a ser utilizado para el enlace. Ejm. Compresión de cabeceras TCP/IP de Van Jacobson.
14	*Login-IP-Host*	Dirección IP del sistema al cual debe ser conectado el usuario cuando el atributo *Login -Service* está presente.
15	*Login-Service*	Servicio al cual debe conectarse el usuario en el *Login Host*. Ejm. Telnet, Rlogin, etc.
16	*Login-TCP-Port*	Puerto TCP al cual el usuario deberá ser conectado, cuando *el Login-Service* está presente.
18	*Reply- Message*	Contiene texto que puede ser mostrado al usuario.
22	*Framed-Route*	Información de ruteo a ser configurada para el usuario en el NAS.
24	*State*	Valor opcional enviado por el servidor al cliente en un paquete *Access-Challenge*, el cual debe ser enviado sin modificación en un nuevo paquete *Access-Request* desde el cliente al servidor.
25	*Class*	Valor opcional enviado por el servidor al cliente, el cual debería ser enviado sin modificación desde el cliente al servidor de tarifación en paquetes de tarifación.
26	*Vendor-Specific*	Permite que vendedores específicos añadan sus propios atributos.
30	*Called-Station-Id*	Permite al NAS enviar en un *Access-Request* el número de teléfono al cual el usuario llamó, utilizando la *tecnología Dialed Number Identification Specification* (DNIS) o alguna similar.
31	*Calling-Station-Id*	Permite al NAS enviar en un paquete *Access-Request* el número telefónico desde donde vino la llamada del usuario, utilizando la tecnología *Automatic Number Identification* (ANI) u otra similar.
32	*NAS-Identifier*	Cadena de caracteres que identifica el NAS que generó un paquete *Access-Request*. Ejm. Nombre de dominio del NAS
40-59	*Tarifación*	Atributos que contienen información de tarifación.
60	*CHAP-Challenge*	Desafío CHAP enviado por el NAS al usuario. Si el desafío es de 16 octetos, se lo envía en el campo *Request Authenticator* en vez de usar este atributo.
61	*NAS-Port-Type*	Tipo de puerto físico del NAS que está autenticando al usuario. Ejm. Asimétrico, Simétrico, ISDN, etc.

Tabla. 2.4. Principales atributos empleados en paquetes RADIUS durante la autenticación y configuración de un usuario [3]

Tipo	Nombre	Contiene
40	*Acct-Status-Type*	Indica si el paquete *Accounting-Request* marca el inicio (*Start*) o fin (*Stop*) del servicio del usuario.
42	*Acct-Input-Octets*	Indica cuantos octetos han sido recibidos desde el puerto sobre el cual el servicio está siendo provisto.
43	*Acct-Output-Octets*	Indica cuantos octetos han sido enviados al puerto en el curso de la entrega del servicio.
44	*Acct-Session-Id*	Es un identificador único que permite identificar registros de inicio y fin de tarifación en la base de datos del servidor.
45	*Acct-Authentic*	Indica como el usuario fue autenticado. Ejm. Servidor RADIUS, por el mismo NAS, u otro protocolo de autenticación remoto. Aquellos usuarios que acceden a un servicio sin ser autenticados no generan registros de tarifación.
46	*Acct-Session-Time*	Indica cuantos segundos el usuario ha recibido servicio.
47	*Acct-Input-Packets*	Indica cuantos paquetes han sido recibidos en el puerto sobre el cual se está brindando servicio a un usuario *Framed.*
48	*Acct-Output-Packets*	Indica cuantos paquetes han sido enviados al puerto sobre el cual se está brindando servicio a un usuario *Framed.*
49	*Acct-Terminate-Cause*	Indica el motivo por el cual una sesión con el usuario fue terminada. Ejm. pedido del usuario, perdida de portadora, error de puerto, error de NAS, error de usuario, etc.

Tabla. 2.5. Principales atributos empleados en paquetes RADIUS para la tarifación de un usuario[4]

- **Longitud:** Indica la longitud del atributo, incluyendo los campos de la cabecera del atributo.
- **Valor:** Contiene cero o más octetos correspondientes a la información específica de ese atributo.

2.2.1 Operación de RADIUS (Autenticación y Configuración)

Cuando un cliente es configurado para usar RADIUS, cualquier usuario del cliente presenta información de autenticación al cliente. Esto podría ser mediante un "*login prompt*" personalizado, donde el usuario ingresa su nombre de usuario y *password.* Luego, el usuario podría utilizar un protocolo tal como PPP, el cual durante la fase de autenticación permite enviar esta información al cliente.

Una vez que el cliente ha obtenido tal información, puede escoger autenticarla utilizando RADIUS. Para hacerlo, el cliente envía un paquete "*Access-Request*" conteniendo atributos tales como: el nombre de usuario, el *password* de usuario, el ID del cliente (*NAS-IP-address* o *NAS-Identifier*) y el ID del puerto en el NAS al cual el

usuario está accesando. Cuando un *password* está presente es ocultado, utilizando el algoritmo MD5.

El paquete *Access-Request* es entregado al servidor RADIUS a través de la red. Si el cliente no recibe ninguna respuesta dentro de un intervalo de tiempo, el pedido es reenviado por un cierto número de veces. El cliente puede también enviar pedidos hacia uno o varios servidores alternos en el caso de que el servidor primario haya fallado o se lo considere inalcanzable.

Una vez que el servidor RADIUS recibe el pedido, valida el cliente que lo envió. Un pedido desde un cliente para el cual el servidor RADIUS no tiene un secreto compartido es desechado. Si el cliente es válido, el servidor RADIUS consulta una base de datos de usuarios para encontrar el usuario cuyo nombre corresponde al pedido. El registro correspondiente al usuario en la base de datos contiene una lista de requerimientos los cuales deben ser cumplidos para permitir al acceso para ese usuario. Esto siempre incluye verificación del *password*, pero puede también especificar el cliente(s) o puerto(s) a los cuales el usuario está permitido acceder.

El servidor RADIUS puede hacer pedidos a otros servidores para satisfacer el pedido, en el caso de que actúe como un cliente *proxy*.

Si cualquier condición no es satisfecha, el servidor RADIUS envía un paquete de respuesta "*Access-Reject*" indicando que el pedido de este usuario es inválido. Si desea, el servidor puede incluir un mensaje de texto en el paquete *Access-Reject* el cual será mostrado por el cliente al usuario. Ningún otro atributo es permitido en un mensaje *Access-Reject*.

Si todas las condiciones son satisfechas y si el servidor RADIUS desea realizar un desafío adicional, al cual el usuario deba responder, el servidor RADIUS enviará un paquete de respuesta "*Access-Challenge*", el cual puede incluir un mensaje de texto que será mostrado por el cliente al usuario solicitando una respuesta al desafío. Recibida la respuesta del usuario, el cliente reenvía su mensaje original *Access-Request* con un nuevo ID de pedido, con el atributo *User-Password* reemplazado por la respuesta

(encriptada) del usuario. El servidor puede responder a este nuevo *Access-Request* ya sea con un *Access-Accept*, un *Access-Reject*, u otro *Access-Challenge*.

Si todas las condiciones son cumplidas, la lista de valores de configuración para el usuario son colocadas dentro de una respuesta "*Access-Accept*". Estos valores incluyen el tipo de servicio (por ejemplo: SLIP, PPP, Telnet, Rlogin) y todos los valores necesarios para entregar el servicio deseado. Para SLIP y PPP, éste puede incluir valores tales como dirección IP, máscara de subred, MTU, compresión deseada, identificadores de filtros de paquetes deseados, etc. Para usuarios en modo carácter, este puede incluir valores tales como protocolo deseado y *host* al cual se debe acceder.

Muchos NASs tienen la capacidad de asignar directamente la información de configuración a los usuarios, sin necesidad de la participación del servidor RADIUS. El ejemplo típico de un NAS de este tipo, como se verá en el capítulo III, es un servidor de acceso remoto (RAS - *Remote Access Server*), empleado en un acceso *dial up* a un ISP. El RAS termina la sesión PPP del usuario, asigna direcciones IP mediante IPCP a cada usuario a partir de un conjunto de direcciones disponibles configuradas en el RAS, etc. El RAS se comunica con el servidor RADIUS básicamente para realizar la autenticación y tarifación de los usuarios.

Un NAS que acepta información de configuración para un usuario, desde un servidor RADIUS, generalmente es aquel a través del cual se tiene acceso a varios proveedores de servicios de red (NSPs) diferentes.

2.2.2 Interoperación con PAP y CHAP

Para PAP, el NAS toma el *Peer-ID* y *password* contenidos en el paquete PAP y los envía en un paquete *Access-Request* como los atributos *User-Name* y *User-Password* respectivamente. El NAS puede incluir los atributos *Service-Type* = *Framed-User* y *Framed-Protocol* = PPP como un indicación para el servidor que se espera servicio PPP. [3]

Para CHAP, el NAS genera un desafío aleatorio (preferiblemente de 16 octetos) y lo envía al usuario, el cual envía de regreso una respuesta CHAP. Luego, el NAS toma del paquete CHAP la información de los campos Identificador (o CHAP-ID), *Name* y *Response Value*, y los envía en un paquete *Access-Request* al servidor RADIUS, de la siguiente forma: el campo *Name* de CHAP es colocado en el atributo *User-Name*, los campos CHAP-ID y *Response-Value* son colocados en el atributo *CHAP-Password.*[3]

El desafío aleatorio hecho por el NAS al usuario también es enviado al servidor RADIUS, ya sea incluido en el atributo *CHAP-Challenge,* ó si es de 16 octetos de longitud, colocado en el campo *Request-Authenticator* del paquete *Access-Request*. El NAS puede incluir los atributos *Service-Type* = *Framed- User* y *Framed-Protocol* = PPP como una indicación al servidor RADIUS que se espera el servicio PPP.[3]

El servidor RADIUS busca en su base de datos el *password* asociado al *User-Name*. A continuación, efectúa el algoritmo MD5 sobre la concatenación: CHAP-ID, el *password* encontrado, y el *CHAP-Challenge* (desde el atributo *CHAP-Challenge* si está presente, de otra forma desde el *Request-Authenticator*). El resultado lo compara con el valor presente en el atributo *CHAP-Password.* Si ellos coinciden, el servidor envía de regreso un *Access-Accept*, de otra forma envía un *Access-Reject.*

Si el servidor RADIUS no es capaz de ejecutar la autenticación requerida, ya que no posee el *password* correspondiente al *User-Name*, debería retornar un *Access-Reject* al cliente.

2.2.3 Operación de RADIUS (Tarifación)

La tarifación mediante RADIUS se basa en el intercambio de 3 tipos de paquetes: *Accounting-Request Start, Accounting-Request Stop* y *Accounting-Response*. El tipo de paquete *Accounting-Request* está especificado según el atributo *Acct-Status-Type*.[4]

Cuando un cliente es configurado para utilizar tarifación RADIUS, al inicio de la entrega de servicio generará un paquete *Accounting-Request-Start* describiendo el tipo de servicio y el usuario al cual está siendo entregado, y enviará ese paquete al servidor

de tarifación RADIUS, el cual enviará de regreso un acuse de recibo al paquete que ha sido recibido. Al final de la entrega de servicio, el cliente generará un paquete *Accounting-Request-Stop* describiendo el tipo de servicio que fue entregado y estadísticas adicionales tales como el tiempo transcurrido, octetos enviados y recibidos, o paquetes enviados y recibidos. Esta información la enviará al servidor de tarifación RADIUS, el cual enviará de regreso un acuse de recibo del paquete que ha sido recibido. El servidor almacenará la información de tarifación del usuario, enviada por el cliente, en un registro para posterior procesamiento.

El paquete *Accounting-Request* (*Start* o *Stop*) es entregado al servidor de tarifación RADIUS a través de la red. El cliente reenvía el paquete *Accounting-Request* hasta que reciba un acuse de recibo *Accounting-Response* desde el servidor. Si no hay respuesta dentro de un intervalo de tiempo, el pedido es reenviado a otro servidor o servidores en el caso de que el servidor primario haya fallado o se lo considere inalcanzable.

El servidor de tarifación RADIUS puede hacer pedidos a otros servidores para satisfacer el pedido, en el caso en que actúe como cliente *proxy*.

Si el servidor de tarifación RADIUS es incapaz de almacenar la información de tarifación, no envía un paquete *Accounting-Response* como acuse de recibo al cliente.

Para brindar un mecanismo de seguridad, el campo Autenticador es utilizado para autenticar mensajes de tarifación entre el cliente y el servidor RADIUS de tarifación. En paquetes *Accounting-Request*, el campo Autenticador se *denomina Request-Authenticator*, y contiene un "*one-way-hash-MD5*" de 16 octetos calculado sobre la cadena de octetos formada por los campos del paquete *Accounting-Request*: Código, Identificador, Longitud, 16 octetos cero, atributos de pedido, secreto compartido.

En paquetes *Accounting-Response*, el campo Autenticador se denomina *Response-Authenticator*, y contiene un "*one-way-hash-MD5*" de 16 octetos calculado sobre la cadena de octetos formada por los campos del paquete *Accounting-Response*: Código, Identificador, Longitud, el *Request-Authenticator* del paquete *Accounting-Request*, atributos de respuesta y secreto compartido. [4]

2.2.4 RADIUS sobre UDP

RADIUS utiliza UDP como protocolo de transporte. Exactamente un paquete es encapsulado en un campo de datos UDP donde el puerto de destino UDP indica 1812 (decimal). Cuando una réplica es generada, los puertos fuente y destino son invertidos.

2.3 *LAYER TWO TUNNELING PROTOCOL* (L2TP)

El protocolo PPP es frecuentemente utilizado en el enlace punto a punto entre una computadora personal, con módem *dial up*, y un proveedor de servicios de red (NSP) (Ejm. ISP, red corporativa, VPN) para configurar el enlace, autenticar usuarios, permitir la multiplexación de varios tipos de protocolos, configurar los parámetros de capa red, etc.

El protocolo L2TP (*Layer Two Tunneling Protocol*) permite a un servidor *dial up* proveer acceso a un NSP remoto, extendiendo la sesión PPP de un cliente (usuario), a través de una red de conmutación de paquetes a la cual ambos, el servidor *dial up* y el NSP, están directamente conectados (Ejm. Internet, red Frame Relay o red ATM).

Al servidor *dial up* se lo denomina un Concentrador de Acceso L2TP o LAC (*L2TP Access Concentrator*). El dispositivo que provee acceso a la red del NSP se lo denomina un Servidor de Red L2TP o LNS (*L2TP Network Server*).

L2TP utiliza el mecanismo de encapsulamiento (*tunneling*) para extender la sesión PPP del usuario desde el LAC hasta el LNS. Un túnel L2TP actúa como una conexión punto a punto a nivel de capa enlace entre el LAC y el LNS, creada a través de la red de conmutación de paquetes común a ambos.

L2TP se encarga de que el encapsulamiento (*tunneling)* de las tramas PPP a través de la red de conmutación de paquetes resulte transparente tanto para el usuario y sus aplicaciones.

A continuación se presenta un esquema de la arquitectura en que se ejecuta L2TP.

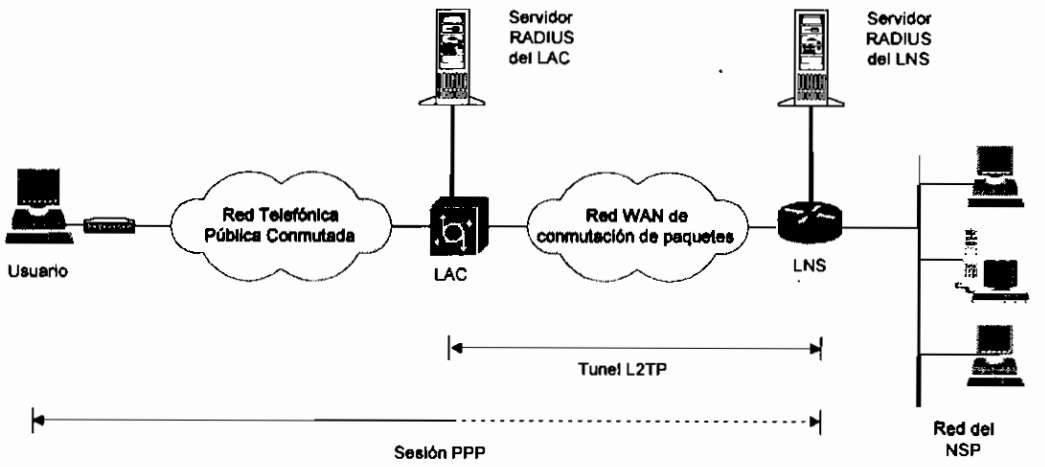


Fig. 2.8. Arquitectura L2TP [5]

En la arquitectura se definen:

- **Sistema Remoto:** Es un sistema final o ruteador conectado a una red de acceso remota (Ejm. Red Telefónica Pública Conmutada), el cual inicia una llamada. También se lo denomina: cliente *dial up* o cliente virtual *dial up*.
- ***L2TP Access Concentrator* (LAC):** Es un nodo que actúa por un lado como un punto extremo de un túnel L2TP y es *peer* del LNS. El LAC está ubicado entre un LNS y un sistema remoto; envía paquetes a y desde cada uno de ellos. Los paquetes enviados desde el LAC al LNS requieren ser encapsulados mediante el protocolo L2TP, creando así un túnel a través de la red de conmutación de paquetes. La conexión de un sistema remoto al LAC generalmente es mediante un enlace PPP. Pueden también existir sistemas localmente conectados al LAC
- ***L2TP Network Server* (LNS):** Es un nodo que actúa por un lado como un punto extremo de un túnel L2TP y es *peer* del LAC. El LNS es el punto de terminación lógica de la sesión PPP, que está siendo enviada a través de un túnel desde el sistema remoto por el LAC.

- ***Peer*:** Cuando se toma en el contexto de L2TP, *peer* se refiere ya sea al LAC o LNS. El *peer* de un LAC es un LNS y viceversa. Cuando se usa en el contexto de PPP, un *peer* es cualquier extremo de la conexión PPP.
- **Servidor Radius:** Servidor encargado de realizar funciones de autenticación, autorización, configuración y tarifación.

En la terminología de L2TP se definen tres conceptos:

- **Llamada (*Call*) :** Se denomina una llamada a una conexión (o intento de conexión) entre un sistema remoto y un LAC. Por ejemplo, una llamada telefónica a través de la red telefónica conmutada. Una llamada la cual es exitosamente establecida entre un sistema remoto y un LAC resulta en una correspondiente sesión L2TP dentro del túnel entre el LAC y el LNS, previamente establecido.

- **Sesión:** L2TP es orientado a conexión. Una sesión L2TP es creada entre el LAC y LNS cuando una conexión PPP extremo a extremo se desea establecer entre el sistema remoto y el LNS. Los datagramas relacionados a la conexión PPP son enviados dentro del túnel entre el LAC y el LNS. Hay una relación entre sesiones L2TP establecidas y sus llamadas asociadas.

- **Túnel:** Un túnel existe entre un par LAC-LNS. El túnel consiste de una conexión de control y cero o más sesiones L2TP. El túnel lleva datagramas PPP encapsulados y mensajes de control entre el LAC y el LNS.

L2TP utiliza dos tipos de mensajes: mensajes de control y mensajes de datos. Los mensajes de control son utilizados en el establecimiento, mantenimiento y finalización de túneles y sesiones. Los mensajes de datos son utilizados para encapsular tramas PPP siendo llevadas sobre el túnel.

La figura 2.9 muestra la estructura del protocolo L2TP.

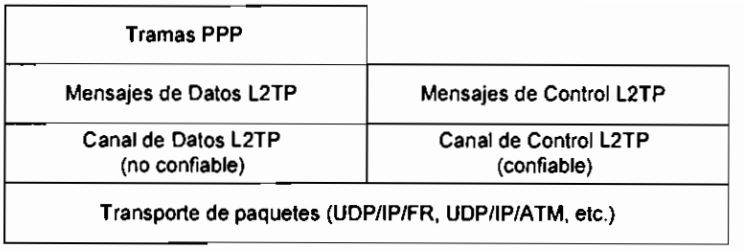


Fig. 2.9. Estructura del Protocolo L2TP [6]

Las tramas PPP son pasadas sobre un canal de datos no confiable encapsulado primero por una cabecera L2TP y luego por las del protocolo de transporte UDP. Los mensajes de control son enviados sobre un canal de control L2TP confiable el cual transmite paquetes en-banda sobre el mismo protocolo de transporte.

Se utilizan números de secuencia en todos los mensajes de control, a fin de proveer entrega confiable en el canal de control. Los mensajes de datos pueden usar números de secuencia para reordenar paquetes y detectar paquetes perdidos.

Los paquetes L2TP para el canal de control y canal de datos comparten un formato de cabecera común. La cabecera contiene algunos campos opcionales, los cuales no existirán en el mensaje si el campo es marcado no presente.

El formato de un paquete L2PT se presenta a continuación:

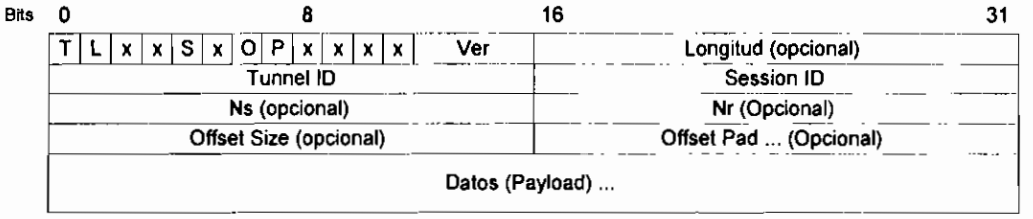


Fig. 2.10. Formato de un paquete L2TP [6]

- **Bit T (Tipo):** Indica el tipo de mensaje. Si T = 0 es un mensaje de datos y si T = 1 es un mensaje de control.
- **Bit L (Longitud):** Si L = 1, el campo Longitud está presente. L = 1 en todos los mensajes de control.

- **Bits x :** Son bits reservados para extensiones futuras. Deben ser puestos a 0.
- **Bit S (Secuencia):** Si S = 1, indica que los campos Ns y Nr están presentes. S = 1 en todos los mensajes de control.
- **Bit O (*Offset*):** Si O = 1, el campo *Offset Size* está presente. O = 0 en todos los mensajes de control.
- **Bit P (Prioridad):** Si P = 1 indica que el mensaje de datos debería recibir tratamiento especial en su transmisión.
- **Ver:** Indica la versión de la cabecera del mensaje de datos L2TP. (Versión actual = 2).
- **Longitud:** Este campo indica la longitud total del mensaje en octetos.
- ***Tunnel ID*:** Contiene el identificador de la conexión de control. Los túneles L2TP son nombrados por identificadores que tienen únicamente significado local. Es decir, un mismo túnel tendrá diferentes identificadores en cada extremo del túnel. El *Tunnel ID* contenido en un mensaje es el asignado por el extremo destino y no el asignado por el extremo que envía el mensaje L2TP. Los *Tunnel IDs* son seleccionados e intercambiados durante la creación del túnel.
- ***Session ID*:** Contiene el identificador para una sesión dentro de un túnel. Las sesiones L2TP son nombradas por identificadores que tienen únicamente significado local. Es decir, una misma sesión tendrá diferentes identificadores en cada extremo. El *Session ID* contenido en la cabecera de un mensaje es el asignado por el extremo destino y no el asignado por el extremo que envía el mensaje L2TP. Los *Session IDs* son seleccionados e intercambiados durante la creación del túnel.
- **Ns:** Indica el número de secuencia para el mensaje de control o datos enviado. Ns empieza en cero y se incrementa en uno para cada mensaje enviado.
- **Nr:** Indica el número de secuencia esperado en el siguiente mensaje de control a ser recibido. Nr contiene el valor Ns del último mensaje recibido más uno. En mensajes de datos, Nr es reservado y, si está presente (como indicado en el bit S), es ignorado después de recibirse.
- ***Offset Size*:** Si está presente, especifica el número de octetos pasados la cabecera L2TP en el cual el campo de datos (*payload*) inicia.
- ***Offset Padding*:** Relleno opcional antes del *payload*.

Los campos de Longitud, Ns y Nr, son opcionales en los mensajes de datos, pero deben estar presentes en todos los mensajes de control.

2.3.1 Mensajes de Control y AVPs usados en L2TP

Los mensajes de control son utilizados en el establecimiento, mantenimiento y finalización de túneles y sesiones. El campo *Payload* de un mensaje de control está formado por uno o varios pares atributo-valor (AVPs - *Attribute-Value Pairs*). Los AVPs, solo están presentes en mensajes de control; los mensajes de datos contienen tramas PPP.

Un AVP tiene el siguiente formato:

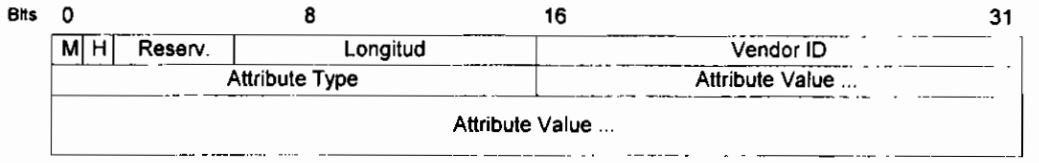


Fig. 2.11. Formato de un AVP de L2TP [6]

Los primeros 6 bits son una máscara de bit, que describe los atributos generales del AVP.

- **Bit M (*Mandatory*):** Controla el comportamiento requerido de una implementación, la cual recibe un AVP el cual no reconoce. Si el bit M = 1 está presente en un AVP no reconocido dentro del mensaje asociado con una sesión particular, la sesión asociada con este mensaje debe ser terminada. Si el bit M = 1 está en un AVP no reconocido dentro de un mensaje asociado con el túnel entero, el túnel entero (y sesiones dentro de él) debe ser terminado. Si el bit M = 0, un AVP no reconocido debe ser ignorado. El resto del mensaje de control debe continuar siendo procesado como si el AVP no hubiese estado presente.
- **Bit H (*Hidden*):** Identifica el ocultamiento de información en el campo *Attribute Value* de un AVP. Esta capacidad, puede ser usada para evitar el paso de información sensitiva, tales como *passwords* en un AVP.
- **Reservados:** Reservados para extensiones futuras. Son puestos a cero.

- **Longitud:** Codifica el número de octetos (incluyendo el campo Longitud y campos de la máscara de bits) contenidas en ese AVP.
- ***Vendor ID*:** Es un número asignado por la IANA, el cual identifica un vendedor específico que implemente sus propias extensiones de L2TP.
- ***Attribute Type*:** Contiene un valor con una interpretación única a través de todos los AVPs bajo un *Vendor ID* dado. Cada AVP está identificado por un tipo y cada uno tiene cierta información específica.
- ***Attribute Value*:** Esto es el valor/información indicado por el *Vendor ID* y *Attribute Type*.

Los AVPs en el mensaje de control permiten: identificar el tipo de mensaje de control, mostrar códigos de error, contienen información específica relacionada a la administración de la conexión de control, administración de llamadas, autenticación entre extremos, estatus de la llamada, etc. Una descripción detallada de cada AVP se encuentra en el RFC 2661.

2.3.2 Operación de L2TP

El establecimiento necesario para realizar el encapsulamiento (*tunneling*) de una sesión PPP con L2TP consiste de dos pasos:

1. Establecimiento de la conexión de control para un túnel, y
2. Establecimiento de una sesión cuando existe un pedido de llamada.

El túnel y la correspondiente conexión de control deben ser establecidas antes de que una llamada sea iniciada. Una sesión L2TP debe ser establecida antes de empezar a enviar tramas PPP a través del túnel. Múltiples sesiones pueden coexistir a través de un único túnel y múltiples túneles pueden existir entre el mismo LAC y LNS como se muestra a continuación:

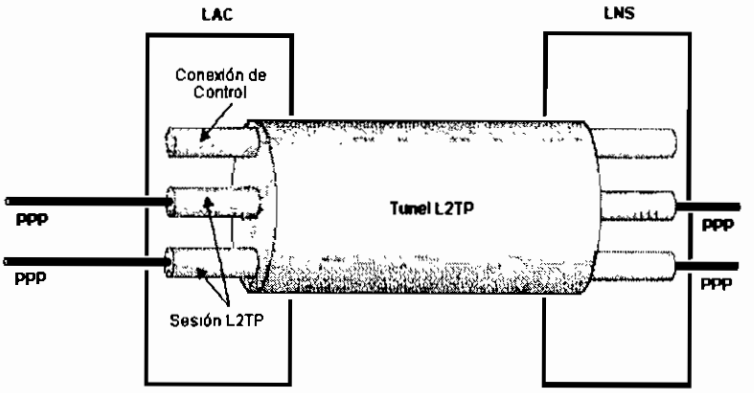


Fig. 2.12. *Tunneling* de PPP mediante L2TP

En forma resumida el funcionamiento de L2TP se indica a continuación:

1. Inicio de sesión PPP

Cuando un usuario remoto inicia una sesión PPP con el LAC (Ejm. llamándolo usando el sistema telefónico analógico, ISDN, etc.), el LAC acepta la conexión y el enlace PPP se establece.

2. Negociación LCP entre el LAC y el sistema remoto

Entre el usuario y el LAC se negocian las opciones LCP que se desean entre el usuario y el LNS. El LAC parcialmente autentica al usuario del sistema remoto, utilizando su nombre de usuario, *password*, nombre de dominio u otra información. En base a esta autenticación preliminar se identifica entre otras cosas, la dirección de red (Ejm. dirección IP) del LNS al cual el LAC debe contactarse.

3. Establecimiento de la Conexión de Control

La conexión de control es la primera conexión que debe ser establecida entre el LAC y el LNS antes de que las sesiones puedan ser iniciadas. Durante el establecimiento de la conexión de control, opcionalmente, el LAC y el LNS se autentican entre sí mediante el intercambio de mensajes de control que llevan AVPs que contienen información de

autenticación (generalmente mediante CHAP), identificación de la versión de protocolo L2TP del *peer*, *Tunnel ID* asignado por cada extremo, etc. El túnel se crea cuando la conexión de control se ha establecido. Si la autenticación falla, el establecimiento del túnel no es permitido.

4. Establecimiento de sesiones

Después de un establecimiento exitoso de la conexión de control, se pueden crear sesiones individuales dentro del túnel. Cada sesión corresponde a un único flujo PPP entre el LAC y el LNS.

El establecimiento de la sesión se lo realiza mediante el intercambio de mensajes de control entre el LAC y el LNS. El LAC propagará las opciones LCP negociadas y la información parcialmente autenticada al LNS. El LNS enviará las opciones negociadas e información de autenticación directamente a la interfaz de acceso virtual. Si las opciones configuradas en la interfaz virtual asignada no corresponden a las opciones negociadas con el LAC, la conexión fallará, y un mensaje de control de desconexión será enviado al LAC.

Durante el establecimiento de la sesión, ciertos AVPs en los mensajes de control intercambiados informan al respectivo *peer*: el *Session ID* escogido por cada extremo, información del sistema remoto (números de quien llama, número llamado, subdirecciones), si se usará o no secuenciamiento en los mensajes de datos, etc.

5. Negociación NCP entre LNS y sistema remoto

Mediante negociación PPP NCP, el sistema remoto es provisto de direcciones de red (Ejm. dirección IP) desde la red del NSP. La autenticación, autorización y tarifación pueden ser provistos por el dominio de administración del NSP como si el usuario estuviese conectado al Servidor de Acceso de red (NAS) directamente.

El resultado final es como si el proceso de intercambio pareciese ser exclusivamente entre el usuario del sistema remoto y el LNS, como si no hubiese dispositivo intermedio (LAC) involucrado.

6. Envío de tramas PPP

Una vez establecida una sesión, las tramas PPP enviadas desde el sistema remoto son recibidas en el LAC. El LAC retira los delimitadores de trama y el campo *checksum* de la trama PPP original y a los restantes campos los encapsula dentro de una trama L2TP, la cual es enviada sobre el túnel apropiado. El LNS recibe el paquete L2TP, y procesa la trama PPP encapsulada como si fuese recibida en una interfaz PPP local. El proceso contrario ocurre cuando se envían tramas PPP encapsuladas en L2TP desde el LNS al LAC.

El extremo que envía un mensaje, asociado con una sesión y túnel particular, coloca los identificadores: *Session ID* y *Tunnel ID*, especificados por el *peer* durante el establecimiento de la sesión y túnel respectivamente, en los campos correspondientes de la cabecera. De esta manera, tramas PPP correspondientes a varias sesiones son multiplexadas y demultiplexadas sobre un único túnel entre un par LNS-LAC dado. Múltiples túneles pueden existir entre un par LNS-LAC dado y múltiples sesiones pueden existir dentro del túnel.

7. *Keepalive* (*Hello*)

Para evitar que un túnel o sesión sea terminado durante períodos sin actividad, se envían períodicamente mensajes de control, denominados *Hello*, después de que el último mensaje de datos o de control fue enviado. En caso de que no se reciban mensajes *Hello* durante cierto tiempo, se declara una falla de conectividad entre el LAC y el LNS y se termina el túnel.

8. Terminación de la sesión

La terminación de la sesión puede ser iniciada sea por el LAC o LNS y es acompañada por el envío de un mensaje de control de terminación de sesión al cual el otro extremo debe acusar. Después de que la última sesión se ha terminado, la conexión de control puede terminarse también (como típicamente sucede).

9. Terminación de la sesión de control

La terminación de la sesión de control puede ser iniciada por el LAC o el LNS y es lograda enviando un único mensaje de control de terminación de túnel, al cual el otro extremo debe enviar un acuse de recibo. La terminación de un túnel termina con todas las sesiones individuales dentro de él.
El detalle de los distintos mensajes de control y los respectivos AVPs utilizados por cada uno de ellos no viene al caso, pero su descripción completa se encuentra especificada en el RFC 2661.

2.3.3 L2TP sobre UDP/IP

L2TP utiliza el puerto UDP bien conocido 1701. El mensaje entero L2TP, incluyendo *payload* y cabecera, es enviado dentro de un datagrama de usuario UDP. El iniciador de un túnel (el LAC) escoge un puerto disponible UDP y envía su mensaje al puerto bien conocido 1701 en el destino (LNS).

Puede ocurrir fragmentación IP cuando el paquete L2TP viaja a través del sustrato IP. L2TP no hace esfuerzo opcional para optimizar esto.

Cuando se envía L2TP sobre UDP/IP, se debe habilitar el uso de la suma de verificación en el datagrama de usuario UDP, la cual por lo general es opcional. El *checksum* permite detectar a nivel de transporte si los datos enviados llegaron intactos al destino, especialmente los mensajes de control.

UDP/IP proporciona un transporte no confiable a L2TP. Los mensajes L2TP están sujetos a pérdidas, daños, reordenamiento. L2TP por si mismo ofrece un canal de control confiable para los mensajes de control, mediante el uso de números de secuencia, los cuales permiten determinar paquetes perdidos o desordenados. El uso de números de secuencia en los mensajes de datos no es utilizado para solicitar retransmisión a nivel de L2TP, únicamente para informar si han existido reordenamiento de los paquetes. Las capas superiores se encargarán de tomar las medidas respectivas.

2.4 *MULTILINK PROTOCOL PPP* (MP-PPP)

Como se revisó en 1.3.5, PPP permite la transmisión de datagramas, correspondientes a una comunicación, sobre un único enlace serial. *Multilink Protocol PPP* (MP-PPP) es un protocolo que permite utilizar múltiples enlaces PPP independientes entre un par de sistemas, para una misma comunicación. MP-PPP provee un enlace virtual con mayor ancho de banda que cualesquiera de sus enlaces constitutivos.

MP-PPP como se verá en el capítulo IV, es principalmente utilizado en una red de acceso ISDN, donde se tienen múltiples canales portadores tipo B, pero es igualmente aplicable en cualquier situación en la cual múltiples enlaces PPP conectan dos sistemas.

El uso de MP-PPP es acordado durante la fase de negociación LCP inicial de PPP, mediante el envío de ciertas opciones de *multilink* en cada enlace. Esta negociación indica tres cosas sobre el sistema que envía la opción:

1. El sistema es capaz de combinar múltiples enlaces en un solo enlace lógico.
2. El sistema es capaz de recibir fragmentos de paquetes de capas superiores y usando una cabecera *multilink* (descrita posteriormente) puede reensamblar los fragmentos en el paquete original para posterior procesamiento.
3. El sistema es capaz de recibir paquetes de tamaño N octetos, donde N es especificado como parte de la opción *multilink*, aun si N es mayor que la máxima unidad de transferencia para un simple enlace físico.

Una vez que el *multilink* ha sido exitosamente negociado, el sistema de origen es libre de enviar paquetes encapsulados y/o fragmentados con la cabecera *multilink* añadida.

En el caso general, cada enlace puede ser configurado individualmente con diferentes opciones LCP y protocolo de autenticación. Los protocolos de red e intercambios de control asociados son normalmente negociados una sola vez, sobre el enlace lógico *multilink.*

Los paquetes de negociación LCP (Ejm. *Configure-Request, -Reject, -Ack, -Nak, Terminate-Request* o *-Ack*) no son permitidos en el enlace *multilink.* Otros paquetes LCP (Ejm. *Code-Reject, Protocol-Reject, Echo-Request, Echo-Reply* y *Discard-Request*) si son permitidos en el enlace *multilink.* Estos últimos solo hacen funciones de control y no cambian los parámetros del enlace como los primeros. [7]

Una vez configurado el enlace lógico *multilink,* es identificado mediante un par de identificadores de los dos sistemas conectados. Un identificador de sistema puede incluir información provista por la autenticación PPP e información provista por la negociación LCP.

En un enlace *multilink,* el protocolo de capa red (Ejm. IP) es primeramente encapsulado de acuerdo al procedimiento normal de PPP, utilizando opciones de compresión de los campos Dirección, Control, y Protocolo, configuradas mediante LCP. Los paquetes grandes son divididos en múltiples segmentos de tamaño apropiado para los múltiples enlaces físicos. Los paquetes no necesariamente deben ser divididos en fragmentos del mismo tamaño, o incluso no requieren ser divididos. Una posible estrategia para soportar enlaces miembros con diferentes velocidades de transmisión es dividir los paquetes en segmentos proporcionales a las velocidades de transmisión. Otra estrategia es dividirlos en muchos fragmentos iguales y distribuir múltiples fragmentos por enlace, el número debe ser proporcional a las velocidades relativas de los enlaces.

A cada fragmento (o paquete sin división), se le añade una nueva cabecera PPP, la cual contiene como identificador de Protocolo el valor 0x00-0x3d, que identifica un paquete MP-PPP, contenido en el campo de datos. [7]

Los posibles tipos de paquetes MP-PPP se presentan a continuación:

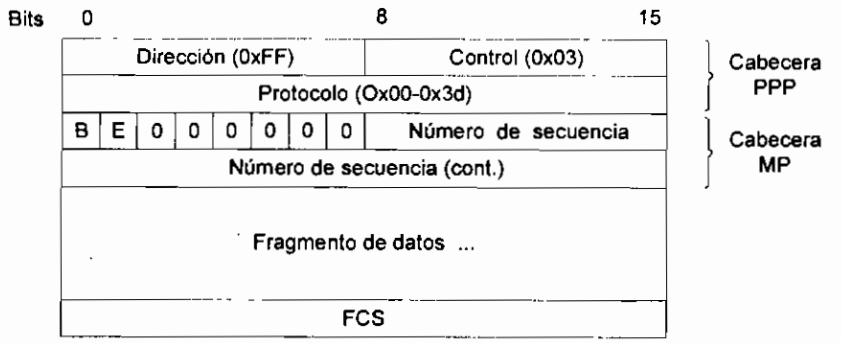


Fig. 2.13 Formato de fragmento MP-PPP con número de secuencia largo [7]

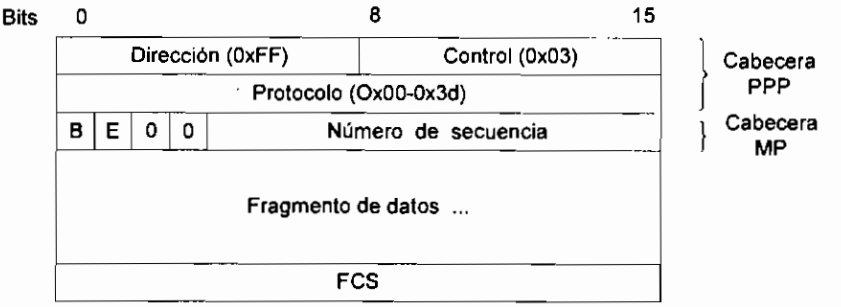


Fig. 2.14 Formato de fragmento MP-PPP con número de secuencia corto [7]

- **Bit B (*Begin*):** Un valor B = 1 indica que un fragmento es el primero del paquete PPP original. Un valor B = 0 indica que es cualquier otro fragmento del paquete PPP.
- **Bit E (*End*):** Un valor E = 1 indica que un fragmento es el último del paquete PPP original; un valor E = 0 indica que es cualquier otro fragmento intermedio del mismo paquete. Un paquete PPP que no ha necesitado ser fragmentado contiene ambos bits B y E puestos en 1.
- **Número de secuencia:** Permite reconstruir el paquete original a partir de los fragmentos. El número de secuencia puede ser 24 o 12 bits, el cual es incrementado para cada fragmento transmitido. Por defecto es de 24 bits. Puede ser negociado a 12 bits mediante el proceso de negociación de opciones con LCP.

El receptor utiliza los números de secuencia para detectar fragmentos perdidos. En caso de pérdida o daño de algún fragmento, todos los fragmentos correspondientes

al paquete PPP transmitido son desechados, y se reinicia el ensamblaje del siguiente paquete llevando el bit B = 1. Cuando se recibe un fragmento con el bit E = 1, y todos los anteriores fragmentos han llegado correctos, el paquete PPP es reensamblado y procesado. No existe retransmisión de fragmentos, ya que PPP se basa en un mecanismo de pedido/respuesta; si no hay respuesta a un pedido tras un determinado período tiempo, el pedido es reenviado.

- **FCS:** Suma de verificación sobre los campos de cabecera y datos del fragmento.

Es necesario notar que a diferencia del tradicional formato PPP, MP-PPP no utiliza banderas de delimitación de trama, pues los campos indicados permiten realizar esta función.

2.5 *BOOTSTRAP PROTOCOL* (BOOTP)

El protocolo RARP de capa interfaz de red, permite a máquinas en una red que no poseen disco enviar un pedido a un servidor RARP durante el proceso de arranque, solicitando se asigne una dirección IP a su dirección física a fin de poder comunicarse con otras máquinas en la red. El pedido RARP va contenido en una trama que utiliza como dirección física de destino, la dirección física de difusión en dicha red.

Hay que aclarar que la dirección física de difusión no es lo misma que la dirección de difusión IP. Así por ejemplo, la dirección física en una red LAN Ethernet consiste de 48 bits 1, en tanto que la dirección de difusión IP en una red local son 32 bits 1. Es decir son direcciones utilizadas por distintas capas en el modelo estratificado.

RARP presenta algunos inconvenientes. El que un pedido RARP utilice una dirección física de difusión implica que cada red física deba tener su propio servidor RARP, ya que un ruteador, dispositivo que interconecta redes a nivel de red, no reconoce las direcciones físicas de difusión, únicamente reconoce direcciones IP y simplemente no permitiría a una máquina enviar un pedido a un servidor RARP ubicado en otra red. Otra desventaja de RARP es que únicamente proporciona a una máquina sin disco su dirección IP; esta información puede resultar insuficiente durante el arranque de una

máquina sin disco.

El Protocolo *Bootstrap* (BOOTP - *Bootstrap Protocol*) se encarga de solucionar las deficiencias de RARP. BOOTP permite a una máquina sin disco identificar a más de su dirección IP: la dirección IP de un servidor que posea la información que le permita arrancar, el nombre de un archivo que debe cargar en su memoria y ejecutarlo durante el proceso de arranque, la máscara de subred que debe utilizar, la dirección de difusión en la subred, la dirección IP de un servidor de nombres y la dirección IP de un ruteador al cual enviar sus paquetes.

Un computador en una red requiere de un dispositivo de almacenamiento para poder ejecutar el sistema operativo y el *software* de red durante el proceso de arranque. Esto implicaría que las máquinas que no poseen almacenamiento (disco flexible o disco duro) no podrían arrancar en una red. En el caso de una red TCP/IP en la cual existen máquinas que no poseen medios de almacenamiento, se utiliza un servidor central en el cual se almacenan para cada máquina una imagen de memoria. Una imagen de memoria contiene todos los archivos que una máquina requiere para su arranque, tales como: archivos del sistema operativo, *software* de red y *drivers* de los dispositivos para cada máquina.

Cada máquina sin disco al iniciarse deberá ejecutar un pequeño programa de arranque, almacenado en ROM, el cual utilizará una implementación de BOOTP contenida en la ROM (al igual que otras implementaciones de protocolos TCP/IP) para enviar mediante difusión de red un paquete de pedido de información de arranque. Este pedido será respondido por un servidor BOOTP. El paquete de respuesta enviado por el servidor informará a la máquina que hizo el pedido: la dirección IP que deberá utilizar, la dirección IP del servidor que posee su imagen de memoria y el nombre del archivo de arranque (imagen) en ese servidor. Una vez que la máquina posee la dirección del servidor, podrá utilizar el protocolo TFTP a fin de transferir desde el servidor la imagen de memoria, almacenarla en memoria RAM y proceder a arrancar. Después podrá utilizar ARP normalmente para comunicarse con otras máquinas.

Resumiendo, el arranque mediante BOOTP es un proceso de dos etapas:

1. Determinación de direcciones IP requeridas por el cliente y nombre del archivo de arranque
2. Transferencia de la imagen de memoria mediante TFTP desde un servidor.

BOOTP utiliza UDP como protocolo de transporte (Puerto 67 para el servidor y puerto 68 para el cliente), el cual a su vez utiliza IP como protocolo de red. [8] Puesto que el cliente no conoce la dirección IP del servidor BOOTP, utiliza la dirección IP especial de difusión en la red local 255.255.255.255 como dirección de destino del datagrama IP que lleva su pedido BOOTP. La dirección IP de origen en el datagrama IP sería 0.0.0.0 que significa "este *host* en esta red".

Una máquina sin disco envía, en un campo del paquete de pedido, su dirección física de *hardware*, la cual generalmente está almacenada en la ROM. El servidor al recibir un pedido buscará en su tabla para ver si tiene la dirección física de la máquina que hizo el pedido. Si su tabla tiene dicha dirección enviará una respuesta indicando las direcciones IP y el nombre del archivo de arranque correspondiente a esa máquina.

La respuesta enviada por el servidor también emplea como dirección de destino la dirección de difusión, ya que el cliente aún no conoce cual es su dirección IP. Para reconocer que un mensaje es la respuesta a su pedido, el cliente utiliza un campo en el paquete de pedido que contiene un número aleatorio que identifica la transacción. El servidor utilizará el mismo número en la respuesta enviada al cliente que hizo el pedido.

Ya que BOOTP utiliza direcciones de difusión IP y no solo direcciones de difusión físicas, a diferencia de RARP, el servidor BOOTP puede encontrarse en otra red física dentro de la red local, es decir, no se requiere un servidor BOOTP por cada red física como en RARP. Un ruteador permitirá el paso de un pedido de BOOTP hacia otras redes físicas hasta encontrar el servidor y de igual forma permitirá el paso de la respuesta hacia el cliente.

El formato de un mensaje BOOTP se presenta a continuación:

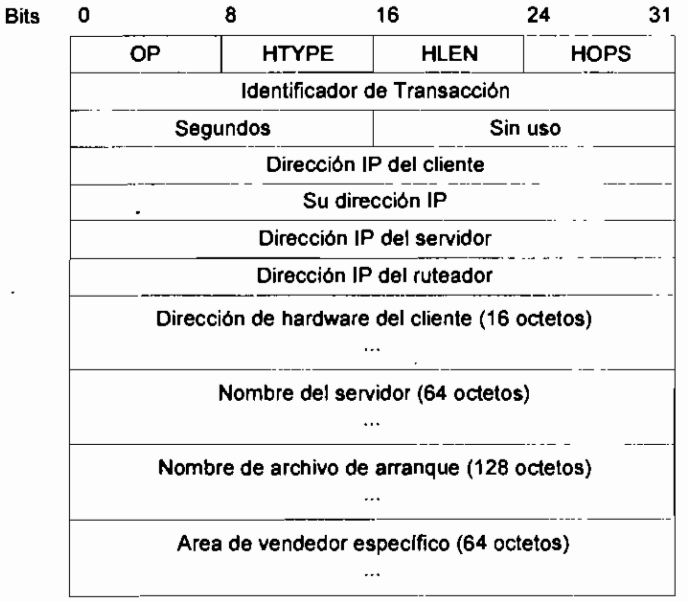


Fig. 2.15. Formato de un mensaje BOOTP [8]

- ***OP*:** Especifica si el mensaje es un pedido de un cliente (1) o una respuesta del servidor (2).
- ***HTYPE*:** Identifica mediante un código número el tipo de *hardware*.
- ***HLEN*:** Determina la longitud en octetos de la dirección física de la máquina contenida en el campo dirección de *hardware* del cliente.
- ***HOPS*:** Inicialmente puesto en cero, es incrementado por cada ruteador por el cual atraviesa el pedido hasta llegar al servidor. Sobrepasado un determinado número de saltos se elimina el datagrama que lleva el pedido.
- **ID de transacción:** Contiene un número aleatorio que permite identificar cual es la respuesta a un pedido.
- **Segundos:** Es el tiempo transcurrido en segundos desde que el cliente inició su proceso de arranque.

El cliente inicialmente pone en cero todos los campos que vienen a continuación y únicamente llena aquellos para los cuales posee información.

- **Dirección IP del Cliente:** Contiene 0.0.0.0 cuando el cliente no conoce su dirección IP y desea averiguarla. Si el cliente conoce su dirección IP, este campo contiene dicha dirección. Por ejemplo un cliente que conoce su dirección IP puede usar

BOOTP para obtener información de su archivo de arranque.

- **Su dirección IP:** Contiene la dirección IP que el servidor asignó a una máquina que envió un pedido.
- **Dirección IP del Servidor:** Si un cliente posee la dirección y nombre de un servidor que desea responda su pedido llenará los campos Dirección IP del Servidor y Nombre del Servidor, caso contrario deja en cero. La respuesta contiene en estos campos la dirección IP y el nombre del servidor en el cual se encuentra el archivo que contiene la imagen de memoria que el cliente deberá transferir usando TFTP para poder arrancar. Generalmente el mismo servidor que responde el pedido es el que posee dicha información.
- **Nombre del Servidor:** Nombre del servidor asociado a la dirección del campo Dirección IP del servidor.
- **Dirección IP del Ruteador:** El cliente podría especificar la dirección IP del ruteador que permite ir hacia el servidor que desea responda su pedido, sino el cliente lo envía en cero y el primer ruteador por el que atraviese el mensaje de pedido llena este campo con su dirección.
- **Dirección de *hardware* del cliente:** Contiene la dirección de *hardware* del cliente.
- **Nombre de Archivo de Arranque:** Puede ser utilizado por el cliente en el pedido para sugerir el nombre genérico de un sistema operativo con el cual desee arrancar, como por ejemplo *unix*. Si queda en blanco arrancará con el sistema operativo por defecto que se use en la red. La respuesta de un servidor contendrá en este campo la ruta completa (disco\ directorio\ nombre de archivo) del archivo de arranque en el servidor donde esté contenido.
- **Area de Vendedor Específico:** Contiene en la respuesta información sobre máscara de subred, dirección de servidores de nombres, direcciones de ruteadores que puede usar, dirección de difusión de subred, tamaño del archivo de arranque , etc.

BOOTP puede también ser utilizando por computadores que tengan disco duro o *drives floppy* para permitirles arrancar desde un servidor antes que desde su disco duro local o *drive floppy* local. El arranque desde un servidor es transparente para un usuario final y no tiene efecto alguno sobre las aplicaciones que el usuario desee correr en su máquina. Si la red se cae por algún motivo, los usuarios pueden arrancar localmente utilizando los archivos de arranque que están en su disco duro o *drive* local.

El permitir que un computador pueda arrancar desde un servidor antes que desde un *drive* local tiene sus ventajas:

- Si las máquinas no utilizan *drives* de almacenamiento, esto significa una considerable reducción en los costos de implementación de una red. Unicamente para implementar BOOTP se requiere instalar la ROM que contenga el código BOOTP en el *socket* de ROM de la tarjeta de interfaz de red (NIC) o incluso solo encender la máquina si el código de BOOTP viene ya almacenado en la ROM de la tarjeta madre de la computadora eliminando así la necesidad de una ROM adicional.
- Permite al administrador de la red actualizar la red en forma rápida y sencilla. Por ejemplo cuando un nuevo *software* o sistema operativo llega a ser disponible, el administrador puede actualizar a todos los clientes de la red desde su escritorio sin necesidad de instalar los archivos y configurarlos individualmente en cada máquina.
- Un sistema de arranque desde un servidor permite mejorar la seguridad de la red, ya que los archivos de arranque almacenados en el servidor están protegidos ante modificaciones de usuarios finales, daño o conspiración. Se puede incluir *software* para detectar y destruir virus en el sector de arranque.
- El administrador puede solucionar problemas de arranque de una máquina desde su escritorio sin necesidad de visitar la máquina

El administrador de la red debe encargarse de mantener la tabla en el servidor BOOTP actualizada y registrar para cada máquina la imagen de memoria y dirección IP que ésta utilizará en la red.

2.6 *DYNAMIC HOST CONFIGURATION PROTOCOL* (DHCP)

El protocolo BOOTP fue diseñado para un ambiente relativamente estático en el que cada *host* tiene una conexión de red permanente. [8] Por lo tanto BOOTP utiliza una asignación estática de direcciones IP. Esto significa que en la base de datos que utiliza el servidor BOOTP, para cada máquina existe asignada una única dirección IP la cual deberá utilizar siempre. Esta dirección únicamente podrá ser utilizada por esa máquina y por ninguna más en la red a pesar de que la máquina no esté activa en algún instante. Cuando una nueva máquina quiera conectarse a la red, el administrador primeramente

debería modificar la tabla, asignando a esa máquina una nueva dirección IP y los parámetros que deberá utilizar en su arranque.

Existen redes en las cuales un *host* (portátil o inalámbrico) puede conectarse a una red solo temporalmente sin necesidad de requerir una dirección IP permanente. También existen redes en las cuales el número de direcciones IP disponibles no es tan grande como el número de máquinas que podrían en un momento dado querer conectarse, y redes que cambian constantemente. En estos casos un esquema de asignación estática de direcciones no podría utilizarse ya que resultaría imposible o muy difícil mantener actualizadas las tablas en los servidores que asignan direcciones IP. Se requiere de un sistema de asignación de direcciones dinámico, que permita a un *host* conectarse temporalmente a una red, asignándole una dirección IP disponible de un conjunto limitado de direcciones IP que se comparten entre un grupo de *hosts*, los cuales no requieren direcciones permanentes, reasignando una dirección que ha dejado de ser utilizada por una máquina y asignándola a una nueva máquina que la requiera, sin necesidad de estar modificando las tablas en los servidores.

El Protocolo de Configuración Dinámica de *Host* (DHCP - *Dynamic Host Configuration Protocol*) soluciona los problemas anteriores. DHCP es un protocolo encargado de dos aspectos:

1. Entregar a un *host*, que se conecta a una red, información de parámetros de configuración específicos para ese *host* desde un servidor y,
2. Asignar direcciones de red a *hosts* en forma automática.

DHCP es muy similar a BOOTP y posee cierta compatibilidad con él. Los formatos de los mensajes DHCP son prácticamente idénticos a los de BOOTP, tan solo con pequeñas diferencias que posteriormente se verán. A diferencia de BOOTP, que requiere preconfigurar manualmente la información de los *hosts* en una base de datos en un servidor, DHCP permite realizar automáticamente la asignación de direcciones de red y entrega de parámetros de configuración que los *hosts* requieren para conectarse a una determinada red evitando la configuración manual de cada máquina en el servidor y ocultado la necesidad de reconfigurar la máquina cuando alguna de la información de

configuración ha cambiado.

DHCP provee tres mecanismos para asignar direcciones IP a un *host*: [9]

- **Asignación manual:** La dirección IP de un *host* es asignada por el administrador de red en forma estática al igual que en BOOTP, es decir el *host* siempre utilizará la misma dirección IP cuando se conecte a la red. DHCP se usa simplemente para transportar la dirección asignada al *host*.
- **Asignación automática:** DHCP asigna a un *host* una dirección IP disponible, en forma permanente, durante todo el tiempo que permanezca conectado a una red. Esto no implica que al *host* siempre se le asigne una misma dirección IP cada que se conecte a la red, sino únicamente que la dirección asignada será permanente durante el tiempo que dure su sesión de trabajo.
- **Asignación dinámica:** La asignación dinámica, se basa en un sistema de arrendamiento de direcciones, en el cual el servidor presta una dirección IP a un cliente por un período de tiempo establecido por el administrador de la red. El período de tiempo asignado para que un cliente use la dirección IP no necesariamente puede ser el tiempo que dure su sesión de trabajo, sino más corto o más largo. Dependiendo de la duración de la sesión y del tiempo asignado de uso de la dirección, el *host* utilizará DHCP para que nuevamente se le reasigne la misma dirección durante otro período de tiempo o se le asigne una nueva dirección IP disponible a fin de seguir trabajado. Por el contrario, si el cliente termina su sesión antes de que el tiempo asignado de posesión de la dirección IP termine, el cliente la puede abandonar, permitiendo que esta dirección pueda ser utilizada por otro *host* que la necesite.

En una red pueden existir *hosts* que se encuentren permanentemente conectados y *hosts* que pueden conectarse temporalmente. Para poder implementar los mecanismos de DHCP el administrador de red debe crear una base de datos en el servidor DHCP en la cual se especifique:

- Las direcciones asignadas en forma estática (fija) a ciertas máquinas (generalmente las permanentemente conectadas) y sus parámetros de configuración.

- El conjunto de direcciones que se pueden asignar automáticamente en forma permanente, y
- El conjunto de direcciones que se pueden asignar dinámicamente mediante un sistema de arrendamiento.

A más de esta base de datos, un servidor DHCP requiere mantener otra tabla en la cual almacena los parámetros de configuración y direcciones asignadas a cada *host* actualmente conectado a la red. Esta tabla permite garantizar que una dirección asignada a un *host*, como respuesta a un pedido, sea única y no esté en conflicto con una dirección previamente asignada. Esta tabla también permite reasignar los mismos parámetros y direcciones a un *host* en posteriores pedidos.

En base a estas tablas el servidor DHCP realizará la asignación de direcciones y configuraciones a los *hosts* que se conecten a una red.

2.6.1 Asignación de direcciones IP mediante DHCP

DHCP utiliza el modelo cliente - servidor en su comunicación, en forma muy similar a BOOTP. DCHP envía sus mensajes empleando UDP como protocolo de transporte y hace uso de los mismos puertos UDP que BOOTP (puerto 67 para el servidor y puerto 68 para el cliente). [8]

La comunicación entre un cliente que se conecta a una red o que inicia una aplicación que requiere de IP y un servidor procede en base a un intercambio de mensajes definidos por DHCP de la siguiente forma: [9]

1. El cliente utiliza la dirección IP de *broadcast* en la red local (255.255.255.255) para difundir un mensaje *DHCPDISCOVER*, en todas las subredes físicas de la red local, en busca de un servidor que responda a su pedido. El mensaje *DHCPDISCOVER* incluye: un número aleatorio que identifica la transacción al igual que en BOOTP, y algunas opciones tales como: la sugerencia de la dirección de red, duración del arrendamiento, etc. Una solicitud de arrendamiento de una dirección en forma permanente para toda la sesión se lo representa utilizado un tiempo infinito

expresado como 0xffffffff. Puesto que se usa la dirección de difusión de red, el pedido puede ser respondido por uno o varios servidores ubicados en la red local, no necesariamente en la misma red física.

2. Cada servidor que recibe el pedido puede responder con un mensaje *DHCPOFFER* que incluye una dirección de red disponible y otras opciones de configuración según el criterio asociado para ese cliente en su base de datos. El servidor envía su respuesta utilizando la dirección de *broadcast* y el mismo identificador de transacción utilizado por el cliente en su pedido.
3. El cliente recibe uno o más mensajes *DHCPOFFER* desde uno o más servidores. El cliente escoge uno en el cual los parámetros de configuración se aproximen a los solicitados en su pedido ó el primero que llegue, y envía en forma de *broadcast* un mensaje *DHCPREQUEST*, el cual incluye la identificación del servidor que el cliente seleccionó.
4. Unicamente el servidor seleccionado al recibir el mensaje *DHCPREQUEST* actualizará su tabla reservando la dirección asignada para ese cliente y responderá con un mensaje *DHCPACK* conteniendo los parámetros de configuración para la petición del cliente. La combinación de la dirección de *hardware* del cliente y la dirección de red asignada constituyen un identificador único para el arrendamiento del cliente y son usados por ambos: el cliente y el servidor, para identificar un arrendamiento referido a cualquier mensaje DHCP.

El cliente en la respuesta enviada por un servidor recibirá, dependiendo de si el cliente está permanente o temporalmente conectado a la red, la siguiente información:

- La dirección IP que aparece en la base de datos del servidor, en el caso de que el cliente tenga asociada una dirección . estática, sino DHCP se encarga automáticamente de asignar y enviar una dirección IP temporal del conjunto de direcciones IP disponibles, apropiada para la red en la que está en ese instante conectado el cliente.
- La máscara de subred IP, la dirección IP de *broadcast*, la dirección del ruteador por defecto y una lista de direcciones de servidores de nombres de dominio de Internet DNS apropiados para la red en la cual el dispositivo está conectado.

- Además recibe el tiempo de expiración asociado con el arrendamiento de la dirección IP.

5. El cliente al recibir el mensaje *DHCPACK* con los parámetros de configuración ejecuta un chequeo final de los mismos, por ejemplo utilizando ARP para ver si la dirección asignada no está siendo usada por otro *host* y si no hay problema almacena la identificación de su arrendamiento y su duración. En caso de que al chequear la dirección, se recibiese un mensaje ARP de respuesta conteniendo una dirección física de destino, esto significaría que la dirección ha sido ya asignada, tal vez por otro servidor, a otro *host*. En este caso el cliente enviará un mensaje *DHCPDECLINE* a su servidor para rechazar su ofrecimiento y reiniciará el proceso de configuración.
6. Si el cliente termina su sesión antes de que el período de tiempo asignado por el servidor expire, el cliente abandona el alquiler de su dirección enviando el mensaje *DCHPRELEASE* al servidor el cual contiene el identificador del arrendamiento.

Si el *host* cliente permanece conectado a la red en la misma localidad y continua usando la dirección IP, periódicamente contactará al servidor DHCP que otorgó el arrendamiento para renovarlo. Este proceso continuará hasta que la computadora del cliente sea apagada o deje de usar IP.

Para proveer un margen de seguridad, el cliente debe intentar renovar su arrendamiento antes de que el tiempo expire. Por ejemplo el cliente puede intentarlo a la mitad del período de arrendamiento. Si el servidor particular DHCP que otorgó el arrendamiento no está disponible, el cliente continuará sus intentos de renovación (sin efectos para el usuario) hasta que el servidor responda, o el arrendamiento expire.

Cuando el cliente desea reutilizar la dirección previamente asignada durante otro período de tiempo la comunicación cliente-servidor procede de la siguiente forma:

1. El cliente envía un mensaje *DHCPREQUEST* al servidor que le asignó la dirección. Este mensaje contiene la dirección de red que está usando el cliente.
2. Si el servidor permite al cliente seguir usando la dirección, envía un mensaje

DHCPACK, caso contrario envía un mensaje *DHCPNACK*. En este último caso, el cliente deberá abandonar la dirección IP asignada y reiniciar el proceso de asignación de dirección como se describió anteriormente.

Puede darse el caso de que el servidor que asignó la dirección no responda. El cliente en este caso debe enviar un mensaje de difusión *DHCPREQUEST* para que otro servidor que conoce los parámetros de configuración del cliente, le responda con un mensaje *DHCPACK* asignándole una nueva dirección. El cliente chequea los parámetros enviados por el nuevo servidor y registra la duración del nuevo arrendamiento y la identificación del mismo, volviendo el cliente a quedar configurado. Si hay un problema de parámetros el cliente envía el mensaje *DHCPDECLINE* y reinicia el proceso de configuración.

Si no encuentra ningún otro servidor, el cliente usará la misma dirección por el tiempo restante hasta que éste expire.

Cuando el cliente decida terminar de usar la dirección o el tiempo expire, el cliente abandona el alquiler de su dirección enviando al servidor el mensaje *DCHPRELEASE* el cual contiene el identificador del arrendamiento.

En el caso de que el tiempo de arrendamiento haya expirado, debido a que todos los intentos de renovación han fallado, el *stack* de protocolos TCP/IP del cliente es responsable de finalizarse por si mismo, terminando todas las tareas basadas en IP que el usuario haya estado ejecutando.

El formato de un mensaje DHCP, nombres y significados de los campos es idéntico al formato del mensaje BOOTP mostrado en la figura 2.15, con excepción de dos campos. El campo Sin Uso de BOOTP, en DHCP se lo denomina campo Banderas y el campo Area de vendedor específico de BOOTP se lo denomina campo Opciones.

DHCP utiliza el primer bit del campo Banderas para solicitar a un servidor que las respuestas a un mensaje enviado por un cliente se envíen en forma de difusión cuando está puesto en (1), o en forma *unicast* (0) directamente a la dirección asignada al cliente,

cuando éste ya la conoce.

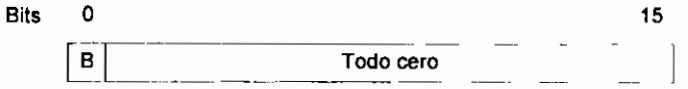


Fig. 2.16. Formato del campo Banderas de DHCP [9]

El campo Opciones es un campo de longitud variable, en el cual se define el tipo de mensaje en un paquete de 24 bits de la siguiente forma:

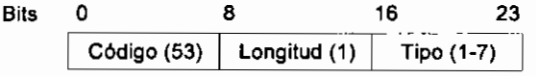


Fig. 2.17. Formato del campo Opciones de DHCP [9]

Este campo de Opciones además contiene los parámetros de configuración enviados por el servidor ante el pedido de un cliente, sugerencias de tiempos de arrendamiento, etc.

Los diferentes tipos de mensajes DHCP se presentan a continuación:

Campo de Tipo	Tipo de Mensaje
1	DHCPDISCOVER
2	DHCPOFFER
3	DHCPREQUEST
4	DHCPDECLINE
5	DHCPACK
6	DHCPNACK
7	DHCPRELEASE

Tabla. 2.6. Códigos y Tipos de mensajes DHCP [8]

Finalmente hay que mencionar que un mismo *host* puede contener varias interfaces de red. Para cada interfaz el *host* debe utilizar en forma separada DHCP para asignar una dirección de red y parámetros de configuración adecuados para esa red.

El campo de Dirección IP de Ruteador al igual que en BOOTP contiene la dirección del ruteador que permite llegar hacia un determinado servidor o si es dejado en blanco será llenado con la dirección del primer ruteador por el que atraviesa el mensaje. Ya que un mismo servidor DHCP puede servir para varias redes, la dirección IP del ruteador permite al servidor DHCP reconocer en que red se encuentra el cliente y de esta forma asignarle la dirección de difusión, máscara de subred, etc., apropiadas para esa red.

REFERENCIAS (Capítulo II)

[1] "*PPP Authentication Protocols*", RFC 1334, October 1992

[2] Tanenbaum, Andrew S.: "*Computer Networks* ", Third Edition, Prentice Hall, New Jersey, 1996

[3] "*Remote Authentication Dial In User Service* (RADIUS)", RFC 2138, April 1997

[4] "*RADIUS Accounting*", RFC 2139, April 1997

[5] http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t1/l2tp.htm "*Layer 2 Tunneling Protocol*"

[6] http://www.townsley.net/mark/l2tp/l2tp-latest.txt, "*Layer 2 Tunneling Protocol* (L2TP)"

[7] "*The PPP Multilink Protocol* (MP)", RFC 1990, August 1996.

[8] Comer, Douglas E.: "*Redes globales de informacion con Internet y TCP/IP*", Prentice Hall, México, 1996.

[9] "*Dynamic Host Configuration Protocol*", RFC 1541, October 1993

INDICE - CAPITULO III

CAPITULO III

ARQUITECTURA BASICA DE UN ISP

3.1 ARQUITECTURA DE LA RED GLOBAL INTERNET: NAPs, PoPs e ISPs

El Internet es realmente una colección de redes interconectadas entre sí formando una única gran red virtual global. Internet no pertenece a ninguna compañía o país específico. Sus principales elementos constitutivos pertenecen a grandes empresas de telecomunicaciones, pero la "Red" en si misma no pertenece a nadie.

La siguiente figura muestra los componentes que forman parte de la arquitectura de la red global Internet:

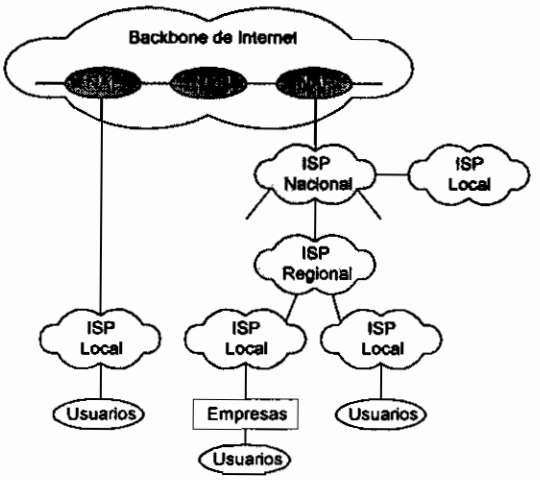


Fig. 3.1. Arquitectura simplificada de la red global Internet [1]

NAP: Al *backbone* de alta velocidad de Internet están interconectadas redes nacionales/regionales a través de varios puntos denominados NAPs (*Network Access Points*).[A]

[A] El *backbone* de Internet está formado principalmente por la red estadounidense de la Fundación Nacional de Ciencia (NSFNET - *National Science Foundation NETwork)* y por otras redes *backbone* de distintos países interconectadas entre sí. Los enlaces de alta velocidad y ruteadores del *backbone* son propiedad de grandes compañias de telecomunicaciones, tales como: MCI, Sprint, Wordlcomm, etc.

PoP: Un PoP (*Point of Presence*) es un punto de entrada a Internet. El propósito principal de los NAPs es proveer la capacidad de ruteo e interconexión entre PoPs de ISPs.

ISP: Un ISP (*Internet Service Provider*) es una compañía que tiene acceso a Internet y que vende su habilidad de conectarse a Internet a miembros del público general, a través de uno o varios PoPs, dependiendo del tamaño del ISP.

En forma general, un ISP puede ofrecer a sus usuarios el servicio de acceso a información contenida en el propio proveedor, así como también ofrecer el servicio de pasarela hacia proveedores de información ubicados en Internet.

Hay varias formas en que un proveedor puede estar conectado a Internet; normalmente el proveedor estará conectado mediante algún tipo de línea de telecomunicaciones que provee un ancho de banda mucho mayor del que cualquier usuario individual podría necesitar o costear. La capacidad y costo de este medio de comunicación son "compartidas" entre todos los usuarios del proveedor.

Los ISPs son negocios que varían ampliamente de tamaño. Un ISP pequeño puede tener un único PoP y estar conectado a un NAP, usando un enlace propio.

Otra posibilidad para un ISP pequeño es aparecer como uno de los PoPs de un ISP más grande, evitando el enlace costoso hacia el NAP. Un ISP grande puede poseer su propia red de datos que interconecte todos sus PoPs entre si y a la Internet global a través de un NAP. Los PoPs de un gran ISP pueden incluso llegar a estar difundidos en todo el mundo, ofreciendo PoPs en varios países. Puede llegarse a crear una verdadera jerarquía: ISPs locales conectados a ISPs regionales, los cuales se conectan a ISPs nacionales, y éstos a su vez a ISPs mundiales, conectados al NAP.

Las formas más comunes de conexión de un ISP hacia un proveedor superior (ISP de mayor jerarquía o NAP) son generalmente mediante enlaces dedicados terrestres o satelitales: T1/E1, fraccionales, ISDN PRI, T3/E3, Frame Relay, ATM, etc. Se estima que ATM será la tecnología más utilizada en el futuro. Estos enlaces son generalmente

proporcionados por las grandes empresas de telecomunicaciones o compañías de servicios portadores nacionales y/o internacionales.

Los servicios que los ISPs ofrecen en la actualidad a sus usuarios van desde *e-mail* básico hasta servicios que brinden a una compañía una completa presencia en Internet incluyendo página Web, catálogos de productos, comercio electrónico, órdenes seguras en línea, así como también soporte a usuarios con audio y vídeo en tiempo real.

La arquitectura general y elementos presentes en un ISP, se presenta a continuación:

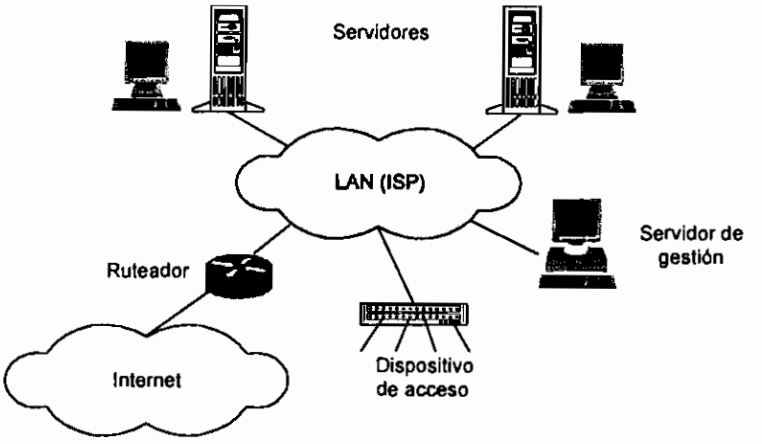


Fig. 3.2 Arquitectura general de un ISP [1]

Básicamente un ISP consiste de servidores ejecutando *software* que proveen varios servicios: *E-mail* (SMTP, POP3), HTTP, FTP, DNS, IRC, etc. También incluye ruteadores que proveen conectividad al Internet y dispositivos que brinden acceso a usuarios.

Implementar una red para un ISP requiere muchas decisiones entre varias plataformas, *hardware*, *software* y opciones de conectividad. Además, requiere cálculos de capacidad, mercado, etc. Ninguno de estos aspectos se trata en este trabajo.

La arquitectura básica detallada se mantendrá común en todas las tecnologías de acceso que se analizarán en el capítulo IV. Sin embargo, dependiendo del tipo de tecnología empleada para brindar acceso a usuarios y la tecnología empleada por el ISP para conectarse a Internet se requerirán dispositivos de acceso con características específicas

y en ciertos casos elementos adicionales. De la tecnología de acceso empleada dependerán las velocidades de acceso y la calidad de los servicios que un ISP pueda ofrecer a sus usuarios.

3.2 INFRAESTRUCTURA DE UN ISP TIPICO

Nos referimos a un ISP típico a aquel que brinda a sus usuarios acceso mediante línea telefónica, empleando módems analógicos, y se conecta a un proveedor superior o NAP empleando líneas dedicadas T1/E1 o superiores, alquiladas a la red telefónica pública o a un proveedor de servicios portadores.

La arquitectura y elementos empleados por un ISP típico se presentan a continuación.

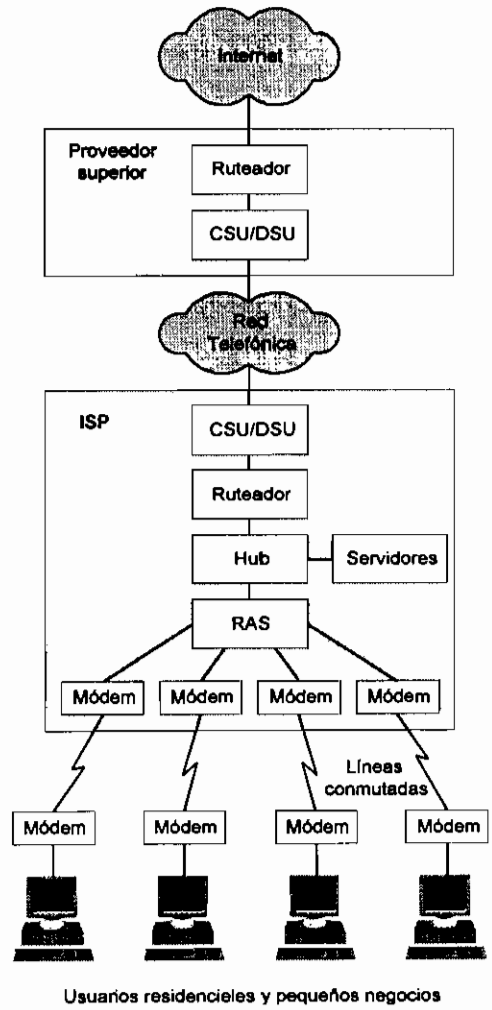


Fig. 3.3. Arquitectura de un ISP típico [1]

Servidores: Como se mencionó en el capítulo I, las aplicaciones TCP/IP utilizan el modelo cliente-servidor como la base de su funcionamiento. Un ISP poseerá servidores de aplicaciones a disposición de sus usuarios (clientes). Dependiendo de los servicios que un ISP ofrezca poseerá servidores de aplicaciones para: DNS, HTTP (Web), FTP, IRC (Chat), USENET (News), POP3, SMTP, audio/vídeo en tiempo real, comercio electrónico, etc. A más de los servidores mencionados se requerirá de servidores para autenticación/tarifación de usuarios (RADIUS o TACACS+) [B] y de gestión de red.

En la mayoría de casos, una misma máquina (o servidor físico) puede correr simultáneamente varios servidores de aplicaciones (*software*).

Aunque existen gran variedad de sistemas operativos usados como plataformas para proveer servicios en Internet, son dos los que destacan por su gran empleo: *Unix* y *Windows NT.*

- **Unix**

Existen versiones de UNIX tanto comerciales (*AIX, SCO Unix, Solaris*) como gratuitas (*FreeBSD o Linux*). UNIX incluye el *software* de servidor de FTP, correo electrónico, News, y Telnet. Debe añadirse el *software* servidor de WWW para obtener una configuración completa. El *software* servidor de WWW puede ser uno comercial ó versiones gratuitas, ó de bajo costo, que se encuentran disponibles en la red, tales como: *NCSA, Apache* o *CERN.*

Dada esta disponibilidad de *software* sin coste y el funcionamiento sobre plataformas estándar, esta configuración constituye el servidor más barato disponible, incluso aunque se elija un *hardware* de un fabricante de reconocida marca y precio.

[B] TACACS es un servidor de autenticación descrito en el RFC 1492

- **Windows NT Server**

Windows NT Server es el único sistema operativo de red con un servidor Web integrado: *Microsoft Internet Information Server* (IIS). La integración de IIS con *Windows NT Server 4.0* significa que la instalación y administración del servidor Web es, simplemente, otra parte del sistema operativo.

Con la instalación integrada del *Internet Information Server*, y la capacidad de programación mejorada y más rápida, *Windows NT Server 4.0* resulta ser una plataforma ideal para Internet. *Windows NT Server* está compitiendo fuertemente con las plataformas UNIX, las cuales poseen en la actualidad un predominio casi absoluto en el mercado de servidores.

Windows NT Server, a diferencia de otras plataformas, trae incorporado un servidor de DNS, sin embargo, requiere la instalación de productos adicionales para brindar servicios tales como: correo electrónico o News. Muchos de estos productos están disponibles en la red de forma gratuita o a un bajo precio.

Los requisitos de *hardware* mínimos que debe cumplir el ordenador tanto para plataformas basadas en *Windows NT* o UNIX son: un procesador 486DX4 o superior, 16 *Megabytes* de memoria y un *Gigabyte* de disco duro. Está por demás mencionar que mientras mejores sean las características de la máquina, mejor será la ejecución del servidor.

Ruteador: Es el elemento más importante requerido en una conexión de un ISP hacia un ISP de jerarquía superior o NAP. Es el responsable del flujo de datagramas IP entre el ISP y el núcleo de Internet en ambas direcciones.

Su principal función es la de examinar las cabeceras IP y decidir por donde deben ser enviados los datagramas. Esta función aparentemente simple debe ser realizada a velocidades extremadamente altas por lo que es recomendado utilizar un ruteador de propósito especial, debido a que es mucho mas rápido que utilizar un computador dedicado a realizar ruteo.

Para un ISP básico, el ruteador debería tener al menos dos interfaces: una para el proveedor del *backbone* y otra para la red local del ISP. Sin embargo, dependiendo del tipo de ancho de banda que soporte el ISP, el ruteador puede soportar otras interfaces, una para cada circuito de datos dedicado.

El protocolo de enrutamiento empleado en el ruteador del ISP debería ser idéntico al empleado en el ruteador del proveedor superior o NAP. Los protocolos más comunes son: RIP, OSPF y BGP-4.

CSU/DSU: Este dispositivo provee la interconexión entre la red LAN del ISP y la compañía telefónica o portadora, a la cual el ISP alquila un enlace digital (T1/E1, fraccional, T3/E3) hacia otro ISP superior o NAP. Aunque a menudo se lo refiere como un solo equipo brinda varias funciones:

El DSU (*Data Service Unit*) provee una interfaz digital estándar al DTE del usuario (el ruteador), generalmente compatible con V.35 y RS-449/422. El DSU convierte las tramas provenientes desde esta interfaz en tramas multiplexadas por división en tiempo usadas en el enlace digital y viceversa.

El CSU (*Channel Service Unit*) provee la interfaz hacia la central telefónica o portadora, realiza acondicionamiento de la línea, ecualización y provee capacidades de diagnóstico. La interfaz WAN con el circuito telefónico puede utilizar un conector RJ-48C, conectores para fibra óptica, etc.

Muchos ruteadores traen ya incorporado el CSU/DSU.

Hub: Este equipo conecta el equipo en la red local del ISP tal como ruteadores, servidores, etc., en una topología física tipo estrella. Esto ayuda en la gestión debido al hecho de que un defecto es aislado en el segmento. El tipo de *hub* depende de la red LAN que el ISP disponga, pudiendo existir *hubs* para: Ethernet/10 Base-T, Fast Ethernet, Token Ring, FDDI, y ATM. Los *hubs* más comúnmente usados son Ethernet con conectores RJ-45. Dependiendo del número de equipos en la LAN del ISP pueden requerirse varios *hubs* conectados en cascada.

Switches: En vez de un *hub* puede utilizarse un *switch*, el cual mejora el rendimiento, garantiza ancho de banda fijo para las distintas comunicaciones en la LAN, sin necesidad de compartir el ancho de banda entre todos los dispositivos en un momento conectados a la red, como lo hacen los *hubs*.

En la actualidad existen *switches* que trabajan a nivel de capa 3, que permiten conectar en forma transparente varios tipos de redes físicas, separarlas en redes virtuales lógicas (VLANS - *Virtual LANs*), permitir el ruteo o no entre redes virtuales, autorizar o no el acceso de usuarios a una red lógica virtual en base a políticas, etc. Estos *switches* incrementan el desempeño y posibilidades de conexión de los distintos elementos en la red del ISP.

RAS: El Servidor de Acceso Remoto (RAS - *Remote Access Server*) es un dispositivo utilizado para conectar a usuarios remotos, que utilizan PCs con módems, a la LAN del ISP mediante conexiones *dial in*. Dentro de este grupo de usuarios están comprendidos los usuarios residenciales y pequeños negocios.

Un RAS, generalmente posee una interfaz LAN del tipo: 10 Base-T o Token Ring, mediante la cual se conecta al *hub/switch*, y muchos puertos seriales donde se conectan varios módems propiedad del proveedor, que terminan el enlace punto a punto con el usuario. Al conjunto de módems del proveedor se le conoce como banco de módems o *pool* de módems. Dependiendo del número de puertos seriales que disponga el RAS, será necesario tener más de uno para atender al número total de usuarios del ISP.

La principal función del RAS es la de capturar información de autenticación desde el cliente y enviarla al servidor de autenticación para aprobarla. Una vez que la autenticación es aprobada, el RAS asigna una dirección IP al cliente, empleando PPP (IPCP). La dirección IP asignada puede basarse en un nombre de usuario o puede ser asignada dinámicamente a partir de un conjunto de direcciones IP disponibles, configuradas en el RAS.

El RAS actúa como agente intermedio entre el cliente y un servidor de autenticación/tarifación tal como RADIUS o TACACS+. El RAS para realizar sus funciones soporta gran cantidad de los protocolos del *stack* TCP/IP, tales como: PPP, SLIP, MP-PPP, SNMP, ARP, *proxy* ARP, IP, ICMP, TCP, UDP, TFTP, BOOTP, TELNET, compresión de cabeceras de Van Jacobson, etc. Un RAS puede ser administrado y configurado desde cualquier estación de trabajo de la LAN del ISP.

Para los usuarios corporativos que requieren conexiones dedicadas de alta velocidad: enlaces T1/E1, fraccionales T1/E1, ISDN PRI, [C] la manera usual de establecer la conexión con un ISP es utilizar ruteadores en ambos extremos del enlace dedicado. El RAS no se emplea en este caso. [D] El enlace dedicado entre el usuario y el ISP es provisto por la compañía telefónica local, un proveedor de servicios portadores o pueden ser enlaces privados autorizados. El enlace dedicado puede ser mediante par trenzado, fibra óptica, radioenlaces, etc. Dependiendo del tipo de enlace cada ruteador se conectará a un CSU/DSU, radio módem, FRAD, etc.

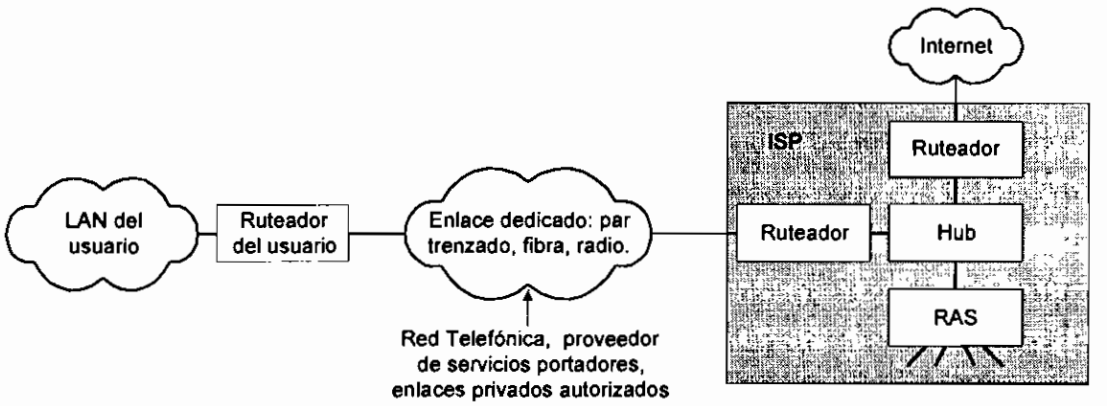


Fig. 3.4. Conexión de un usuario corporativo a un ISP típico usando un enlace dedicado

El servicio típico en un enlace de una red corporativa a un ISP generalmente incluye:

[C] El enlace del ISP hacia Internet, debe tener la capacidad suficiente para garantizar el ancho de banda de estos enlaces dedicados.

[D] Existen ciertos RAS que traen incorporadas un cierto número límitado de interfaces para enlaces dedicados de alta velocidad.

- Negociación IP y DNS
- Instalación de un servidor DNS secundario
- Instalación de un servidor DNS primario (opcional)
- Web *Hosting*, etc.

Banco de Módems: Entre el RAS y las líneas telefónicas existen módems que modulan el flujo binario saliente a una portadora analógica y demodulan el flujo binario entrante desde la portadora analógica. Un RAS puede ya traer incorporados internamente tantos módems como líneas telefónicas soporte.

Líneas Telefónicas: Un ISP generalmente se conecta con la central telefónica local mediante enlaces T1/E1 y emplea servicios PBX para separar cada línea telefónica a un módem independiente. El número de enlaces T1/E1 depende de la capacidad de usuarios que el ISP vaya a soportar.

3.3 ELEMENTOS ADICIONALES RECOMENDADOS EN UN ISP

Otros elementos adicionales que el ISP puede utilizar son: un *firewall* y un servidor *proxy/cache*. Estos elementos también se encuentran presentes en redes de organizaciones u empresas que se conectan al Internet.

3.3.1 *Firewall*

La información que una organización, empresa o proveedor de servicio posee es un recurso valioso que se debe proteger. Un *firewall* permite proteger o reducir el riesgo de una amenaza externa a la información contenida en una red de computadores.

Un *firewall*, divide una red en dos partes, una red segura y una red no segura o desmilitarizada. El *firewall* posee una interfaz de red con cada una de estas redes.

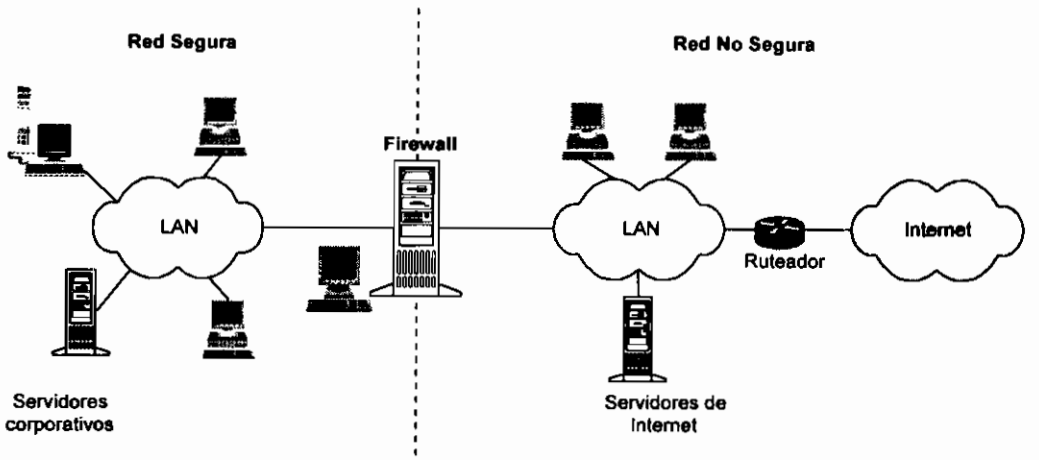


Fig. 3.5. *Firewall*, red segura y red no segura.

En la red segura, se ubicarán todos los servidores (correo, FTP, DNS) y máquinas que contienen la información que no se desea ofrecer libre acceso desde redes externas. En cambio, en la red no segura se ubicarán generalmente servidores (Web, FTP, DNS, correo, etc.) que contienen información a la cual la empresa/organización desea dar libre acceso desde redes externas tal como Internet, y que en caso de un fallo o ataque externo a estos servidores no involucrará grandes pérdidas para la empresa, a más del trabajo de volverlos a poner activos.

El objetivo del *firewall* es proteger los recursos localizados en la red segura de amenazas desde otras redes, tales como: modificación o daño de la información contenida en bases de datos, envío de grandes archivos mediante FTP que harían lenta a la red, robo de secretos de la empresa, virus, etc.

Un *firewall*, instalado en una empresa/organización, también permite al administrador de la red definir políticas de seguridad tales que no permitan a los usuarios dentro de la red segura poder acceder a recursos en el Internet que posiblemente no tengan nada que ver con el negocio o actividad de la empresa (Ejm. páginas pornográficas, chat, etc.).

Dependiendo de políticas de seguridad definidas en el *firewall*, se puede permitir el acceso de usuarios remotos a la red segura (Ejm. Teletrabajo) usando mecanismos de autenticación, protocolos para redes virtuales, etc.

Generalmente, un *firewall* por motivos de seguridad incluye o trabaja con un servidor *proxy*. Esto permite ocultar las direcciones IP usadas en la red segura. Además, el *firewall* puede realizar traslaciones de nombres de dominios empleados en la red segura con otros empleados en la red no segura, creación de túneles seguros entre servidores ubicados en la red segura con servidores ubicados en la red no segura, túneles seguros entre *firewalls*, etc. Todos estos y otros mecanismos de seguridad evitan o dificultan que *hackers* que hayan monitoreado constantemente la red puedan adquirir información que les permita atacarla.

El funcionamiento de un *firewall*, hoy en día es muy complejo, involucra tanto *hardware* como *software*, definición de políticas de seguridad, definición de servicios a los cuales se brindará acceso o no, personalización de servicios según el usuario, etc. El *firewall* en base a las políticas de seguridad definidas se encargará de filtrar los paquetes, permitiendo la salida o entrada de pedidos/respuestas desde y hacia la red segura.

Un ISP podría emplear un *firewall* para proteger las máquinas en las cuales almacena información de tarifación, de usuarios, realiza desarrollo de aplicaciones, etc.

3.3.2 Servidor *Cache*

Un servidor *cache*, es un servidor muy relacionado con los usuarios de Internet y está típicamente dentro de una organización/empresa/proveedor para poder almacenar páginas Web y posiblemente archivos FTP u otros que los usuarios del servidor hayan solicitado, de tal forma que pedidos posteriores a estas mismas páginas puedan ser satisfechos por el servidor *cache* antes de requerirse que el usuario deba obtenerla desde Internet. Un servidor *cache* no solo permite a sus usuarios obtener la información más rápido sino también permite reducir el tráfico de Internet.

Un servidor *cache* generalmente también es un servidor *proxy*.

3.3.3 Servidor *Proxy*

En una red privada (organización, empresa, proveedor) que usa el Internet, un servidor *proxy* actúa como un intermediario entre un usuario en una estación de trabajo y el Internet.

Un servidor *proxy* recibe un pedido para un servicio de Internet (tal como un pedido de página Web) desde un usuario. Si el pedido pasa los requerimientos de filtrado del *firewall* (en caso de existir), el servidor *proxy*, asumiendo que también es un servidor *cache*, busca en su *cache* local de páginas Web previamente bajadas. Si encuentra la página, la devuelve al usuario sin necesidad de enviar el pedido al Internet. Si la página no está en el *cache*, el servidor *proxy*, actúa como un cliente en nombre de sus usuarios, usa una de sus propias direcciones IP para solicitar la página desde el servidor fuera en Internet. Cuando la página es obtenida, el servidor *proxy*, la relaciona con el pedido original y la envía al usuario.

Un servidor *proxy* emplea la función NAT (*Network Address Translator*), la cual traduce direcciones IP privadas no válidas (direcciones de Intranet) ó que la empresa por razones de seguridad desea mantener ocultas, en una dirección IP global (válida de Internet).

La función NAT realiza la correspondencia entre una dirección IP local y un puerto (TCP/UDP) con una direción IP global y un puerto (TCP/UDP).[E] Por ejemplo, cuando un pedido HTTP es enviado mediante una conexión TCP a:

{dirección IP destino, dirección IP local, puerto de destino (80), puerto de origen (X)}

NAT cambia los campos correspondientes del datagrama IP, y segmento TCP o datagrama de usuario UDP con:

{dirección IP destino, dirección IP global, puerto de destino (80), puerto dc origen (Y)}

[E] A esta combinación se la denomina *socket*. Una comunicación extremo a extremo en TCP/IP está definida por los *sockets* en cada extremo.

donde Y es un número de puerto dinámicamente asignado por NAT. NAT almacena en una tabla interna los valores: [dirección IP global, puerto de origen (Y)] y [dirección IP local, puerto de origen (X)] a fin de realizar la función de traducción inversa.

De esta forma, se puede utilizar un única dirección IP global para dar servicio a múltiples usuarios.

En el caso de que se disponga de múltiples direcciones IP globales (y por lo tanto, se puedan crear múltiples sesiones simultáneas), la asignación de la dirección IP local a una dirección IP global puede ser regulada mediante filtros a nivel de direcciones de capa MAC o direcciones IP locales.

Para el usuario, el servidor *proxy* es invisible; todos los pedidos y respuestas parecen ser directamente atendidas por el servidor direccionado en Internet. El servidor *proxy*, en realidad no es invisible, su dirección IP tiene que ser especificada en una opción de configuración del *browser* u otro programa de protocolo.

Una ventaja del servidor *proxy* es que su *cache* puede servir para todos los usuarios. Si uno o más sitios de Internet son frecuentemente visitados, éstos probablemente estarán en el *cache* del *proxy*, lo cual mejorará el tiempo de respuesta del usuario.

A un ISP le conviene asignar un conjunto limitado de direcciones IP válidas a usuarios grandes (Ejm. Intranets de grandes negocios) para evitar el agotamiento de sus direcciones IP. El usuario podrá a través de un servidor *proxy* conectar toda la Intranet a Internet. Internamente la Intranet asignará a sus *hosts* direcciones IP no válidas (direcciones reservadas para Intranet) y utilizará direcciones válidas únicamente en el servidor *proxy* y otros servidores que desee poner en Internet. El servidor *proxy* como se ha descrito se encargará de realizar transparentemente la transformación de conjuntos de direcciones IP no válidas en una o varias direcciones IP válidas asignadas al servidor *proxy* y viceversa.

Las funciones de *proxy*, *firewall* y *cache* pueden estar en programas servidores separados o combinadas en un mismo paquete. Diferentes programas servidores pueden

estar en diferentes computadores. Por ejemplo, un servidor *proxy*, puede estar en la misma máquina que el *firewall* o puede estar en una máquina servidor separado y enviar los pedidos a través del *firewall*.

REFERENCIAS (Capítulo III)

[1] http://www.redbooks.ibm.com, "*The Technical Side of Being an Internet Service Provider*", IBM