CAPITULO II

Marco Teórico

2. CAPITULO II – MARCO TEÓRICO.

2.1. ANTECEDENTES DE LA INVESTIGACIÓN.

Al presente trabajo de investigación, no le antecede proyecto de tesis similar, luego de buscar las referencias de tesis en la Biblioteca de la Facultad de Administración, Finanzas e Informática y en todas las Bibliotecas de la Universidad Técnica de Babahoyo, naciendo esta idea de un análisis y observación realizada a los alumnos y docentes que acuden a este departamento.

Previamente se observo que en la Biblioteca Virtual de la Facultad de Administración, Finanzas e Informática no cuenta con un buen sistema de gestión y equipos de control para los estudiantes y docentes que utilizan el internet dentro de este departamento, además no se encuentra configurado un servidor que permita realizar todos estos procesos de control y administración.

El presente trabajo de investigación tiene como principal objetivo proponer la configuración de un servidor proxy que administre los datos que circulan en la intranet de la Biblioteca Virtual de la Facultad de Administración, Finanzas e Informática hacia internet, con la ayuda de una aplicación web que permita realizar esta configuración de una forma fácil y rápida.

Bajo este proceso de investigación, el trabajo que se plantea pudo dar como resultado la optimización del ancho de banda que dispone la Biblioteca Virtual de la Facultad de Administración, Finanzas e Informática con estándares de calidad QoS, para que la navegación en internet sea más segura y para los fines con que cuenta dicho departamento de una conexión a internet.

2.2. INTRODUCCIÓN.

El acelerado desarrollo de nuevas tecnologías ha servido de plataforma para la aparición de nuevas formas de comunicación, Internet ha sido y sigue siendo una autopista sin fronteras por donde usuarios, sin importar su ubicación geográfica, pueden tener acceso a una ilimitada fuente de información. Es incontable la gran cantidad de operaciones que viajan a través de la gran telaraña mundial y las ventajas que estas han traído consigo, las organizaciones han crecido y han estrechado sus lazos a través de este medio.

Pero a medida que crecen las comunicaciones en internet cobra mayor importancia para toda organización que las transacciones sean precisas y seguras tanto dentro como fuera de sus redes. El personal de tecnología de la información, tiene la gran responsabilidad de velar por el buen uso tanto del ancho de banda, como del tránsito que circula en la red, pero en muchas ocasiones se escapan detalles referentes al verdadero uso por parte de los usuarios de Internet. Las aplicaciones de mensajería instantánea se han convertido en programas de uso masivo, al igual que las redes sociales, y los innumerables sitios de ocio disponible en la red de redes. Sin embargo, esto no se considera un problema, hasta el momento en que este comportamiento es emulado dentro de las organizaciones, donde el uso incorrecto del internet como recurso se convierte en un mal que atenta contra los objetivos y metas definidas por la organización.

Proporcionar mecanismos seguros que restrinjan este tipo de accesos y limiten el uso de Internet a solo material pertinente, puede ser una tarea costosa. Sin embargo, existen herramientas en el mercado que permiten administrar los recursos de la red de forma organizada y centralizada. Algunas de estas herramientas son de software propietario, y otras tantas son software libre. La diferencia entre ellas, es el costo de la licencia, que en el caso del software propietario tiende a ser elevado, en contraposición a las de software libre donde no existe costo por licenciamiento del producto.

Dentro de las herramientas que permiten la administración y el control del recurso red, encontramos a Squid Proxy, que es una aplicación que tiene como objetivo actuar como mediador entre un cliente y un servidor, recibiendo peticiones web de un usuario, procesándolas y brindándole respuestas a sus solicitudes. Esto permite que se aligere el tráfico en la red, y disminuya la carga de trabajo de los servidores destino, aumentando la velocidad de respuesta. El servidor Proxy crea una caché que evita transferencias idénticas de la información entre servidores durante un tiempo preestablecido por el usuario administrador. Adicionalmente el Squid Proxy permite definir reglas para restringir el contenido disponible para los usuarios de la red interna, por tanto es posible disminuir el uso indebido de los recursos, aumentando el ancho de banda disponible y disminuyendo los elementos distractores.

El Squid Proxy es una herramienta de gran importancia dentro de las organizaciones, y su uso extensivo a nivel mundial lo ha convertido en una herramienta indispensable de todo departamento de tecnología de la información, sin embargo, este software carece de una interfaz de control, administración y monitoreo, que se ajuste a las necesidades propias de usuarios cada vez más exigentes.

La carencia de un mecanismo de configuración amigable, y la falta de un generador de reportes adecuado, hace que el Squid pierda valor frente al software privativo equivalente. Ante esta situación, se desarrollo una herramienta bajo software libre que permite la configuración, administración y el control de la herramienta Squid Proxy, con una interfaz amigable a usuarios de Tecnología de la Información, disponible a la hora de configurar y administrar la herramienta y que proporciona de manera inmediata estadísticas que revelen el integro uso de las demandas Web, orientado a todo tipo de institución educativa.

2.3. FUNDAMENTACIÓN TEÓRICA.

2.3.1. BIBLIOTECA VIRTUAL F.A.F.I.

2.3.1.1. Historia.

La biblioteca de la Facultad de Administración, Finanzas e Informática, nace en el local de la Av. Universitaria Km 1,5 vía Montalvo, y responde a una inquietud, para contar con material bibliográfico que los estudiantes y docentes, que están siguiendo sus carreras y dictando sus cátedras, tengan material de consulta y referencia sobre las cátedras que están cursando.

La biblioteca es una dependencia más de la Facultad de Administración, Finanzas e Informática (F.A.F.I.), que sin descuidar su calidad contribuye a la formación académica y humana de los estudiantes de la entidad y público en general, para ello vela por la actualización y conservación del fondo bibliográfico y documental a su cargo.

2.3.1.2. Objetivos.

Contribuir a la calidad del estudiante y docentes de la F.A.F.I., mediante el acebo documental disponible y actualizado, fruto del nexo entre las fuentes y la universidad.

Difundir su material bibliográfico entre otros usuarios como: investigadores, estudiantes en general y público que necesite hacer consultas.

2.3.1.3. Misión.

Crear la experiencia más satisfactoria en la búsqueda de la información para contribuir al crecimiento y formación de nuestros alumnos, docentes y ciudadanía en general, en la parte académica, profesional y personal.

2.3.1.4. Visión.

Ser una dependencia académica de la Facultad de Administración, Finanzas e Informática con los mejores estándares de servicio, agilidad y calidad. Para proveer

la mejor información posible a la comunidad universitaria y ciudadana, cubriendo las necesidades y expectativas.

2.3.1.5. Políticas.

La biblioteca "F.A.F.I." es una instancia Académica que busca por todos los medios compartir nuestra información con los usuarios inquietos por el conocimiento, el saber y la investigación. Nuestra base documental, está compuesta por volúmenes en todas las áreas del saber, contamos con Tesis e Investigaciones generadas en nuestros espacios académicos.

2.3.1.6. Servicios.

- **Biblioteca Abierta:** Ponemos a disposición y al alcance del usuario, toda la colección que dispone nuestra biblioteca, para que se realice una consulta directa. Nos empeñamos en depurar continuamente la base de datos.

- **Área de Reserva:** Para documentos de uso interno. Material de uso restringido o de soporte variado.

- **Hemeroteca:** Buscamos mejorar las investigaciones y consultas en general a través de información actualizada sobre todo en temas nacionales, internacionales y profesionales de nuestras áreas de oferta educativa.

- **Cd teca y DVD teca:** Queremos hacer más atractivos el aprendizaje, a partir de medios electrónicos. Contamos con equipos y tecnologías adecuadas para que los aprendizajes sean más amenos. Contamos con el espacio adecuado para la utilización individual y en grupo, de esta información. Queremos transformar a CD's otra información.

- **Mapoteca:** La información actual se la tiene en revistas; por esto es muy importante contar con bases internacionales.

- **Videoteca:** Contamos con un espacio adecuado para el almacenaje de este material. Ofrecemos un área acondicionada para consultar este material.

- **Servicios de autocopia y fotocopia:** Evitar mutilaciones de libros o atrasos en entrega de documentos. A partir de un buen manejo de los derechos de autor, facilitamos las copias de pequeñas partes de documentos.

- **Promoción y comunicación:** Consideramos muy importante abrir espacios de comunicación con los usuarios internos y externos. Ofrecemos guías de los recursos de Información, Trípticos, campañas de lectura y otros eventos fundamentales en el desempeño de la biblioteca.[1]

2.3.2. SISTEMAS DE INFORMACIÓN PARA EL CONTROL DE DATOS E INFORMACIÓN.

2.3.2.1. Introducción a los Sistemas de Información.

Un sistema de información es un conjunto de procedimientos ordenados que, al ser ejecutados, proporcionan información para apoyar la toma de decisiones y el control de la Institución. La información se define como una entidad tangible o intangible que permite reducir la incertidumbre acerca de algún estado o suceso.

Los sistemas de información administrativa están volviéndose indispensables, a gran velocidad, para la planificación, la toma de decisiones y el control. La velocidad y exactitud con que los directivos pueden recibir información sobre lo que está funcionando bien o lo que está funcionando mal determinarán, en gran medida, la eficacia que tendrán los sistemas de control.

Las Tecnologías de la Información han sido conceptualizadas como la integración y convergencia de la computación, las telecomunicaciones y la técnica

[1]Reglamentos de la Biblioteca Virtual F.A.F.I., Universidad Técnica de Babahoyo, Pág. 42 a 46.

para el procesamiento de datos, donde sus principales componentes son: el factor humano, los contenidos de la información, el equipamiento, la infraestructura, el software y los mecanismos de intercambio de información, los elementos de política y regulaciones, además de los recursos financieros.

Los componentes anteriores conforman los protagonistas del desarrollo informático en una sociedad, tanto para su desarrollo como para su aplicación, además se reconoce que las tecnologías de la información constituyen el núcleo central de una transformación multidimensional que experimenta la economía y la sociedad; de aquí lo importante que es el estudio y dominio de las influencias que tal transformación impone al ser humano como ente social, ya que tiende a modificar no sólo sus hábitos y patrones de conducta, sino, incluso, su forma de pensar.

2.3.2.2. Sistemas de Información.

El término sistemas de información hace referencia a un concepto genérico que tiene diferentes significados según el campo del conocimiento al que se aplique dicho concepto, a continuación se enumeran algunos de dichos campos y el sentido concreto que un Sistema de Información tiene en ese campo.

En informática, un sistema de información es cualquier sistema computacional que se utilice para obtener, almacenar, manipular, administrar, controlar, procesar, transmitir o recibir datos, para satisfacer una necesidad de información.

En teoría de sistemas, un sistema de información es un sistema, automatizado o manual, que abarca personas, máquinas, y/o métodos organizados de recolección de datos, procesamiento, transmisión y diseminación de datos que representa información para el usuario.

Los sistemas de información tratan el desarrollo, uso y administración de la infraestructura de la tecnología de la información en una organización.

Un Sistema de Información realiza cuatro actividades básicas:

Entrada de Información: Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfaces automáticas. Las unidades típicas de entrada de datos a las computadoras son las terminales, las cintas magnéticas, las unidades de diskette, los códigos de barras, los escáneres, la voz, el teclado y el mouse, entre otras.

Almacenamiento de información: El almacenamiento es una de las actividades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. La unidad típica de almacenamiento son los discos duros, los discos flexibles o diskettes y los discos compactos.

Procesamiento de Información: Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones.

Salida de Información: La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, diskettes, cintas

magnéticas, la voz y los plotters, entre otros. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo. En este caso, también existe una interface automática de salida.

2.3.2.3. Los sistemas de información y su importancia.

Los Sistemas de Información y las Tecnologías de Información han cambiado la forma en que operan las organizaciones actuales. A través de su uso se logran importantes mejoras, pues automatizan los procesos operativos, suministran una plataforma de información necesaria para la toma de decisiones y, lo más importante, su implantación logra ventajas competitivas o reducir la ventaja de los rivales.

Por otro lado es importante tener una comprensión básica de los sistemas de información para entender cualquier otra área funcional en la organización, por eso es importante también, tener una cultura informática en nuestras organizaciones que permitan y den las condiciones necesarias para que los sistemas de información logren los objetivos citados anteriormente. Muchas veces las organizaciones no han entrado en la etapa de cambio hacía la era de la información sin saber que es un riesgo muy grande de fracaso debido a las amenazas del mercado y su incapacidad de competir.

2.3.2.4. Tipos y usos de los sistemas de información.

Durante los próximos años, los Sistemas de Información cumplirán tres objetivos básicos dentro de las organizaciones:

1. Automatización de procesos operativos.
2. Proporcionar información que sirva de apoyo al proceso de toma de decisiones.
3. Lograr ventajas competitivas a través de su implantación y uso.

Los Sistemas de Información que logran la automatización de procesos operativos dentro de una organización, son llamados frecuentemente Sistemas Transaccionales, ya que su función primordial consiste en procesar transacciones tales como pagos, cobros, pólizas, entradas, salidas, etc. Por otra parte, los Sistemas de Información que apoyan el proceso de toma de decisiones son los Sistemas de Soporte a la Toma de Decisiones, Sistemas para la Toma de Decisión de Grupo, Sistemas Expertos de Soporte a la Toma de Decisiones y Sistema de Información para Ejecutivos. El tercer tipo de sistema, de acuerdo con su uso u objetivos que cumplen, es los Sistemas Estratégicos, los cuales se desarrollan en las organizaciones con el fin de lograr ventajas competitivas, a través del uso de la tecnología de información.

A continuación se mencionan las principales características de estos tipos de Sistemas de Información.

Sistemas Transaccionales. Sus principales características son:

- Ahorros significativos de mano de obra, debido a que automatizan tareas operativas de la organización.
- Con frecuencia son el primer tipo de Sistemas de Información que se implanta en las organizaciones.
- Son intensivos en entrada y salid de información; sus cálculos y procesos suelen ser simples y poco sofisticados.
- Tienen la propiedad de ser recolectores de información.
- Son fáciles de justificar ante la dirección general, ya que sus beneficios son visibles y palpables.

Sistemas de Apoyo de las Decisiones. Las principales características de estos son:

- Suelen introducirse después de haber implantado los Sistemas Transaccionales más relevantes de la empresa, ya que estos últimos constituyen su plataforma de información.

- La información que generan sirve de apoyo a los mandos intermedios y a la alta administración en el proceso de toma de decisiones.
- Suelen ser intensivos en cálculos y escasos en entradas y salidas de información.
- No suelen ahorrar mano de obra. Debido a ello, la justificación económica para el desarrollo de estos sistemas es difícil.
- Suelen ser Sistemas de Información interactivos y amigables, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.
- Apoyan la toma de decisiones que, por su misma naturaleza son repetitivos y de decisiones no estructuradas que no suelen repetirse.
- Estos sistemas pueden ser desarrollados directamente por el usuario final sin la participación operativa de los analistas y programadores del área de informática.

Sistemas Estratégicos. Sus principales características son:

- Su función primordial no es apoyar la automatización de procesos operativos ni proporcionar información para apoyar la toma de decisiones.
- Suelen desarrollarse in house, es decir, dentro de la organización, por lo tanto no pueden adaptarse fácilmente a paquetes disponibles en el mercado.
- Típicamente su forma de desarrollo es a base de incrementos y a través de su evolución dentro de la organización. Se inicia con un proceso o función en particular y a partir de ahí se van agregando nuevas funciones o procesos.
- Su función es lograr ventajas que los competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores.
- Apoyan el proceso de innovación de productos y proceso dentro de la empresa debido a que buscan ventajas respecto a los competidores y una forma de hacerlo en innovando o creando productos y procesos.

Por último, es importante aclarar que algunos autores consideran un cuarto tipo de sistemas de información denominado Sistemas Personales de Información, el cual está enfocado a incrementar la productividad de sus usuarios.

2.3.2.5. Ventajas de utilizar sistemas de información.

- Automatizan los procesos operativos.
- Suministran plataforma para la toma de decisiones.
- Logra ventaja competitiva.
- Reducción de ventaja de los rivales.[2]

2.3.2.6. Sistema Informático.

Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, hardware, software y de recurso humano que permite almacenar y procesar información. El hardware incluye computadoras o cualquier tipo de dispositivo electrónico inteligente, que consisten en procesadores, memoria, sistemas de almacenamiento externo, etc. El software incluye al sistema operativo, firmware y aplicaciones, siendo especialmente importante los sistemas de gestión de bases de datos. Por último el soporte humano incluye al personal técnico que crean y mantienen el sistema y a los usuarios que lo utilizan.

Un sistema informático puede formar parte de un sistema de información; en este último la información, uso y acceso a la misma, no necesariamente está informatizada. Por ejemplo, el sistema de archivo de libros de una biblioteca y su actividad en general es un sistema de información. Si dentro del sistema de información hay computadoras que ayudan en la tarea de organizar la biblioteca, entonces ese es un sistema informático.

2.3.2.7. Diferencias entre un sistema informático y un sistema de información.

Un sistema informático utiliza computadoras para almacenar, procesar y acceder a la información.

[2]VÁZQUEZ Roberto, "Sistemas de Información y sus ventajas competitivas"
http://www.punksolid.com/sistemas-de-informacion-y-sus-ventajas-competitivas-diapositivas/2008/

En un sistema de información se pueden utilizar computadoras (de hecho en casi siempre se utilizan), pero no es necesario. Puede accederse a la información utilizando un método mecánico o físico. Por ejemplo, buscar un expediente en un archivador.

Ambos sistemas tienen como objetivo hacer más simple la gestión y procesamiento de la información.

2.3.3. LAS REDES DE INFORMACIÓN.

Se puede definir una red informática como un sistema de comunicación que conecta computadoras y otros periféricos informáticos entre sí, con la finalidad de compartir información y recursos.

Al compartir información y recursos en una red, los usuarios de los sistemas informáticos de una organización podrán hacer un mejor uso de los mismos, mejorando de este modo el rendimiento global de la organización.

Las redes informáticas ofrecen muchos beneficios como:

- Mayor facilidad en la comunicación entre usuarios.
- Reducción en el presupuesto para software y hardware.
- Posibilidad de organizar grupos de trabajo.
- Mejoras en la administración de los equipos y programas.
- Mejoras en la integridad de los datos.
- Mayor seguridad para acceder a la información.

2.3.3.1. Tipos de Redes de Información.

Las redes de información se pueden clasificar según su extensión y su topología. Una red puede empezar siendo pequeña para crecer junto con la

organización o institución. A continuación se presenta los distintos tipos de redes disponibles:

Por su extensión de acuerdo con la distribución geográfica:

Segmento de red (subred).- Un segmento de red suele ser definido por el "hardware" o una dirección de red específica.

Red de área local (LAN).- Una LAN es un segmento de red que tiene conectadas estaciones de trabajo y servidores o un conjunto de segmentos de red interconectados, generalmente dentro de la misma zona.

Red de campus.- Una red de campus se extiende a otros edificios dentro de un campus o área industrial. Los diversos segmentos o LAN de cada edificio suelen conectarse mediante cables de la red de soporte.

Red de área metropolitana (MAN).- Una red MAN es una red que se expande por pueblos o ciudades y se interconecta mediante diversas instalaciones públicas o privadas.

Red de área extensa (WAN y redes globales).- Las WAN y redes globales se extienden sobrepasando las fronteras de las ciudades, pueblos o naciones. Los enlaces se realizan con instalaciones de telecomunicaciones públicas y privadas, además por microondas y satélites.

Por su Topología o forma lógica de la red:

Anillo.- Las estaciones están unidas unas con otras formando un círculo por medio de un cable común. El último nodo de la cadena se conecta al primero cerrando el anillo. Las señales circulan en un solo sentido alrededor del círculo, regenerándose en cada nodo. Con esta metodología, cada nodo examina la

información que es enviada a través del anillo. Si la información no está dirigida al nodo que la examina, la pasa al siguiente en el anillo. La desventaja del anillo es que si se rompe una conexión, se cae la red completa.

Estrella.- La red se une en un único punto, normalmente con un panel de control centralizado, como un concentrador de cableado. Los bloques de información son dirigidos a través del panel de control central hacia sus destinos. Este esquema tiene una ventaja al tener un panel de control que monitorea el tráfico y evita las colisiones y una conexión interrumpida no afecta al resto de la red.

Bus.- Las estaciones están conectadas por un único segmento de cable. A diferencia del anillo, el bus es pasivo, no se produce regeneración de las señales en cada nodo. Los nodos en una red de "bus" transmiten la información y esperan que ésta no vaya a chocar con otra información transmitida por otro de los nodos. Si esto ocurre, cada nodo espera una pequeña cantidad de tiempo al azar, después intenta retransmitir la información.

Híbridas.- El bus lineal, la estrella y el anillo se combinan algunas veces para formar combinaciones de redes híbridas.

2.3.3.2. Dirección IP.

Una dirección IP es una etiqueta numérica que identifica, de manera lógica y jerárquica, a un interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP, que corresponde al nivel de red del protocolo TCP/IP. Dicho número no se ha de confundir con la dirección MAC que es un identificador de 48 bits para identificar de forma única a la tarjeta de red y no depende del protocolo de conexión utilizado ni de la red. La dirección IP puede cambiar muy a menudo por cambios en la red o porque el dispositivo encargado dentro de la red de asignar las direcciones IP, decida asignar

otra IP (por ejemplo, con el protocolo DHCP), a esta forma de asignación de dirección IP se denomina dirección IP dinámica.

Los sitios de Internet que por su naturaleza necesitan estar permanentemente conectados, generalmente tienen una dirección IP fija (comúnmente, IP fija o IP estática), esta, no cambia con el tiempo. Los servidores de correo, DNS, FTP públicos y servidores de páginas web necesariamente deben contar con una dirección IP fija o estática, ya que de esta forma se permite su localización en la red.

Las computadoras se conectan entre sí mediante sus respectivas direcciones IP. Sin embargo, a los seres humanos nos es más cómodo utilizar otra notación más fácil de recordar, como los nombres de dominio; la traducción entre unos y otros se resuelve mediante los servidores de nombres de dominio DNS, que a su vez, facilita el trabajo en caso de cambio de dirección IP, ya que basta con actualizar la información en el servidor DNS y el resto de las personas no se enterarán ya que seguirán accediendo por el nombre de dominio.

2.3.3.2.1. Direcciones IPv4.

Las direcciones IPv4 se expresan por un número binario de 32 bits permitiendo un espacio de direcciones de hasta 4.294.967.296 (232) direcciones posibles. Las direcciones IP se pueden expresar como números de notación decimal: se dividen los 32 bits de la dirección en cuatro octetos. El valor decimal de cada octeto está comprendido en el rango de 0 a 255. En la expresión de direcciones IPv4 en decimal se separa cada octeto por un carácter único ".". Cada uno de estos octetos puede estar comprendido entre 0 y 255, salvo algunas excepciones. Los ceros iníciales, si los hubiera, se pueden obviar.

2.3.3.2.2. Direcciones privadas.

Hay ciertas direcciones en cada clase de dirección IP que no están asignadas y que se denominan direcciones privadas. Las direcciones privadas pueden ser

utilizadas por los hosts que usan traducción de dirección de red (NAT) para conectarse a una red pública o por los hosts que no se conectan a Internet. En una misma red no pueden existir dos direcciones iguales, pero sí se pueden repetir en dos redes privadas que no tengan conexión entre sí o que se conecten mediante el protocolo NAT. Las direcciones privadas son:

- Clase A: 10.0.0.0 a 10.255.255.255 (8 bits red, 24 bits hosts).
- Clase B: 172.16.0.0 a 172.31.255.255 (16 bits red, 16 bits hosts). 16 redes clase B contiguas.
- Clase C: 192.168.0.0 a 192.168.255.255 (24 bits red, 8 bits hosts). 256 redes clase C contiguas.

2.3.3.2.3. Máscara de red.

La máscara de red es una combinación de bits que sirve para delimitar el ámbito de una red de computadoras. Su función es indicar a los dispositivos qué parte de la dirección IP es el número de la red, incluyendo la subred, y qué parte es la correspondiente al host.

Se utiliza para saber si una dirección IP pertenece a una subred. Con un conjunto de 32 bits consecutivos para determinar la parte de red de una dirección y el resto de los 32 bits todos a cero para determinar la parte que identifica a host.

La máscara también es lo más habitual encontrarla en formato decimal lo cual sería 255.255.255.0. Se utilizan valores predeterminados para varios tipos de red:

255.0.0.0	/8	Para redes de clase A
255.255.0.0	/16	Para redes de clase B
255.255.255.0	/24	Para redes de clase C

Tabla Nº 2.1: Mascara de subredes predeterminadas.

Aunque estos son los valores predeterminados en algunos casos se pueden definir mascaras personalizadas para la creación de las subredes. Para ello modificamos la parte del host y modificamos a si el número de redes y host disponibles.

2.3.3.2.4. IP dinámica.

Una dirección IP dinámica es una IP asignada mediante un servidor DHCP (Dynamic Host Configuration Protocol) al usuario. La IP que se obtiene tiene una duración máxima determinada. El servidor DHCP provee parámetros de configuración específicos para cada cliente que desee participar en la red IP. Entre estos parámetros se encuentra la dirección IP del cliente.

Ventajas

- Reduce los costos de operación a los proveedores de servicios de Internet (ISP).
- Reduce la cantidad de IP asignadas (de forma fija) inactivas.

Desventajas

- Obliga a depender de servicios que redirigen un host a una IP.
- Asignación de direcciones IP.
- Dependiendo de la implementación concreta, el servidor DHCP tiene tres métodos para asignar las direcciones IP:
    - manualmente, cuando el servidor tiene a su disposición una tabla que empareja direcciones MAC con direcciones IP, creada manualmente por el administrador de la red. Sólo clientes con una dirección MAC válida recibirán una dirección IP del servidor.
    - automáticamente, donde el servidor DHCP asigna permanentemente una dirección IP libre, tomada de un rango prefijado por el administrador.
    - dinámicamente, el único método que permite la reutilización de direcciones IP. El administrador de la red asigna un rango de direcciones IP para el DHCP y cada computadora cliente de la LAN tiene su software

de comunicación TCP/IP configurado para solicitar una dirección IP del servidor DHCP cuando su tarjeta de interfaz de red se inicie. El proceso es transparente para el usuario y tiene un periodo de validez limitado.

2.3.3.2.5. IP fija.

Una dirección IP fija es una dirección IP asignada por el usuario de manera manual, o por el servidor de la red con base en la dirección MAC del cliente. Mucha gente confunde IP fija con IP pública e IP dinámica con IP privada.

Una IP puede ser privada ya sea dinámica o fija como puede ser IP pública dinámica o fija. Una IP pública se utiliza generalmente para montar servidores en internet y necesariamente se desea que la IP no cambie por eso siempre la IP pública se la configura de manera fija y no dinámica.

En el caso de la IP privada generalmente es dinámica asignada por un servidor DHCP, pero en algunos casos se configura IP privada fija para poder controlar el acceso a internet o a la red local, otorgando ciertos privilegios dependiendo del número de IP que tenemos, si esta cambiara (fuera dinámica) sería más complicado controlar estos privilegios.

Las IP públicas fijas actualmente en el mercado de acceso a Internet tienen un costo adicional mensual. Estas IP son asignadas por el usuario después de haber recibido la información del proveedor o bien asignadas por el proveedor en el momento de la primera conexión.

Esto permite al usuario montar servidores web, correo, FTP, etc. y dirigir un nombre de dominio a esta IP sin tener que mantener actualizado el servidor DNS cada vez que cambie la IP como ocurre con las IP públicas dinámicas.

2.3.3.3. Modelo Cliente – Servidor.

Este es un modelo de proceso en el que las tareas se reparten entre programas que se ejecutan en el servidor y otros en la estación de trabajo del usuario. En una red cualquier equipo puede ser el servidor o el cliente. El cliente es la entidad que solicita la realización de una tarea, el servidor es quien la realiza en nombre del cliente. Este es el caso de aplicaciones de acceso a bases de datos, en las cuales las estaciones ejecutan las tareas del interfaz de usuario (pantallas de entrada de datos o consultas, listados, entre otros) y el servidor realiza las actualizaciones y recuperaciones de datos en la base.

Aunque clientes y servidores suelen verse como máquinas separadas, pueden, de hecho, ser dos áreas separadas en la misma máquina. Por tanto, una única máquina Unix puede ser al mismo tiempo cliente y servidor.

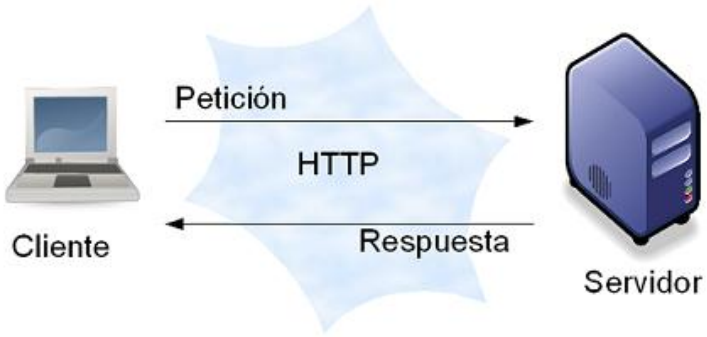

Grafico Nº 2.1: Modelo Cliente – Servidor.

Las acciones que debe llevar a cabo el servidor son las siguientes:

- Abrir el canal de comunicaciones e informar a la red tanto de la dirección a la que responderá como de su disposición para aceptar peticiones de servicio.
- Esperar a que un cliente le pida servicio en la dirección que él tiene declarada.

- Cuando recibe una petición de servicio, crea un proceso *fork* para que le de servicio al cliente.
- Se regresa al punto número 2 para esperar nuevas peticiones de servicio.

El cliente, por su parte, lleva a cabo las siguientes acciones:

- Abrir el canal de comunicaciones y conectarse a la dirección de red atendida por el servidor. Esta dirección de red debe ser conocida por el cliente y debe responder al esquema de generación de direcciones de la familia de *sockets* que se esté empleando.
- Enviar al servidor un mensaje de petición de servicio y esperar hasta recibir la respuesta.

2.3.3.4. Protocolos de Red.

Podemos definir un protocolo como el conjunto de normas que regulan la comunicación (establecimiento, mantenimiento y cancelación) entre los distintos componentes de una red informática.

Los detalles precisos de lo que hacen los protocolos dependen del tipo de protocolo y de las tareas que les estemos pidiendo a la computadora, pero las funciones generales que cumplen aquellos en nuestra red son comunes:

- Enviar y recibir mensajes de cualquier tipo a través del hardware de la red.
- Identificar quien envía y cuál es el destino del mensaje, y determinar si la computadora que recibe es el destino final.
- Para las computadoras con múltiples conexiones de red, enviar si son posibles los mensajes recibidos a lo largo del camino hacia su destino final.

- Verificar que el mensaje recibido ha llegado intacto o solicitar la retransmisión de mensajes dañados.
- Descubrir las computadoras que están operando en la red de área local.
- Convertir los nombres de las computadoras en direcciones usadas por el software y hardware de la red y viceversa.
- Publicitar los servicios ofrecidos por esta computadora y solicitar cuales son los servicios ofrecidos por las otras computadoras.
- Recibir la identificación del usuario y la información de autenticación, y el control de acceso a los servicios.
- Codificar y decodificar la información transmitida para mantener la seguridad a través de una red poco segura.
- Transferir información en ambos sentidos de acuerdo a los requerimientos del software y servicios específicos.

2.3.3.4.1. Protocolo TCP/IP.

TCP/IP es el protocolo común utilizado por todos los computadores conectados a Internet, de manera que éstos puedan comunicarse entre sí. Hay que tener en cuenta que en Internet se encuentran conectados computadores de clases muy diferentes y con hardware y software incompatibles en muchos casos, además de todos los medios y formas posibles de conexión. Aquí se encuentra una de las grandes ventajas del TCP/IP, pues este protocolo se encargará de que la comunicación entre todos sea posible. TCP/IP es compatible con cualquier sistema operativo y con cualquier tipo de hardware.

2.3.3.4.2. Protocolo HTTP.

El propósito del protocolo HTTP es permitir la transferencia de archivos (principalmente, en formato HTML) entre un navegador (el cliente) y un servidor web (denominado, entre otros, httpd en equipos UNIX) localizado mediante una cadena de caracteres denominada dirección URL.

2.3.3.4.3. Protocolo P2P.

A grandes rasgos, una red informática entre iguales (en inglés, peer-to-peer - que se traduciría de par a par- o de punto a punto, y más conocida como P2P) se refiere a una red que no tiene clientes ni servidores fijos, sino una serie de nodos que se comportan simultáneamente como clientes y como servidores respecto de los demás nodos de la red. Es una forma legal de compartir archivos de forma similar a como se hace en el email o mensajeros instantáneos, sólo que de una forma más eficiente.

2.3.3.4.4. Protocolo IRC.

IRC (Internet Relay Chat) es un protocolo de comunicación en tiempo real basado en texto, que permite debates en grupo o entre dos personas y que está clasificado dentro de los servicios de comunicación en tiempo real. Se diferencia de la mensajería instantánea en que los usuarios no deben acceder a establecer la comunicación de antemano, de tal forma que todos los usuarios que se encuentran en un canal pueden comunicarse entre sí, aunque no hayan tenido ningún contacto anterior.

2.3.4. INTERNET LA RED DE REDES.

El Internet algunas veces llamado simplemente "La Red", es un sistema mundial de redes de computadoras, un conjunto integrado por las diferentes redes de cada país del mundo, por medio del cual un usuario en cualquier computadora puede, en caso de contar con los permisos apropiados, acceder a la información de otra computadora y poder tener inclusive comunicación directa con otros usuarios en otras computadoras.

Fue concebido por la agencia de nombre ARPA del gobierno de los Estados Unidos en el año de 1969 y se le conocía inicialmente como ARPANET. El propósito original fue crear una red que permitiera a los investigadores en un

Campus poder comunicarse a través de los sistemas de cómputo con investigadores en otras Universidades.

Hoy en día, el Internet es un medio de comunicación pública, cooperativa y autosuficiente en términos económicos, accesible a cientos de millones de personas en el mundo entero. Físicamente, el Internet usa parte del total de recursos actualmente existentes en las redes de telecomunicaciones.

2.3.4.1. La web como medio de acceso a páginas de información.

World Wide Web, o simplemente Web, es el universo de información accesible a través de Internet, una fuente inagotable del conocimiento humano. El componente más usado en el Internet es definitivamente el Web. Su característica sobresaliente es el texto remarcado, un método para referencias cruzadas instantáneas. Usando el Web, se tiene acceso a millones de páginas de información. La exploración en el Web se realiza por medio de un software especial denominado Browser o Explorador. La apariencia de un Sitio Web puede variar ligeramente dependiendo del explorador que use. Así mismo, las versiones más recientes disponen de una funcionalidad mucho mayor tal como animación, realidad virtual, sonido y música.

2.3.4.2. Las Páginas Web, los documentos electrónicos.

Una página Web es un documento electrónico que contiene información específica de un tema en particular y que es almacenado en algún sistema de cómputo que se encuentre conectado a la red mundial de información, de tal forma que este documento pueda ser consultado por cualesquier persona que se conecte a esta red mundial de comunicaciones y que cuente con los permisos apropiados para hacerlo.

Una página Web es la unidad básica del World Wide Web, misma que tiene la característica peculiar de que el texto se combina con imágenes para hacer que el documento sea dinámico y permita que se puedan ejecutar diferentes acciones, una

tras otra, a través de la selección de texto remarcado o de las imágenes, acción que nos puede conducir a otra sección dentro del documento, abrir otra página Web, iniciar un mensaje de correo electrónico o transportarnos a otro Sitio Web totalmente distinto a través de sus hipervínculos.

2.3.4.3. Teoría de portales.

Los portales nacen de la necesidad que se presenta en el nuevo modelo de la economía, la cual nos enseña una guía de bienvenida con información detallada cada día. Portal es un término el cual hace referencia a un Sitio Web el cual pretende servir como un sitio principal de partida para las personas las cuales se conectan a la World Wide Web. Los portales tienen un gran reconocimiento en el Internet por el poder de influencia que tienen sobre grandes comunidades a nivel mundial.

El objetivo principal de emplear un portal es el de localizar información, es un servicio de valor añadido que ofrece al usuario la posibilidad de personalizar al máximo su página Web, indicando qué quiere encontrar o los campos en los que esté interesado.

2.3.4.4. Tipos de Portales.

Podemos distinguir fundamentalmente dos tipos de portales:

2.3.4.4.1. Portales generales, horizontales o mega-portales:

Son aquellos cuyo contenido abarca casi todos los temas posibles de Internet. Están encaminados a un usuario estándar de Internet, su contenido recoge información de interés general como noticias de actualidad, información sobre el clima, servicios de valor añadido, etc. Es habitual además, que estos portales estén dirigidos a una comunidad específica de usuarios, generalmente de índole geográfica, un ejemplo de estos portales es el de Terra.

2.3.4.4.2. Portales especializados o temáticos:

Los portales especializados son fruto también de la verticalización de la información en Internet. Son modelos de portales los cuales tratan de proyectar los servicios genéricos de un portal horizontal en un ámbito más específico, con un criterio temático. Los portales especializados pretenden pues satisfacer las necesidades de información de una comunidad de usuarios concreta.

Según esta última afirmación podemos destacar dos tipos de portales especializados:

- Portales verticales: Un portal es un sitio Web que proporciona información y servicios a una industria en particular. Es el equivalente industrial específico de los portales generales de la WWW pero, además de ofrecer servicios típicos centran su cobertura de contenidos en un tema o sector concreto.

- Portales corporativos o institucionales: Es un sitio Web que proporciona información de la empresa o institución a la que pertenece fundamentalmente a los empleados de la misma. Normalmente este tipo de portales corporativos son una prolongación natural de las intranets corporativas en las que se ha cuidado la organización de la información a través de la WWW que suministran.

2.3.5. GPL Y SOFTWARE PROPIETARIO.

GPL.

GPL (General Public License o Licencia Publica General) es un tipo de licencia que está orientada a la libre distribución, modificación y uso de software bajo la cual trabaja el proyecto GNU. El software que se distribuye bajo esta licencia permite al usuario su total manipulación como distribución de acorde a las necesidades que este tenga.

Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

Software propietario.

Se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.

2.3.5.1. GNU/Linux.

GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux, que es usado con herramientas de sistema GNU. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL (Licencia Pública General de GNU, en inglés: General Public License) y otra serie de licencias libres.

2.3.5.1.1. Historia de GNU/Linux.

El nacimiento del sistema operativo Linux no ha sido fruto de la casualidad, sino todo lo contrario. Es el resultado de varios acontecimientos que se han sucedido en diferentes momentos a lo largo de las últimas décadas que podríamos resumir, principalmente, en los tres siguientes:

El primero de ellos se relaciona con la aparición del sistema operativo UNIX cuya gestación se inicia con los trabajos de Dennis Ritchie, durante los años 70, en los laboratorios de AT & T. En un principio estaba escrito en lenguaje ensamblador, aceptaba tan solo dos usuarios y recibió el nombre de UNICS. En 1973

se reescribió todo el código en lenguaje C, se amplió el número de usuarios y se le bautizó con el nombre de UNIX. Se distribuyó por universidades de todo el mundo. Una de éstas llegó a la Universidad de California en Berkeley la cual participó con muchas innovaciones a través de la BSD (Berkeley Software Foundation). En el año 1982 salieron al mercado las diferentes versiones AIX de IMB, XENIX de Microsoft, UNIX de BSD, etc. Unos años más tarde se homologaron todas las distribuciones bajo el mismo estándar UNIX SYSTEM V4. Su interfaz, para entonces, era solo alfanumérica (solo en modo texto). UNIX era un sistema que necesitaba de unos recursos de hardware muy potentes que estaban sólo al alcance de organizaciones militares, administrativas o académicas.

El segundo acontecimiento tuvo como punto de partida la FSF (Free Software Foundation) que con carácter no lucrativo nació en 1984. Su objetivo principal era crear un sistema operativo GNU, que se llamaría UNIX y que sería de libre distribución. Otro éxito de la FSF fue el asentamiento de las bases de un nuevo tipo de licencia para el software. Es la llamada GPL (General Public License), que permite distribuir los programas de modo gratuito siempre que éstos se acompañen con el código fuente correspondiente. Hoy en día los términos GNU y GPL son prácticamente equivalentes.

Un tercer paso decisivo se produce en 1987 a raíz de la necesidad que el profesor de sistemas operativos Andrews S. Tanenbaum tenía para explicar a sus alumnos cómo funciona por dentro un sistema operativo. Al no disponer de suficiente información sobre los sistemas de software propietarios que había, por aquellos años, optó por escribir un sistema operativo muy sencillo publicando, al mismo tiempo, todo el código fuente. Le llamó MINIX por su parecido con UNIX y su sistema de archivos "minix" todavía se emplea hoy en día debido a su elevada eficacia, sobre todo, en dispositivos de poca capacidad como disquetes o discos-ram. La idea de Tanembaum le gustó mucho a un estudiante finlandés de informática llamado Linus Torvalds quien tenía en mente crear un sistema operativo como UNIX pero que fuese capaz de adaptarse al hardware de un computador personal. Linus,

además, tuvo otra buena idea: usar la incipiente Internet para dar a conocer su proyecto, bajo licencia GPL y a todo el mundo, el 5 de octubre de 1991 desde la Universidad de Helsinki. Comienza así la andadura y el desarrollo de un sistema operativo edificado, desde el primer momento, sobre las necesidades, la creatividad y la participación de sus mismos usuarios.

Desde entonces, el crecimiento, uso y aumento de prestaciones de Linux no se ha detenido gracias al elevado número de desarrolladores, colaboradores altruistas y usuarios de todo el mundo. Se utiliza en empresas, administraciones y usuarios domésticos, ofreciendo una alternativa al software comercial de la competencia. Sin embargo, donde realmente brilla por sus cualidades es en el sector educativo.

2.3.5.1.2. Open Source.

Open Source y Software libre, son esencialmente lo mismo, la diferencia radica en que los defensores del Free Software no están ciento por ciento de acuerdo con que las empresas disfruten y distribuyan Free Software ya que, según ellos, el mercado corporativo antepone la utilidad a la libertad, a la comunidad y a los principios y por ende no va de la mano con la filosofía pura detrás del Software libre, por otra parte, los seguidores del software Open Source sostienen que el proceso normal de crecimiento de la tendencia debe llegar al mercado corporativo y no seguir escondida bajo el manto de la oposición, sino que, por el contrario, están en el deber de lanzar software potente y de excelente calidad.

Para lograrlo, creen en la necesidad de un software Open Source más confiable que el software propietario ya que son más las personas que trabajan en el al mismo tiempo y mayor la cantidad de que pueden detectar errores y corregirlos, es pues, el software que puede ser compartido abiertamente entre desarrolladores y usuarios finales de tal forma que todos aprendan de todos.

2.3.5.2. CentOS como sistema operativo en la implementación de servidores.

CentOS (Community ENTerprise Operating System) es una bifurcación a nivel binario de la distribución Linux Red Hat Enterprise Linux RHEL, compilado por voluntarios a partir del código fuente liberado por Red Hat.[3]

Grafico Nº 2.2: Logo de CentOS.

Red Hat Enterprise Linux se compone de software libre y código abierto, pero se publica en formato binario usable (CD-ROM o DVD-ROM) solamente a suscriptores pagados. Como es requerido, Red Hat libera todo el código fuente del producto de forma pública bajo los términos de la Licencia pública general de GNU y otras licencias. Los desarrolladores de CentOS usan ese código fuente para crear un producto final que es muy similar al Red Hat Enterprise Linux y está libremente disponible para ser bajado y usado por el público, pero no es mantenido ni asistido por Red Hat. Existen otras distribuciones también derivadas de las fuentes de Red Hat. CentOS usa yum para bajar e instalar las actualizaciones, herramienta también utilizada por Fedora.

2.3.5.2.1. Requisitos del sistema.

Hardware recomendado para operar:

- Memoria RAM: 64 MB (mínimo).
- Espacio en Disco Duro: 1024 MB (mínimo) - 2 GB (recomendado).
- Procesador: Intel x86-compatible (32 bit) (Intel Pentium I/II/III/IV/Celeron/Xeon, AMD K6/K7/K8, AMD Duron, Athlon /XP/MP) y AMD64 (Athlon 64, etc) e Intel EM64T (64 bit).

[3] CentOS, http://es.wikipedia.org/wiki/CentOS

2.3.6. SERVIDORES.

Un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al computador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos.

Este uso dual puede llevar a confusión. Por ejemplo, en el caso de un servidor web, este término podría referirse a la máquina que almacena y maneja los sitios web, y en este sentido es utilizada por las compañías que ofrecen hosting u hospedaje. Alternativamente, el servidor web podría referirse al software, como el servidor de http de Apache, que funciona en la máquina y maneja la entrega de los componentes de las páginas web como respuesta a peticiones de los navegadores de los clientes.

Los archivos para cada sitio de Internet se almacenan y se ejecutan en el servidor. Hay muchos servidores en Internet y muchos tipos de servidores, pero comparten la función común de proporcionar el acceso a los archivos y servicios. Un servidor sirve información a los computadores que se conecten a él. Cuando los usuarios se conectan a un servidor pueden acceder a programas, archivos y otra información del servidor.

En la web, un servidor web es un computador que usa el protocolo http para enviar páginas web al computador de un usuario cuando el usuario las solicita. Los servidores web, servidores de correo y servidores de bases de datos son a lo que tiene acceso la mayoría de la gente al usar Internet.

Algunos servidores manejan solamente correo o solamente archivos, mientras que otros hacen más de un trabajo, ya que un mismo computador puede tener diferentes programas de servidor funcionando al mismo tiempo.

2.3.6.1. Servidor HTTP.

Servidor HTTP o Servidor Web es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se utiliza el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al computador que ejecuta el programa.

2.3.6.1.1. Funcionamiento del servidor HTTP.

El Servidor web se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente (un navegador web) y que responde a estas peticiones adecuadamente, mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error. A modo de ejemplo, al teclear www.proxy.org en nuestro navegador, éste realiza una petición HTTP al servidor de dicha dirección. El servidor responde al cliente enviando el código HTML de la página; el cliente, una vez recibido el código, lo interpreta y lo exhibe en pantalla. Como vemos con este ejemplo, el cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página; el servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

Además de la transferencia de código HTML, los Servidores web pueden entregar aplicaciones web. Éstas son porciones de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP. Hay que distinguir entre:

Aplicaciones en el lado del cliente: el cliente web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java "applets" o

Javascript, el servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas scripts).

Aplicaciones en el lado del servidor: el servidor web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Las aplicaciones de servidor muchas veces suelen ser la mejor opción para realizar aplicaciones web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste no necesita ninguna capacidad añadida, como sí ocurre en el caso de querer ejecutar aplicaciones javascript o java. Así pues, cualquier cliente dotado de un navegador web básico puede utilizar este tipo de aplicaciones.

2.3.6.2. Servidor HTTPs.

Protocolo seguro de transferencia de hipertexto más conocido por sus siglas HTTPS, es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de Hiper Texto, es decir, es la versión segura de HTTP.

Es utilizado principalmente por entidades bancarias, tiendas en línea, y cualquier tipo de servicio que requiera el envío de datos personales o contraseñas.

El sistema HTTPS utiliza un cifrado basado en SSL/TLS para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible (usuario y claves de paso normalmente) no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará imposible de descifrar. El puerto estándar para este protocolo es el 443.

2.3.6.3. Servidor Intermediario (Proxy).

El término en ingles «Proxy» tiene un significado muy general y al mismo tiempo ambiguo, aunque invariablemente se considera un sinónimo del concepto de «Intermediario». Se suele traducir, en el sentido estricto, como delegado o apoderado (el que tiene el que poder sobre otro).

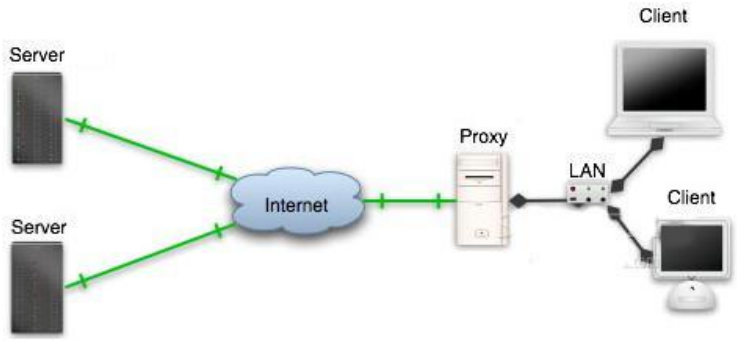

Grafico Nº 2.3: Esquema de un servidor proxy.

Un servidor proxy es un equipo intermediario situado entre el sistema del usuario e Internet, es el elemento activo del sistema que controla los tipos de paquetes de datos que entran a la red LAN, así como también cumple con la función de ser un punto de entrada a Internet, desde las estaciones de trabajo o computadores de la red. Esta situación estratégica de punto intermedio suele ser aprovechada para soportar una serie de funcionalidades: proporcionar caché, control de acceso, registro del tráfico, prohibir cierto tipo de tráfico etcétera.

Su finalidad consiste en interceptar las conexiones de red que un cliente hace a un servidor de destino, por varios motivos posibles como seguridad, rendimiento, anonimato, etc. Esta función de servidor proxy puede ser realizada por un programa o dispositivo.

Por su naturaleza de servidor el proxy cumple con las siguientes tareas:

Filtro de contenido: se pueden restringir en sus archivos de configuración, a qué tipo de contenidos pueden acceder las estaciones de trabajo.

Cache de páginas: El proxy almacena todas las páginas que se navegan desde las estaciones de trabajo, de manera que si en algún momento no hay navegación, el Proxy proveerá las páginas que se requieran desde las estaciones de trabajo, como si se estuviese navegando en Internet.

Administración del Firewall: Algunos servidores Proxy también pueden contar con lo que se denomina un Firewall o pared de fuego, que cumple la tarea de "detener" posibles intromisiones externas a la Red Interna. Este software puede filtrar algunos agentes de virus y programas dañinos que pueden hacer que la navegación no funcione en las estaciones de trabajo.

2.3.6.3.1. Características del servidor intermediario proxy.

El Servidor Proxy permitirá dar un acceso compartido y eficaz a todos los equipos de red local, permitiéndole controlar los contenidos visitados en Internet y asegurar que el uso que se hace es con fines laborales.

Asimismo podrá prevenir la entrada de virus y otro tipo de malware, evitando que sus usuarios descarguen archivos infectados.

Características principales:

- Compartir la conexión a Internet para todos los contenidos.
- Almacenamiento de las páginas visitadas acelerando las conexiones a las páginas visitadas.
- Conexiones compartidas equitativamente entre los usuarios reduciéndose así la espera.
- Ahorro de ancho de banda de Internet.
- Control de contenidos visitados.
- Establecimiento de listas negras de sitios de internet.
- Bloqueo de direcciones IP.

- Denegación de archivos no permitidos, posibles focos de infección de virus.
- Control de usuarios que pueden acceder a Internet.
- Evitar que los recursos de la institución no sean usados para fines no profesionales.
- El usar un Servidor Proxy aumenta la seguridad de nuestra red, protegiéndola contra posibles intrusiones.

2.3.6.3.2. Principio operativo de un servidor proxy.

El principio operativo básico de un servidor proxy es bastante sencillo: se trata de un servidor que actúa como "representante" de una aplicación efectuando solicitudes en Internet en su lugar. De esta manera, cuando un usuario se conecta a Internet con una aplicación del cliente configurada para utilizar un servidor proxy, la aplicación primero se conectará con el servidor proxy y le dará la solicitud. El servidor proxy se conecta entonces al servidor al que la aplicación del cliente desea conectarse y le envía la solicitud. Después, el servidor le envía la respuesta al proxy, el cual a su vez la envía a la aplicación del cliente

El proxy comprende la sintaxis de un protocolo pero no implementan ninguna de sus funcionalidades. Simplemente verifican que un mensaje proveniente de un host externo es apropiado, y luego lo envía al sistema encargado de procesar los datos.

El uso de un proxy tiene dos propósitos, mejorar el desempeño de la red: los servidores proxy pueden mejorar en gran medida el desempeño para un grupo de usuarios ya que ahorra la obtención de los resultados (consultas al servidor real) de todas las solicitudes para una cierta cantidad de tiempo; y filtrar solicitudes: de esta forma puede ofrecer un servicio de seguridad básico y muy importante para proteger una intranet o un sistema de información conectado a una red pública.

Otra ventaja de utilizar estos sistemas es que permite monitorear y controlar toda actividad de la red que involucre comunicación con el exterior (en ambas direcciones). Cuando este sistema actúa como firewall, verifica si tales solicitudes o mensajes son permitidos y las rechaza en caso de que así lo determine, en función a las reglas que se le hayan impuesto. El gateway está asociado con un router para determinar dónde son enviados los paquetes en función de tablas de ruteo e información del paquete.

2.3.6.3.3. Funcionamiento del Servidor Proxy.

El funcionamiento de un proxy cache es el siguiente: cuando un cliente realiza una petición a un servidor web, en la que genera peticiones a todos los objetos que componen dicha web, la petición llega al proxy, que revisa su cache para comprobar si dispone de los objetos que se van solicitando. En el caso de que el objeto buscado se encuentre en cache, el proxy verifica que no ha expirado: que el objeto se corresponde con el actual, y en ese caso se produce un HIT y el sistema devuelve al cliente el objeto en cuestión. Si por el contrario el objeto buscado no se encuentra en cache o la versión encontrada no está actualizada, se produce un MISS en cache, tras lo cual el proxy descarga el elemento solicitado y lo sirve al cliente.

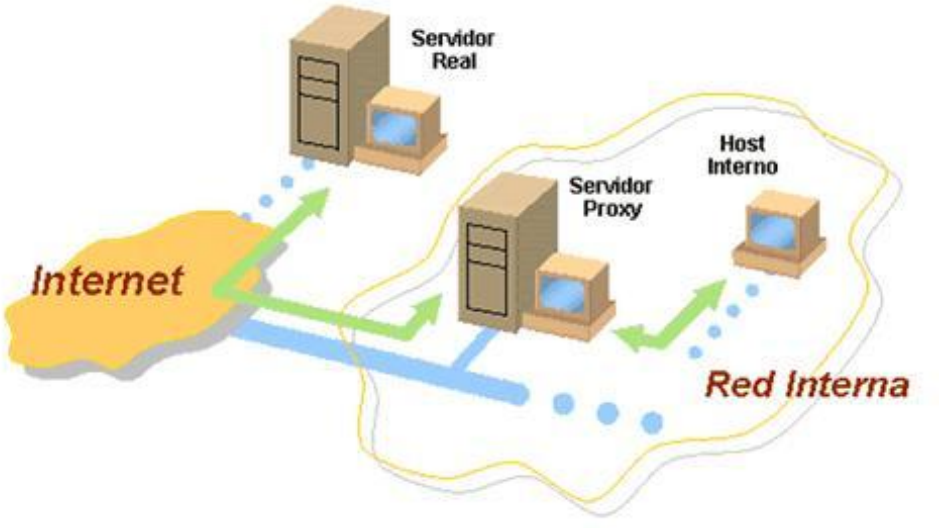

Grafico Nº 2.4: Funcionamiento básico de un Servidor Proxy.

Como puede verse, en el mejor de los casos nos ahorramos el enrutamiento desde el proxy al servidor sobre el que se realiza la petición y la transferencia de la información solicitada a través de Internet, mientras que en el peor de los casos añadimos un salto más en el enrutamiento. Por supuesto, cuánto más accedido sea un determinado contenido por los usuarios de la red interna, mayor probabilidad de que el objeto se encuentre en cache y por tanto, mayor incremento del rendimiento del sistema.

2.3.6.3.4. Esquema de conexión estándar y habitual de proxy.

Un proxy nos permite conectarnos a un equipo de forma indirecta. Cuando un equipo conectado a una red desea acceder a una información o recurso de Internet, es realmente el proxy quien realiza la comunicación y a continuación envía el resultado al equipo que solicitó dicha información.

Si en una conexión estándar o habitual tendríamos un esquema similar a este:

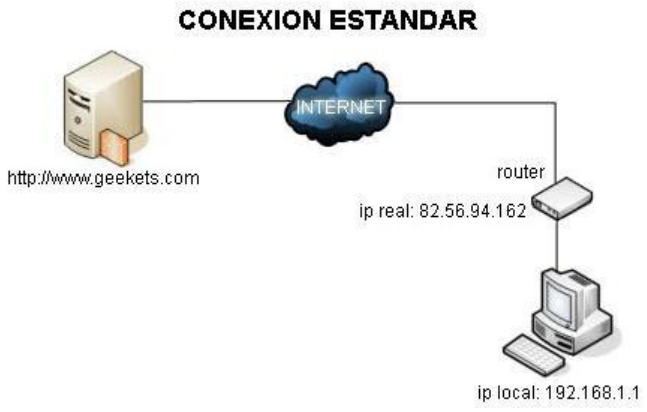

Grafico Nº 2.5: Conexión estándar a internet.

En este esquema podemos ver que al solicitar una página web alojada en un servidor, escribiendo para ello la URL en el navegador de nuestro PC, esta información sale a través de nuestro router/modem, y viaja a través de Internet hasta el servidor que contiene dicha página web. Este, gracias a que conoce nuestra IP real, nos contestará enviándonos la página solicitada.

En cambio cuando nuestra conexión se realiza a través de un proxy el esquema sería similar a este:

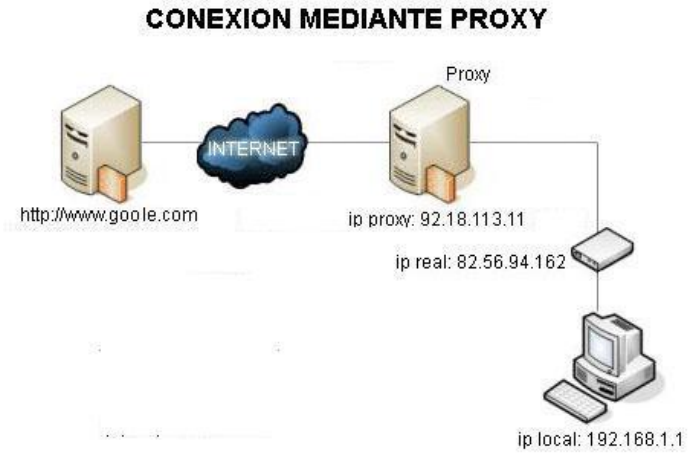

Grafico Nº 2.6: Conexión a internet utilizando proxy.

Al solicitar la misma página web, en vez de enviar directamente nuestra petición al servidor donde esta página está alojada, esta petición es enviada a través del servidor proxy. Este servidor proxy acudirá, en nuestro nombre, al servidor donde se encuentra la página (por lo que nuestra IP nunca será almacenada en este servidor) y nos la devolverá a nuestro equipo.

La gran ventaja de esto es que como hemos dicho nuestra presencia en Internet se difumina. ¿Quiere decir esto que nuestra navegación es totalmente anónima?, por desgracia no, ya que nuestras peticiones que viajan desde nuestro equipo al servidor proxy son completamente visibles por nuestro ISP, pero sin duda es un comienzo a la hora de conseguir cierto anonimato en la red.

La única forma en la que ni siquiera nuestro ISP sepa que páginas visitamos, o que archivos nos descargamos, es que la conexión se realice mediante un canal seguro, por ejemplo mediante la utilización de SSL (aunque esto sólo es utilizado en algunas páginas web como la de los bancos) o bien que se utilicen técnicas de ofuscación de protocolo.

2.3.6.3.5. Almacenamiento en caché.

La mayoría de los proxys tienen una caché, es decir, la capacidad de guardar en memoria (“en caché”) las páginas que los usuarios de la red de área local visitan comúnmente para poder proporcionarlas lo más rápido posible. De hecho, el término "caché" se utiliza con frecuencia en informática para referirse al espacio de almacenamiento temporal de datos (a veces también denominado "búfer").

Un servidor proxy con la capacidad de tener información en caché generalmente se denomina servidor "proxy-caché". Esta característica, implementada en algunos servidores proxy, se utiliza para disminuir tanto el uso de ancho de banda en Internet como el tiempo de acceso a los documentos de los usuarios.

Sin embargo, para lograr esto, el proxy debe comparar los datos que almacena en la memoria caché con los datos remotos de manera regular para garantizar que los datos en caché sean válidos.

2.3.6.3.6. Filtrado de conexiones a Internet.

Por otra parte, al utilizar un servidor proxy, las conexiones pueden rastrearse al crear registros de actividad (logs) para guardar sistemáticamente las peticiones de los usuarios cuando solicitan conexiones a Internet.

Gracias a esto, las conexiones de Internet pueden filtrarse al analizar tanto las solicitudes del cliente como las respuestas del servidor. El filtrado que se realiza comparando la solicitud del cliente con una lista de solicitudes autorizadas se denomina lista blanca; y el filtrado que se realiza con una lista de sitios prohibidos se denomina lista negra.

Finalmente, el análisis de las respuestas del servidor que cumplen con una lista de criterios (como palabras clave) se denomina filtrado de contenido.

2.3.6.3.7. Autenticación de usuarios.

Como el proxy es una herramienta intermediaria indispensable para los usuarios de una red interna que quieren acceder a recursos externos, a veces se lo puede utilizar para autenticar usuarios, es decir, pedirles que se identifiquen con un nombre de usuario y una contraseña. También es fácil otorgarles acceso a recursos externos sólo a las personas autorizadas y registrar cada uso del recurso externo en archivos de registro de los accesos identificados.

Este tipo de mecanismo, cuando se implementa, obviamente genera diversos problemas relacionados con las libertades individuales y los derechos personales.

2.3.6.3.8. Ventajas y Desventajas de los Servidores Proxy.

Ventajas:

- *Ahorro de Tráfico*: Las peticiones de páginas Web se hacen al servidor Proxy y no a Internet directamente. Por lo tanto, aligera el tráfico en la red y descarga los servidores destino, a los que llegan menos peticiones.

- *Velocidad en Tiempo **de respuesta***: El servidor Proxy crea un caché que evita transferencias idénticas de la información entre servidores durante un tiempo (configurado por el administrador) así que el usuario recibe una respuesta más rápida.

- *Demanda a Usuarios*: Puede cubrir a un gran número de usuarios, para solicitar, a través de él, los contenidos Web.

- *Filtrado de contenidos*: El servidor proxy puede hacer un filtrado de páginas o contenidos basándose en criterios de restricción establecidos por el administrador dependiendo valores y características de lo que no se permite, creando una restricción cuando sea necesario.

- *Modificación de contenidos*: Basándose en la misma función del filtrado, tiene el objetivo de proteger la privacidad en Internet, puede ser configurado para bloquear direcciones y Cookies por expresiones regulares y modifica en la petición el contenido.

Desventajas:

- Las páginas mostradas pueden no estar actualizadas si éstas han sido modificadas desde la última carga que realizó el proxy caché.

- Un diseñador de páginas web puede indicar en el contenido de su web que los navegadores no hagan una caché de sus páginas, pero este método no funciona habitualmente para un proxy.

- El hecho de acceder a Internet a través de un Proxy, en vez de mediante conexión directa, impide realizar operaciones avanzadas a través de algunos puertos o protocolos.
- Almacenar las páginas y objetos que los usuarios solicitan puede suponer una violación de la intimidad para algunas personas.

2.3.6.3.9. Aplicaciones de un servidor intermediario.

El concepto de proxy es aplicado de muy distintas formas para proporcionar funcionalidades específicas.

2.3.6.3.9.1. Proxy de web.

Se trata de un proxy para una aplicación específica el acceso a la web (principalmente los protocolos HTTP y HTTPS). Aparte de la utilidad general de un proxy a veces proporciona una caché para las páginas web y los contenidos descargados. Cuando esto sucede se dice que el proxy web está haciendo un servicio de proxy-cache. Esta caché es compartida por todos los usuario del proxy, con la

consiguiente mejora en los tiempos de acceso para consultas coincidentes. Al mismo tiempo libera la carga de los enlaces hacia Internet.

Posibles usos:

Los proxys web pueden aportar una serie de funcionalidades interesantes en distintos ámbitos:

Reducción del tráfico mediante la implementación de caché en el proxy. Las peticiones de páginas Web se hacen al servidor Proxy y no a Internet directamente. Por lo tanto se aligera el tráfico en la red y descarga los servidores destino, a los que llegan menos peticiones.

El caché utiliza normalmente un algoritmo configurable para determinar cuándo un documento está obsoleto y debe ser eliminado de la caché. Como parámetros de configuración utiliza la antigüedad, tamaño e histórico de acceso. Dos de esos algoritmos básicos son el LRU (el usado menos recientemente, en inglés "Least Recently Used") y el LFU (el usado menos frecuentemente, "Least Frequently Used").

Mejora de la velocidad en tiempo de respuesta mediante la implementación de caché en el proxy. El servidor Proxy crea un caché que evita transferencias idénticas de la información entre servidores durante un tiempo (configurado por el administrador) así que el usuario recibe una respuesta más rápida. Por ejemplo supongamos que tenemos un ISP que tiene un servidor Proxy con caché. Si un cliente de ese ISP manda una petición por ejemplo a Google esta llegará al servidor Proxy que tiene este ISP y no irá directamente a la dirección IP del dominio de Google. Esta página concreta suele ser muy solicitada por un alto porcentaje de usuarios, por lo tanto el ISP la retiene en su Proxy por un cierto tiempo y crea una respuesta en mucho menor tiempo. Cuando el usuario crea una búsqueda en Google el servidor

Proxy ya no es utilizado; el ISP envía su petición y el cliente recibe su respuesta ahora sí desde Google.

Los programas P2P se pueden aprovechar de la cache que proporcionadas por algunos proxys. Es el llamado Web caché.

El proxy puede servir para implementar funciones de filtrado de contenidos. Para ello es necesaria la configuración de una serie restricciones que indiquen lo que no se permite. Observar que esta funcionalidad puede ser aprovechada no sólo para que ciertos usuarios no accedan a ciertos contenidos sino también para filtrar ciertos ficheros que se pueden considerar como peligrosos como pueden ser virus y otros contenidos hostiles servidos por servidores web remotos.

Un proxy puede permitir esconder al servidor web la identidad del que solicita cierto contenido. El servidor web lo único que detecta es que la ip del proxy solicita cierto contenido. Sin embargo no puede determinar la ip origen de la petición. Además, si se usa una caché, puede darse el caso de que el contenido sea accedido muchas más veces que las detectadas por el servidor web que aloja ese contenido.

Inconvenientes:

Si se realiza un servicio de caché, las páginas mostradas pueden no estar actualizadas si éstas han sido modificadas desde la última carga que realizó el proxy caché.

Un diseñador de páginas web puede indicar en el contenido de su web que los navegadores no hagan una caché de sus páginas, pero este método no funciona habitualmente para un proxy.

El hecho de acceder a Internet a través de un Proxy, en vez de mediante conexión directa, dificulta (necesario configurar adecuadamente el proxy) realizar operaciones avanzadas a través de algunos puertos o protocolos.

Almacenar las páginas y objetos que los usuarios solicitan puede suponer una violación de la intimidad para algunas personas.

2.3.6.3.9.2. Web Proxy.

Su funcionamiento se basa en el de un Proxy HTTP y HTTPs, pero la diferencia fundamental es que la petición se realiza mediante una Aplicación Web servida por un servidor HTTP al que se accede mediante una URL, esto es, una página web que permite estos servicios.

2.3.6.3.9.3. Proxy SOCKS.

Los proxy SOCKS son muy diferentes de los proxys 'normales'. Cuando por ejemplo usas un proxy HTTP lo que éste hace es coger las peticiones HTTP y hace la petición por ti y te devuelve los resultados. Sin embargo lo que hace el protocolo SOCKS, es casi equivalente a establecer un túnel IP con un firewall y a partir de ahí las peticiones del protocolo son entonces realizadas desde el firewall.

El cliente negocia una conexión con el servidor proxy SOCKS usando el protocolo SOCKS, nivel 5 del modelo OSI (capa de sesión). Una vez establecida la conexión todas las comunicaciones entre el cliente y proxy se realizan usando el protocolo SOCKS. El cliente le dice al proxy SOCKS que es lo que quiere y el proxy se comunica con el servidor externo y obtiene los resultados y se los manda al cliente. De esta forma el servidor externo sólo tiene que estar accesible desde el proxy SOCKS que es el que se va a comunicar con él.

En el proxy SOCKS es habitual implementar, como en la mayoría de proxys, autenticación y loggeo de de las sesiones.

2.3.6.3.9.4. Proxies transparentes.

Muchas organizaciones (incluyendo empresas, colegios y familias) usan los proxies para reforzar las políticas de uso de la red o para proporcionar seguridad y servicios de caché. Normalmente, un proxy Web o NAT no es transparente a la aplicación cliente: debe ser configurada para usar el proxy, manualmente. Por lo tanto, el usuario puede evadir el proxy cambiando simplemente la configuración. Una ventaja de tal es que se puede usar para redes de empresa.

Un proxy transparente combina un servidor proxy con NAT (Network Address Translation) de manera que las conexiones son enrutadas dentro del proxy sin configuración por parte del cliente, y habitualmente sin que el propio cliente conozca de su existencia. Este es el tipo de proxy que utilizan los proveedores de servicios de internet.

2.3.6.3.9.5. Reverse Proxy o Proxy inverso.

Un proxy inverso es un servidor proxy-caché "al revés". Es un servidor proxy que, en lugar de permitirles el acceso a Internet a usuarios internos, permite a usuarios de Internet acceder indirectamente a determinados servidores internos.

El servidor de proxy inverso es utilizado como un intermediario por los usuarios de Internet que desean acceder a un sitio web interno al enviar sus solicitudes indirectamente. Con un proxy inverso, el servidor web está protegido de ataques externos directos, lo cual fortalece la red interna. Además, la función caché de un proxy inverso puede disminuir la carga de trabajo del servidor asignado, razón por la cual se lo denomina en ocasiones acelerador de servidor.

Finalmente, con algoritmos perfeccionados, el proxy inverso puede distribuir la carga de trabajo mediante la redirección de las solicitudes a otros servidores similares. Este proceso se denomina equilibrio de carga.

Hay varias razones para instalar un "reverse proxy":

- **Seguridad:** el servidor proxy es una capa adicional de defensa y por lo tanto protege los servidores web.
- **Cifrado / Aceleración SSL:** cuando se crea un sitio web seguro, habitualmente el cifrado SSL no lo hace el mismo servidor web, sino que es realizado por el "reverse proxy", el cual está equipado con un hardware de aceleración SSL (Security Sockets Layer).
- **Distribución de Carga:** el "reverse proxy" puede distribuir la carga entre varios servidores web. En ese caso, el "reverse proxy" puede necesitar reescribir las URL de cada página web (traducción de la URL externa a la URL interna correspondiente, según en qué servidor se encuentre la información solicitada).
- **Caché de contenido estático:** Un "reverse proxy" puede descargar los servidores web almacenando contenido estático como imágenes u otro contenido gráfico.

2.3.6.3.9.6. Proxy NAT (Network Address Translation) / Enmascaramiento.

Otro mecanismo para hacer de intermediario en una red es el NAT. La traducción de direcciones de red (NAT, Network Address Translation) también es conocida como enmascaramiento de IPs. Es una técnica mediante la cual las direcciones fuente o destino de los paquetes IP son reescritas, sustituidas por otras (de ahí el "enmascaramiento").

Esto es lo que ocurre cuando varios usuarios comparten una única conexión a Internet. Se dispone de una única dirección IP pública, que tiene que ser compartida. Dentro de la red de área local (LAN) los equipos emplean direcciones IP reservadas para uso privado y será el proxy el encargado de traducir las direcciones privadas a esa única dirección pública para realizar las peticiones, así como de distribuir las páginas recibidas a aquel usuario interno que la solicitó.

Esta situación es muy común en empresas y domicilios con varios computadores en red y un acceso externo a Internet. El acceso a Internet mediante NAT proporciona una cierta seguridad, puesto que en realidad no hay conexión directa entre el exterior y la red privada, y así nuestros equipos no están expuestos a ataques directos desde el exterior.

Mediante NAT también se puede permitir un acceso limitado desde el exterior, y hacer que las peticiones que llegan al proxy sean dirigidas a una máquina concreta que haya sido determinada para tal fin en el propio proxy.

La función de NAT reside en los Cortafuegos y resulta muy cómoda porque no necesita de ninguna configuración especial en los equipos de la red privada que pueden acceder a través de él como si fuera un router.

2.3.6.3.9.7. Proxy abierto.

Este tipo de proxy es el que acepta peticiones desde cualquier computador, esté o no conectado a su red. En esta configuración el proxy ejecutará cualquier petición de cualquier computador que pueda conectarse a él, realizándola como si fuera una petición del proxy. Por lo que permite que este tipo de proxy se use como pasarela para el envío masivo de correos de spam.

Un proxy se usa, normalmente, para almacenar y redirigir servicios como el DNS o la navegación Web, mediante el cacheo de peticiones en el servidor proxy, lo que mejora la velocidad general de los usuarios. Este uso es muy beneficioso, pero al aplicarle una configuración "abierta" a todo internet, se convierte en una herramienta para su uso indebido.

Debido a lo anterior, muchos servidores, como los de IRC, o correo electrónicos, deniegan el acceso a estos proxys a sus servicios, usando normalmente listas negras ("BlackList").

2.3.7. SQUID COMO HERRAMIENTA PROXY.

El caché proxy por excelencia para plataformas Linux/UNIX es Squid, del que veremos cómo realizar su configuración, qué especificaciones requerirá el sistema donde lo vayamos a instalar, cómo llevar a cabo la configuración de un servidor proxy transparente y, finalmente, cómo obtener estadísticas sobre el uso del caché.

2.3.7.1. ¿Qué es Squid?

Squid es un Servidor Intermediario de alto desempeño que se ha venido desarrollando desde hace varios años y es hoy en día un muy popular y ampliamente utilizado entre los sistemas operativos como GNU/Linux y derivados de Unix®. Es muy confiable, robusto y versátil y se distribuye bajo los términos de la Licencia Pública General GNU (GNU/GPL).

Squid se comporta como un caché proxy, esto es, actúa como un agente que recibe peticiones de clientes (en este caso navegadores web) y pasa estas peticiones al proveedor de servicios apropiado. Cuando los datos llegan de nuevo al agente, este almacena una copia de los datos en un caché de disco.

Las ventajas de este sistema aparecen cuando varios clientes intentan acceder a los mismos datos: ya no hará falta ir a buscarlos otra vez a Internet, sino que se servirán directamente desde el caché de disco. De esta forma, los usuarios se benefician de un ahorro importante del volumen de transferencias en red.

Squid ofrece ventajas como la posibilidad de intercomunicar jerarquías de servidores proxys para repartir la carga entre ellos o establecer estrictas reglas de control de acceso para los clientes de las redes que quieran acceder al proxy. Además, con la ayuda de otras aplicaciones es posible controlar el acceso a determinadas páginas web u obtener estadísticas sobre cuáles son las webs más visitadas, con qué frecuencia los usuarios se conectan, etc.

Squid ha sido desarrollado durante muchos años y se le considera muy completo y robusto. Aunque orientado principalmente a HTTP y FTP es compatible con otros protocolos como Internet Gopher. Implementa varias modalidades de cifrado como TLS, SSL, y HTTPS.

2.3.7.2. Información general sobre cachés proxy.

2.3.7.2.1. Squid y seguridad.

También es posible emplear Squid junto con un cortafuegos para proteger una red interna del exterior mediante un caché proxy. Exceptuando a Squid, el cortafuego impide a todos los clientes establecer conexiones a servicios externos, haciendo que sea el proxy el que establezca todas las comunicaciones con la World Wide Web.

2.3.7.2.2. Cachés multinivel.

Es posible configurar varios proxys para que cooperen intercambiando objetos entre ellos. De esta forma se reduce la carga total del sistema y se aumenta la probabilidad de que el objeto se encuentre ya en la red local. Es posible configurar incluso jerarquías de cachés, de forma que se pueda pedir páginas a cachés del mismo nivel o enviar peticiones a otros proxys de jerarquía más alta para que pidan las páginas a otros cachés existentes en la red o las obtengan directamente de la fuente.

Elegir una buena topología para los cachés es muy importante para no acabar creando más tráfico del que ya había en la red antes de instalar los cachés. Por ejemplo, en el caso de una red local muy extensa conviene configurar un servidor proxy para cada subred y conectar estos a un proxy de jerarquía superior conectado a su vez al caché proxy del ISP. Toda esta comunicación se lleva a cabo mediante el protocolo ICP (Internet Cache Protocol) basado en UDP. Las transferencias de datos entre la mayoría de cachés se realizan mediante HTTP, protocolo basado en TCP.

Para encontrar el servidor más apropiado desde el que obtener un objeto, un caché envía una petición ICP a sus proxys vecinos. Estos le enviarán respuestas ICP con código “HIT”, si el objeto se encuentra efectivamente allí, o bien “MISS” en caso contrario. En caso que haya varios HIT, el proxy se decidirá por un servidor en especial en función de factores como la velocidad de respuesta o la proximidad, entre otros. Si las respuestas de los proxys vecinos no son satisfactorias, la petición se realizará al caché principal.

2.3.7.2.3. Objetos cacheados en Internet.

No todos los objetos disponibles en la red son estáticos. Existen páginas generadas dinámicamente por CGIs, contadores de visitantes o bien documentos que incluyen SSL para codificar el contenido y hacerlo más seguro. Por esos motivos se considera este tipo de objetos como no cacheables, ya que cada vez que se accede a ellos ya han cambiado.

Pero para todos los demás objetos que se guardan en el caché existe el problema de cuánto tiempo deben quedarse allí. Para determinarlo se asignan diferentes estados a los objetos del caché.

Los servidores web y los cachés proxy controlan el estado de un objeto añadiendo cabeceras como Last modified (última modificación) o Expires (expira) y la fecha correspondiente. Normalmente, los objetos desaparecerán antes del caché por la falta de espacio en el disco. Se utiliza algoritmos para sustituir objetos en el caché, como:

LRU Acrónimo de Least Recently Used, que traduce como Menos Recientemente Utilizado. En este algoritmo los objetos que no han sido accedidos en mucho tiempo son eliminados primero, manteniendo siempre en el caché a los objetos más recientemente solicitados. Ésta política es la utilizada por Squid de modo predefinido.

LFUDA Acrónimo de Least Frequently Used with Dynamic Aging, que se traduce como Menos Frecuentemente Utilizado con Envejecimiento Dinámico. En este algoritmo los objetos más solicitados permanecen en el caché sin importar su tamaño optimizando la eficiencia (hit rate) por octetos (Bytes) a expensas de la eficiencia misma, de modo que un objeto grande que se solicite con mayor frecuencia impedirá que se pueda hacer caché de objetos pequeños que se soliciten con menor frecuencia.

GDSF Acrónimo de Greedy Dual Size Frequency, que se traduce como Frecuencia de tamaño *Greedy Dua* (*codicioso dual*), que es el algoritmo sobre el cual se basa GDSF. Optimiza la eficiencia (hit rate) por objeto manteniendo en el caché los objetos pequeños más frecuentemente solicitados de modo que hay mejores posibilidades de lograr respuesta a una solicitud (hit). Tiene una eficiencia por octetos (Bytes) menor que el algoritmo LFUDA debido a que descarta del caché objetos grandes que sean solicitado con frecuencia.

2.3.7.3. Características de Squid.

Squid posee las siguientes características:

Proxy y Caché de HTTP, FTP, y otras URL.

Squid proporciona un servicio de Proxy que soporta peticiones http, HTTPS y FTP a equipos que necesitan acceder a Internet y a su vez provee la funcionalidad de caché especializado en el cual almacena de forma local las páginas consultadas recientemente por los usuarios. De esta forma, incrementa la rapidez de acceso a los servidores de información Web y FTP que se encuentra fuera de la red interna.

Proxy para SSL.

Squid también es compatible con SSL (Secure Socket Layer) con lo que también acelera las transacciones cifradas, y es capaz de ser configurado con amplios controles de acceso sobre las peticiones de usuarios.

Jerarquías de caché.

Squid puede formar parte de una jerarquía de caches. Diversos proxys trabajan conjuntamente sirviendo las peticiones de las páginas. Un navegador solicita siempre las páginas a un sólo proxy, si este no tiene la página en la caché hace peticiones a sus hermanos, que si tampoco las tienen las hacen a su/s padre/s... Estas peticiones se pueden hacer mediante dos protocolos: HTTP e ICMP.

ICP, HTCP, CARP, cache digests.

Squid sigue los protocolos ICP, HTCP, CARP y caché digests que tienen como objetivo permitir a un proxy "preguntarle" a otros proxys caché si poseen almacenado un recurso determinado.

Caché transparente.

Squid puede ser configurado para ser usado como proxy transparente de manera que las conexiones son enrutadas dentro del proxy sin configuración por parte del cliente, y habitualmente sin que el propio cliente conozca de su existencia. De modo predefinido Squid utiliza el puerto 3128 para atender peticiones, sin embargo se puede especificar que lo haga en cualquier otro puerto disponible o bien que lo haga en varios puertos disponibles a la vez.

WCCP.

A partir de la versión 2.3 Squid implementa WCCP (Web Cache Control Protocol). Permite interceptar y redirigir el trafico que recibe un router hacia uno o más proxys caché, haciendo control de la conectividad de los mismos. Además permite que uno de los proxys caché designado pueda determinar cómo distribuir el tráfico redirigido a lo largo de todo el array de proxys caché.

Control de acceso.

Ofrece la posibilidad de establecer reglas de control de acceso. Esto permite establecer políticas de acceso en forma centralizada, simplificando la administración de una red.

Aceleración de servidores HTTP.

Cuando un usuario hace petición hacia un objeto en Internet, este es almacenado en el caché, si otro usuario hace petición hacia el mismo objeto, y este no ha sufrido modificación alguna desde que lo accedió el usuario anterior, Squid mostrará el que ya se encuentra en el caché en lugar de volver a descargarlo desde Internet. Esta función permite navegar rápidamente cuando los objetos ya están en el caché y además optimiza enormemente la utilización del ancho de banda.

SNMP.

Squid permite activar el protocolo SNMP, este proporciona un método simple de administración de red, que permite supervisar, analizar y comunicar información de estado entre una gran variedad de máquinas, pudiendo detectar problemas y proporcionar mensajes de estados.

Caché de resolución DNS.

Squid está compuesto también por el programa dnsserver, que se encarga de la búsqueda de nombres de dominio. Cuando Squid se ejecuta, produce un número configurable de procesos dnsserver, y cada uno de ellos realiza su propia búsqueda en DNS. De este modo, se reduce la cantidad de tiempo que la caché debe esperar a estas búsquedas DNS.

2.3.7.4. Proxy cache en Squid.

El proxy caché es una manera de guardar los objetos solicitados de Internet (por ejemplo, datos como páginas web) disponibles vía protocolos HTTP, FTP y Gopher en un sistema más cercano al lugar donde se piden. Los navegadores web pueden usar la caché local Squid como un servidor proxy HTTP, reduciendo el tiempo de acceso así como el consumo de ancho de banda. Esto es muchas veces útil para los proveedores de servicios de Internet para incrementar la velocidad de sus consumidores y para las redes de área local que comparten la conexión a Internet.

Debido a que también es un proxy (es decir, se comporta como un cliente en lugar del cliente real), puede proporcionar un cierto grado de anonimato y seguridad. Sin embargo, también puede introducir problemas significativos de privacidad ya que puede registrar mucha información, incluyendo las URL solicitadas junto con otra información adicional como la fecha de la petición, versión del navegador y del sistema operativo, etc.

Un programa cliente (por ejemplo, un navegador) o bien tiene que especificar explícitamente el servidor proxy que quiere utilizar (típico para consumidores de ISP) o bien podría estar usando un proxy sin ninguna configuración extra. A este hecho se le denomina caché transparente, en el cual todas las peticiones HTTP son interceptadas por Squid y todas las respuestas guardadas en caché. Esto último es típico en redes corporativas dentro de una red de acceso local y normalmente incluye los problemas de privacidad mencionados previamente.

Squid tiene algunas características que pueden facilitar establecer conexiones anónimas. Características tales como eliminar o modificar campos determinados de la cabecera de peticiones HTTP de los clientes. Esta política de eliminación y alteración de cabeceras se establece en la configuración de Squid.

El usuario que solicita páginas a través de una red que utiliza Squid de forma transparente, normalmente no es consciente de este proceso o del registro de información relacionada con el proceso.[4]

2.3.7.5. Requerimientos del sistema.

Los requerimientos de Hardware de Squid son generalmente modestos, en las siguientes secciones se explican en orden de importancia los distintos factores del sistema.

[4] SQUID
http://es.wikipedia.org/wiki/Squid_(programa)

2.3.7.5.1. Discos duros.

Cuando se trata de cachés, la velocidad es un parámetro importantísimo. En los discos duros este parámetro se mide mediante su "tiempo medio de acceso" en milisegundos, que debe ser lo más bajo posible. Para lograr una velocidad elevada se recomienda utilizar discos duros rápidos.

Debido a que en la mayoría de los casos Squid lee o escribe pequeños bloques del disco duro, el tiempo de acceso del disco duro es más importante que su capacidad de transferencia de datos. Precisamente en este contexto muestran su valía los discos duros con una alta velocidad de rotación, ya que permiten un posicionamiento más rápido de la cabeza de lectura. Hoy en día, los discos duros SCSI de mayor rapidez pueden alcanzar tiempos de acceso inferiores a 4 milisegundos. Otra posibilidad para aumentar la velocidad consiste en el uso paralelo de varios discos duros o de una estructura RAID.

2.3.7.5.2. Tamaño del caché de disco.

Depende de varios factores. En un caché pequeño la probabilidad de un HIT (el objeto ya se encuentre en el caché) será pequeña, ya que el caché se llenará con facilidad y se deberá sustituir los objetos antiguos por nuevos. En cambio, en el caso de disponer de por ejemplo 1 GB de disco para cachear, y de que los usuarios sólo necesiten 10 MB al día para navegar, se tardará al menos 100 días en llenar el caché. El método más fácil para determinar el tamaño del caché es en función del tráfico máximo que pase por el mismo.

Como Squid utiliza una pequeña cantidad de memoria por cada respuesta de cache existe una relación entre el espacio de la memoria y el cache de disco, como regla general se puede utilizar que se necesitan 32 MB de memoria RAM por cada GB de cache de disco, por lo cual un sistema con 1 GB de memoria puede almacenar 32 GB de cache sin afectar su rendimiento.

2.3.7.5.3. Memoria RAM.

La cantidad de memoria requerida por Squid está relacionada directamente con la cantidad de objetos que se encuentran en el caché. Squid también almacena referencias a los objetos en el caché y objetos utilizados frecuentemente en la memoria RAM para optimizar la obtención de los mismos. La memoria RAM es muchísimo más rápida que el disco duro.

Squid también guarda muchos otros datos en la memoria, como por ejemplo una tabla con todas las direcciones IP utilizadas, un caché para los nombres de dominio totalmente cualificados, objetos "calientes" (los que más se solicitan), buffers, listas de control de acceso, etc. El no poseer memoria suficiente causa una drástica degradación en el rendimiento de los sistemas.

2.3.7.5.4. Potencia del procesador.

Squid no es un programa que consuma mucho CPU. Solamente al arrancar y comprobar el contenido del caché es cuando se trabaja más intensamente con el procesador. El uso de máquinas con multiprocesador tampoco incrementa el rendimiento del sistema. Para obtener una mayor efectividad, es preferible aumentar la cantidad de memoria RAM o bien utilizar discos más rápidos antes que cambiar el procesador por otro más potente.

2.3.7.6. Equipamiento lógico necesario.

Para poder llevar al cabo los procedimientos descritos en este documento, es necesario tener instalado al menos lo siguiente:

- Al menos la última versión de Squid- 3.1.15.STABLE6.
- Todos los parches de seguridad disponibles para la versión del sistema operativo que esté utilizando.
- Un muro cortafuegos configurado de preferencia Iptables.

Debe tomarse en consideración que, de ser posible, se debe utilizar siempre las versiones estables más recientes de todo equipamiento lógico que vaya a ser instalado para realizar los procedimientos descritos, a fin de contar con los parches de seguridad necesarios.

2.3.7.6.1. Instalación de Squid.

Para poder instalar el servicio de Squid tendremos que ejecutar lo siguiente en un terminal de CentOS como usuario root.

```
[root@servidorCentOS ~]# yum –y install squid
```

2.3.7.6.2. Actualización del sistema operativo CentOS.

Linux CentOS utiliza Yum como herramienta para administrar sus paquetes, de manera análoga, a continuación se describe el breve proceso que se debe realizar para actualizar los paquetes y el sistema operativo desde la línea de comando utilizando esta herramienta.

```
[root@servidorCentOS ~]# yum check-update
[root@servidorCentOS ~]# yum –y update
```

2.3.7.6.3. Firewall de Linux o Iptables.

El kernel Linux incluye el subsistema Netfilter, que es usado para manipular o decidir el destino del tráfico de red entre o a través de su red. Todas las soluciones firewall Linux modernas utilizan este sistema para el filtrado de paquetes.

El sistema de filtrado de paquetes del núcleo resulta de poca ayuda a los administradores si no se tiene una interfaz de usuario para gestionarlo. Éste es el propósito de Iptables. Cuando un paquete llega a su servidor, éste es gestionado por

el subsistema Netfilter para aceptarlo, manipularlo o rechazarlo basándose en las reglas suministradas a éste vía Iptables. Así, Iptables es todo lo que necesita para manejar su cortafuego si está familiarizado con él, pero existen muchos interfaces de usuario disponibles para simplificar esta tarea.

Iptables está basado en el uso de TABLAS dentro de las tablas, CADENAS, formadas por agrupación de REGLAS, parámetros que relativizan las reglas y finalmente una ACCIÓN, que es la encargada de decir qué destino tiene el paquete.

2.3.7.6.3.1. Activación del enrutamiento en Iptables de Linux.

Las funciones de enrutamiento mediante NAT son realizadas por el cortafuego que analizará los paquetes provenientes de la red local interna cuyo destino sea Internet y los modificará convenientemente para que salgan hacia Internet como si fueran emitidos por el servidor.

Para posibilitar que nuestro servidor Linux sea capaz de comportarse como un router y hacer de puerta de enlace para los PCs de nuestra red local, será necesario crear un script que configure el cortafuego Iptables para que realice NAT desde dentro de la red local hacia Internet.

Para activar el enrutamiento en un sistema Linux, tan solo basta con poner a '1' la variable ip_forward del sistema, es decir, basta con ejecutar desde un terminal como usuario de root:

```
[root@servidorCentOS ~]# echo "1" > /proc/sys/net/ipv4/ip_forward
```

Posteriormente tendríamos que configurar el filtrado para que acepte el redireccionamiento de paquetes desde dentro hacia fuera de nuestra red y mediante NAT permita que los PCs de la red interna naveguen con la dirección IP 'publica' del servidor.

2.3.7.6.3.2. Resumen de operación de Iptables.

Iptables permite al administrador del sistema definir reglas acerca de qué hacer con los paquetes de red. Las reglas se agrupan en cadenas: cada cadena es una lista ordenada de reglas. Las cadenas se agrupan en tablas: cada tabla está asociada con un tipo diferente de procesamiento de paquetes.

Las Tablas:

TABLA	FUNCIÓN	CADENA	FUNCIÓN de la CADENA
FILTER	Filtrado de paquetes	INPUT	Filtrado de paquetes que llegan al firewall
		OUTPUT	Filtrado de los paquetes de salida
		FORWARD	Permite el paso de paquetes a otra dirección del firewall
NAT	Enrutamiento de direcciones de red	PREROUTING	Chequea la dirección de red antes de reenviarla. Facilita la modificación de la información para facilitar el enrutado Se usa también como DESTINATION NAT o DNAT
		POSTROUTING	Tratamiento de la dirección IP después del enrutado. Esto hace que no sea necesaria la modificación del destino de la dirección IP del paquete como en pre-routing. Se usa como o SNAT
		OUTPUT	Interpretación de las direcciones de Red de los paquetes que salen del firewall. Escasamente usado.

Tabla Nº 2.2: Las tablas Filter y NAT que utiliza el firewall Iptables.

Comandos de Iptables:

Como hacíamos referencia más arriba, dentro de las tablas hay cadenas a su vez formadas por agrupaciones de reglas. Es importante ver que cada tabla tiene cadenas por defecto, que no se pueden eliminar.

A las CADENAS por defecto podemos unir cadenas creadas por nosotros mismos para un mejor funcionamiento del filtrado o el enrutamiento.

El comando IPTABLES tiene a su vez parámetros y comandos que permitirán definir el comportamiento de una o varias reglas. Esto es, agregar una regla, modificar una regla existente, eliminar el nombre de una cadena.

COMANDO	FUNCIÓN
-A	Agregar nueva regla a la cadena especificada.
-I	Insertar nueva regla antes de la regla rulenum en la cadena especificada.
-R	Reemplazar la regla (rulenum) en la cadena especificada.
-E	Modifica el nombre de la cadena.
-L	Listado de reglas de la cadena especificada. Si no se determina una cadena en particular, listará todas las cadenas existentes.
-N	Crear nueva cadena asociándola a un nombre.
-P	Modifica la acción por defecto de la cadena preseleccionada.
-D	Eliminar la regla_número(rulenum) en la cadena seleccionada.
-Z	Pone los contadores de paquetes y bytes a cero en la cadena seleccionada.

Tabla Nº 2.3: Comandos que definen las reglas de Iptables.

Parámetros:

Todas las reglas en Iptables tienen definida su condición por los parámetros, que constituyen su parte primordial. Algunos de estos parámetros son:

PARÁMETRO	FUNCIÓN
-i	Interfaz de entrada (eth0,eth1,eth2...)
-o	Interfaz de salida (eth0,eth1,eth2...)
--sport	Puerto de origen
--dport	Puerto destino
-p	El protocolo del paquete a comprobar, tcp, udp, icmp ó all. Por defecto es all
-j	Esto especifica el objetivo de la cadena de reglas, o sea una acción
--line-numbers	Cuando listamos las reglas, agrega el número que ocupa cada regla dentro de la cadena

Tabla Nº 2.4: Parámetros que definen las reglas de Iptables.

Acciones:

Y finalmente, las ACCIONES que estarán siempre al final de cada regla que determinará qué hacer con los paquetes afectados. Si no se especifica ninguna acción, se ejecutará la opción por defecto que cada cadena tiene asignada. Las acciones serían:

ACCEPT	Paquete aceptado.
REJECT	Paquete rechazado. Se envía notificación a través del protocolo ICMP.
DROP	Paquete rechazado. Sin notificación
MASQUERADE	Enmascaramiento de la dirección IP origen de forma dinámica. Esta acción es sólo válida en la tabla NAT en la cadena postrouting.
DNAT	Enmascaramiento de la dirección destino, muy utilizada para re-enrutado de paquetes.
SNAT	Enmascaramiento de la IP origen de forma similar a masquerade, pero con IP fija.

Tabla Nº 2.5: Acciones a tomar en las reglas de Iptables.

Ya tenemos los componentes esenciales para formar las reglas que determinarán la aceptación o denegación de entrada y salida de paquetes de nuestra máquina, tanto a través de Internet, como de nuestra propia red local.

Paso de los paquetes por los filtros

Cuando un paquete llega al firewall, primero pasa unos filtros, ver Grafico Nº 2.7. En primer lugar pasa al filtro de PREROUTING donde se puede manipular el paquete modificando sus datos de destino, por ejemplo redirigir a otra máquina o a otro puerto. Esto se conoce como DNAT (destination network address translation).

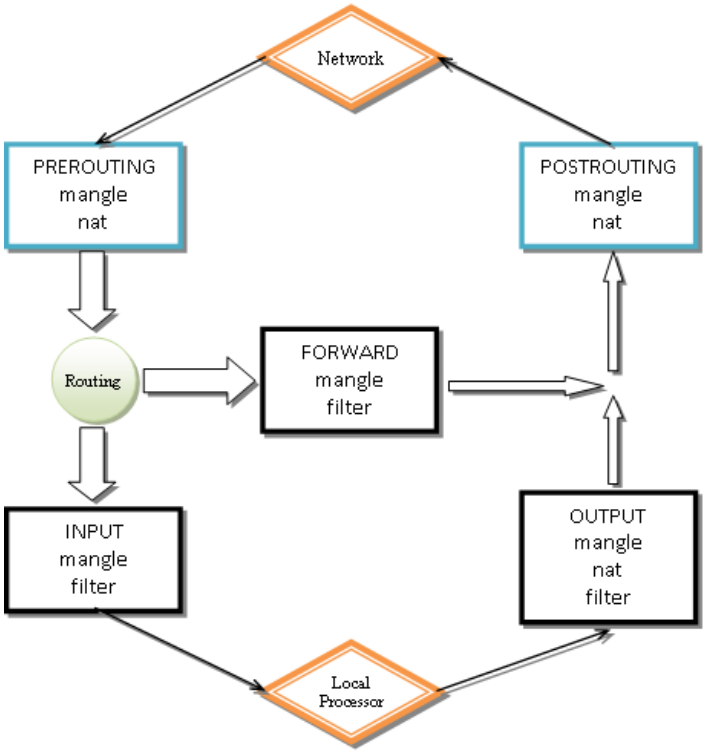

Grafico Nº 2.7: Tratamiento de paquetes en Iptables.

Cuando un paquete llega al firewall, primero pasa unos filtros, ver Grafico Nº 2.7. En primer lugar pasa al filtro de PREROUTING donde se puede manipular el paquete modificando sus datos de destino, por ejemplo redirigir a otra máquina o a otro puerto. Esto se conoce como DNAT (destination network address translation).

Una vez que el paquete de entrada está preparado se comprueba si va dirigido al propio ordenador en cuyo caso pasa al filtro de entrada (INPUT), o bien si va dirigido a otra máquina, en cuyo caso se dirige al filtro de reenvío (FORWARD).

Si el paquete iba destinado a la máquina local y ha pasado el filtro de entrada se le entrega al proceso local que lo solicitó. Si un proceso local genera un paquete que tiene que enviar a la red primero pasa por el filtro de salida (OUTPUT). El filtro de salida podría manipular el paquete modificando el destino.

Si el paquete ha pasado el filtro de reenvío (FORWARD) pasaría el filtro de POSTROUTING. Igualmente si el paquete local pasa el filtro de salida también pasa al filtro de POSTROUTING). En el filtro de POSTROUTING se pueden manipular los datos de origen de un paquete. En esta fase podemos realizar SNAT (Source network address translation), es decir, manipular los datos de origen del paquete. Cada vez que se tenga que configurar un filtro se tiene que tener muy en cuenta el esquema descrito.

Creación de las Reglas de Iptables

Iptables tiene una serie de comandos que permite manipular (insertar, eliminar) las reglas del conjunto predeterminado. En general, la estructura de un comando de Iptables es el siguiente:

```
iptables [-t <table-name>] <command> <chain-name> <parameter-1>
\<option-1> <parameter-n> <option-n>
```

Donde:

- `<table-name>`: permite al usuario seleccionar una tabla diferente de la tabla filter por defecto que se usa con el comando.
- `<command>`: es el centro del comando, dictando cuál es la acción específica a realizar, como pueda ser añadir o borrar una regla de una cadena particular, que es lo que se especifica en la opción `<chain-name>`

- `<parameter-n> <option-n>`: Los pares de parámetros y opciones que realmente definen la forma en la que la regla funcionará y qué pasará cuando un paquete cumpla una regla.

Orden de las Reglas

El orden de las reglas de un firewall define su comportamiento. El filtro va comparando el paquete con cada una de las reglas hasta que se encuentra una que verifica y en ese caso se lleva a cabo lo que indique esa regla (aceptar o denegar); una vez realizada la acción no se comprueban más reglas. Si ponemos reglas muy permisivas entre las primeras del firewall, puede que las siguientes no se apliquen y no sirvan de nada.

Política Predeterminada

También se puede dar el caso de que un paquete no haya verificado ninguno de los filtros una vez que se los ha comprobado todos. En este caso no se sabe si se tiene que aceptar o rechazar este paquete. Para resolver esta situación Iptables dispone de una política predeterminada que permite indicar qué hacer con los paquetes que no hayan verificado ninguna regla. La política predeterminada será algunas de las acciones que hemos definido y se define con la opción "-P".

Por ejemplo:

```
iptables -P INPUT -j ACCEPT
```

Definiría como aceptar la política predeterminada del filtro INPUT.

La política predeterminada define un esquema distinto de cortafuegos. Si la política predeterminada es aceptar entonces se tendrá que denegar todo aquello que no interese. Si la política predeterminada es denegar entonces se tiene que permitir todo aquello que sea de interés.

Cada uno de los modelos de cortafuegos tiene sus ventajas e inconvenientes. En el primer caso, con la política predeterminada de aceptar es mucho más fácil la gestión del firewall. En este caso es útil cuando se sabe claramente qué puertos se quiere proteger y el resto no importa y se acepta. El problema que plantea es no poder controlar qué se tiene abierto o que en un momento dado se instale un software nuevo, un troyano por ejemplo, que abra un puerto determinado, o que no se sepa que determinados paquetes ICMP son peligrosos. Si la política predeterminada es ACEPTAR y no se protege explícitamente el sistema puede ser inseguro.

Cuando la política predeterminada es DENEGAR todo lo que no se acepte explícitamente será denegado por lo que el sistema puede fallar por despiste o desconocimiento en lo que se está permitiendo. Es más complicado establecer este tipo de cortafuegos, hay que tener muy claro cómo funciona el sistema y qué es lo que se tiene que aceptar sin caer en la tentación de introducir reglas muy permisivas. Esta configuración de cortafuegos es la recomendada, aunque no es aconsejable usarla si no se domina mínimamente el sistema.

2.3.7.7. Configuración manual de Squid.

El fichero de configuración de SQUID se halla en "/etc/squid/squid.conf" y hemos de editarlo con nuestra herramienta favorita para realizar los cambios adecuados y conseguir que cumpla su tarea con cierta seguridad para nuestro sistema.

```
[root@servidorCentOS ~]# sudo gedit /etc/squid/squid.conf
```

Este fichero de configuración consta de multitud de parámetros configurables que ajustan el servidor a nuestras necesidades. Trataremos de reflejar aquellos indispensables para un óptimo funcionamiento, sin antes recordar que la aplicación desarrollada realiza todo esta configuración y otros datos necesarios en una interfaz amigable.

Para poder iniciar Squid por primera vez, no es necesario hacer cambios en este archivo, aunque los clientes externos tendrán inicialmente el acceso denegado. Las opciones son muy extensas y están documentadas con muchos ejemplos en el archivo "/etc/squid/squid.conf" pre instalado. Casi todas las líneas comienzan por el símbolo "#" (significa que la línea está comentada y su contenido no se evaluará); las opciones relevantes se encuentran al final de la línea. Los valores por defecto corresponden casi siempre a los valores que necesitaremos, así que para muchas opciones sólo será necesario quitar el símbolo de comentario.

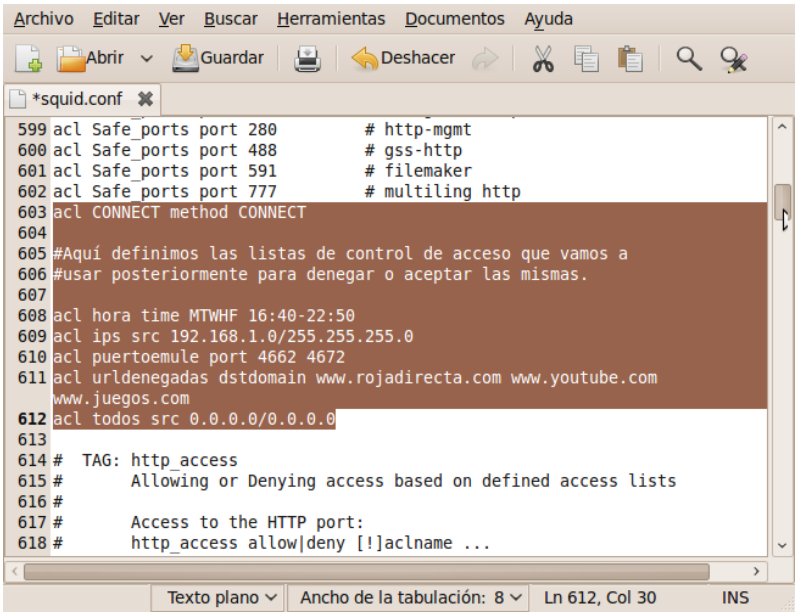

Grafico Nº 2.8: Archivo de configuración de Squid.

2.3.7.8. Opciones generales de configuración.

Existen un gran número de parámetros, de los cuales recomendamos configurar los siguientes:

- Al menos una Lista de Control de Acceso
- Al menos una Regla de Control de Acceso
- http_port
- cache_dir

2.3.7.8.1. Parámetro http_port ¿Que puerto utilizar para Squid?

Squid por defecto utiliza el puerto 3128 para atender peticiones, sin embargo se puede especificar que lo haga en cualquier otro puerto o bien que lo haga en varios puertos a la vez.

En el caso de un Proxy Transparente, se utilizará el puerto 80 y se valdrá del re-direccionamiento de peticiones de modo tal que no habrá necesidad alguna de modificar la configuración de los navegadores Web para utilizar el servidor Proxy, bastará con utilizar como puerta de enlace al servidor.

Regularmente algunos programas utilizados comúnmente por los usuarios suelen traer por defecto el puerto 8080 para utilizarse al configurar que servidor proxy utilizar. Si queremos aprovechar esto en nuestro favor y ahorrarnos el tener que dar explicaciones innecesarias al usuario, podemos especificar que Squid escuche peticiones en dicho puerto también. Siendo así localice la sección de definición de http_port, y especifique:

```
http_port 3128
http_port 8080
```

2.3.7.8.2. Parámetro cache_mem.

Establece la cantidad de memoria RAM dedicada para almacenar los datos más solicitados. El parámetro cache_mem establece la cantidad ideal de memoria para:

- Objetos en tránsito.
- Objetos Hot.
- Objetos negativamente almacenados en el caché.

Los datos de estos objetos se almacenan en bloques de 4 Kb. El parámetro cache_mem especifica un límite máximo en el tamaño total de bloques acomodados, donde los objetos en tránsito tienen mayor prioridad. Sin embargo los objetos Hot y aquellos negativamente almacenados en el caché podrán utilizar la memoria no utilizada hasta que esta sea requerida. De ser necesario, si un objeto en tránsito es mayor a la cantidad de memoria especificada, Squid excederá lo que sea necesario para satisfacer la petición. Por defecto se establecen 8 MB. Puede especificarse una cantidad mayor si así se considera necesario.

```
cache_mem 8MB
```

2.3.7.8.3. Parámetro cache_dir ¿Cuanto almacenar de Internet en el disco duro?

Es cuanto tamaño se desea que tenga el cache en el disco duro para Squid, es decir ¿Cuanto desea almacenar de Internet en el disco duro? Por defecto Squid utilizará un cache de 100 MB, de modo tal que encontrará la siguiente línea:

```
cache_dir ufs /var/spool/squid 100 16 256
```

Se puede incrementar el tamaño del cache hasta donde lo desee el administrador. Mientras más grande el cache, más objetos de almacenarán en éste. Y por lo tanto se utilizará menos el ancho de banda. Los números 16 y 256 significan que el directorio del cache contendrá 16 subdirectorios con 256 niveles cada uno. No modifique esto números, no hay necesidad de hacerlo.

Es muy importante considerar que si se especifica un determinado tamaño de cache y este excede al espacio real disponible en el disco duro, Squid se bloqueará inevitablemente.

2.3.7.8.4. Parámetro cache_access_log.

Especifica en que directorio se realizara el registro de accesos al Squid, este parámetro es importante para definir un análisis de estadísticas.

```
cache_access_log /var/log/squid/access.log
```

2.3.7.8.5. Parámetro cache_log.

Define en donde se almacenaran los mensajes del comportamiento de la cache de Squid.

```
cache_log /var/log/squid/cache.log
```

2.3.7.8.6. Parámetro cache_store_log.

El parámetro cache_store_log define el archivo de log en el que se registran los objetos puestos en el cache.

```
cache_store_log /var/log/squid/store.log
```

2.3.7.8.7. Parámetro error_directory.

El parámetro error_directory define la ruta del directorio en el cual se almacenan los mensajes que son devueltos a los clientes web, los mensajes por default están en inglés, si desea cambiar los mensajes a español cambié el parámetro error_directory por ejemplo:

```
error_directory /usr/share/squid/errors/Spanish
```

2.3.7.8.8. Parámetro visible_hostname.

Es el nombre del equipo, el nombre debe ser igual a los siguientes ficheros /etc/hosts y en /etc/sysconfig/network. Este parámetro no viene configurado en el archivo de configuración, tendremos que agregar y que en ocasiones pueda ser que nuestro servicio de Squid no quiera iniciar.

```
visible_hostname servidorCentOS
```

2.3.7.8.9. Controles de Acceso (ACL).

Los controles de acceso son la parte más importante en la configuración del servidor Squid, estas se utilizan para dar acceso a los usuarios y también para negarlo. Las ACL pueden usarse para restringir o prevenir acceso a ciertos sitios o contenidos.

```
acl <nombre_acl> <tipo_acl> <datos>
acl <nombre_acl> <tipo_acl> <archivo_de_datos>
```

Cuando usamos un "archivo_de_datos", cada descripción se corresponde con una línea del archivo.

El control de acceso define si se permite o deniega el acceso a las reglas para que empecemos a crear el filtrado. Nomenclatura:

```
http_access allow/deny Regla
```

Define a quién le está permitido usar el proxy y quién puede acceder a Internet. Para todo esto se deberán definir primero las ACL correspondientes. En general se puede permitir el acceso mediante *allow* o bien negarlo con *deny*. Se

puede crear una lista completa de entradas *http_access* que será procesada de arriba hacia abajo y dependiendo de cómo estén configuradas las reglas se podrá acceder o no a Internet para cada URL. Por eso la última entrada de todas debe ser *http_access deny all*. En el ejemplo siguiente localhost dispone de acceso libre mientras que todos los otros hosts tienen el acceso denegado.

```
http_access allow localhost
http_access deny all
```

2.3.7.8.9.1. Tipos de ACL.

src

Especifica una dirección origen de una conexión en formato IP/máscara. Por ejemplo, utilizaremos una ACL de tipo src para especificar la red local:

```
acl red_local 192.168.10.0/24
```

También podemos especificar rangos de direcciones mediante una acl de tipo src:

```
acl administrativo src 192.168.10.10-192.168.10.25/24
```

dst

Especifica una dirección destino de una conexión en formato IP/máscara.

```
acl google dst 216.239.0.0/24
```

Las definiciones son idénticas a las acl de tipo src salvo que se aplican al destino de las conexiones, no al origen.

srcdomain y dstdomain

Estos tipos de acl especifican un nombre de dominio. En el caso de srcdomain es el dominio origen y se determina por resolución DNS inversa de la IP de la máquina. En el caso de dstdomain el nombre del dominio se comprueba con el dominio que se haya especificado en la petición de página web. Por ejemplo:

```
acl google_com dstdomain google.com
```

srcdom_regex y dstdom_regex

Especifican una expresión regular que verifican los dominios origen o destino. La expresión regular hace distinción entre mayúsculas y minúsculas salvo que incluyamos la opción "-i" que evita dicha distinción. Ejemplo:

```
acl google_todos dstdom_regex -i google\..*
```

Observamos como al incluir "-i" estamos indicando que no haga distinción entre mayúsculas y minúsculas.

url_regex

Permite especificar expresiones regulares para comprobar una url completa, desde el http:// inicial. Por ejemplo, vamos a establecer una acl que se verifique con todos los servidores cuyo nombre sea adserver:

```
url_regex serv_publicidad ^http://adserver.*
```

En otro ejemplo podemos ver una acl que verifique las peticiones de archivos mp3:

```
url_regex archivos_mp3 -i mp3$
```

2.3.7.9. Configuración Squid Transparente.

Este tipo de configuración de squid transparente, lo que hace es que conexiones son enrutadas al proxy sin hacer ninguna configuración en los clientes para que tengan salida a internet. Este tipo de configuración depende de reglas de nuestro firewall.

2.3.7.9.1. Parámetro http_port para el Squid transparente.

Solamente tendremos que configurar este parámetro para que se un proxy transparente. Se le debe indicar la IP del servidor squid, puerto de escucha y la palabra transparente.

```
http_port 3128
```

por

```
http_port 192.168.10.1:3128 transparent
```

2.3.7.9.2. Reglas del Firewall para el Squid transparente.

Para poder configurar este tipo de proxy transparente, tendremos que configurar reglas de firewall, en nuestro caso usaremos reglas de Iptables ya que es la herramienta más utilizada en todas distribuciones GNU/Linux. Pero para que funcione de manera transparente debemos de aplicar la siguiente regla en Iptables.

```
iptables -t nat -A PREROUTING -i eth1 -p tcp –dport 80 -j REDIRECT --to-port 3128
```

Con esto estamos desviando el tráfico que venga por la LAN que vaya por web al puerto 3128. Con esto ya hicimos transparente nuestro proxy pero no se pueden desplegar las páginas seguras, para eso necesitamos aplicar otras reglas en Iptables liberando el puerto 443, y lo hacemos de la siguiente manera:

```
iptables -t nat -A PREROUTING -i eth1 -p tcp –dport 443 -j REDIRECT
--to-port 3128
```

Habilitamos el reenvío de paquetes dentro de la red.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Y Guardamos las reglas con el siguiente comando.

```
iptables-save > /etc/sysconfig/iptables
```

Reiniciamos el servicio de firewall

```
/etc/init.d/iptables restart
```

2.3.7.10. Iniciando, reiniciando y añadiendo Squid al arranque del sistema.

Una vez terminada la configuración, ejecute el siguiente comando para iniciar por primera vez Squid:

```
/etc/rc.d/init.d/squid start
```

Si necesita reiniciar para probar cambios hechos en la configuración, ejecute lo siguiente:

```
/etc/rc.d/init.d/squid restart
```

Si desea que Squid inicie de manera automática la próxima vez que inicie el sistema, ejecute lo siguiente:

```
chkconfig squid on
```

2.3.7.11. Verificación Logs.

Squid almacena en el directorio "/var/log/Squid" información sobre los accesos, diálogos con otros servidores Squid, etc. Existen varios archivos de logs, el que nos brinda información sobre el acceso al servidor es access.log. Cuando se entrega a un cliente un objeto que se encontraba almacenado, se produce un HIT y si el objeto debe ser consultado hacia internet entonces es un MISS.

El análisis de los logs por lo general se realiza con herramientas de software independientes de Squid. Dos de las más utilizadas son SARG (Squid Analysis Report Grpahics) y Webalizer, las mismas generan reportes gráficos con estadísticas en un archivo HTML. Son una excelente herramienta para llevar un control detallado sobre la utilización de la navegación web.

2.3.8. CALIDAD DE SERVICIO (QoS).

QoS o Calidad de Servicio (Quality of Service) son las tecnologías que garantizan la transmisión de cierta cantidad de información en un tiempo dado. Calidad de servicio es la capacidad de dar un buen servicio. Es especialmente importante para ciertas aplicaciones tales como la transmisión de vídeo o voz.

Normalmente la Internet trabaja con la filosofía del mejor esfuerzo: cada usuario comparte ancho de banda con otros y, por lo tanto, la transmisión de sus datos corriente con las transmisiones de sus datos concurre con las transmisiones de los demás usuarios. Los datos empaquetados son encaminados de la mejor forma posible, conforme las rutas y bandas disponibles. Cuando hay congestionamiento, los paquetes son descartados sin distinción.

Con la implantación de calidad de servicio (QoS), es posible ofrecer más garantía y seguridad para las aplicaciones avanzadas, una vez que el tráfico de estas aplicaciones pasa a tener prioridad en relación con aplicaciones tradicionales.

Con el uso del QoS los paquetes son marcados para distinguir los tipos de servicios y los enrutadores son configurados para crear filas distintas para cada aplicación, de acuerdo con las prioridades de las mismas. Así, una faja de ancho de banda, dentro del canal de comunicación, es reservada para que, en el caso de congestionamiento, determinados tipos de flujos de datos o aplicaciones tengan prioridad en la entrega.

2.3.8.1. Problemas en redes de datos conmutados.

Muchas cosas le ocurren a los paquetes desde su origen al destino, resultando los siguientes problemas vistos desde el punto de vista del transmisor y receptor:

Paquetes sueltos

Los ruteadores pueden fallar en liberar algunos paquetes si ellos llegan cuando los buffers ya están llenos. Algunos, ninguno o todos los paquetes pueden quedar sueltos dependiendo del estado de la red, y es imposible determinar qué pasará de antemano. La aplicación del receptor puede preguntar por la información que será retransmitida posiblemente causando largos retardos a lo largo de la transmisión.

Retardos

Puede ocurrir que los paquetes tomen un largo período en alcanzar su destino, debido a que pueden permanecer en largas colas o tomen una ruta menos directa para prevenir la congestión de la red. En algunos casos, los retardos excesivos pueden inutilizar aplicaciones tales como VoIP o juegos en línea.

Jitter

Los paquetes del transmisor pueden llegar a su destino con diferentes retardos. Un retardo de un paquete varía impredeciblemente con su posición en las colas de los ruteadores a lo largo del camino entre el transmisor y el destino. Esta

variación en retardo se conoce como jitter y puede afectar seriamente la calidad del flujo de audio y/o vídeo.

Entrega de paquetes fuera de orden

Cuando un conjunto de paquetes relacionados entre sí son encaminados a Internet, los paquetes pueden tomar diferentes rutas, resultando en diferentes retardos. Esto ocasiona que los paquetes lleguen en diferente orden de cómo fueron enviados. Este problema requiere un protocolo que pueda arreglar los paquetes fuera de orden a un estado isócrono una vez que ellos lleguen a su destino. Esto es especialmente importante para flujos de datos de vídeo y VoIP donde la calidad es dramáticamente afectada tanto por latencia y pérdida de sincronía.

Errores

A veces, los paquetes son mal dirigidos, combinados entre sí o corrompidos cuando se encaminan. El receptor tiene que detectarlos y justo cuando el paquete es liberado, pregunta al transmisor para repetirlo así mismo.

2.3.8.2. QoS en escenarios inalámbricos.

El entorno inalámbrico es muy hostil para medidas de Calidad de Servicio debido a su variabilidad con el tiempo, ya que puede mostrar una calidad nula en un cierto instante de tiempo. Esto implica que satisfacer la QoS resulta imposible para el 100% de los casos, lo que representa un serio desafío para la implementación de restricciones de máximo retardo y máxima varianza en el retardo (jitter) en sistemas inalámbricos.

Los sistemas de comunicaciones ya estandarizados con restricciones QoS de retardo y jitter en entornos inalámbricos (por ejemplo en GSM y UMTS) sólo pueden garantizar los requisitos para un porcentaje (<100%) de los casos. Esto implica una caída del servicio, generando los cortes de llamadas y/o los mensajes de “red ocupada”.

2.3.8.3. Control del ancho de banda utilizando Delay Pools.

Uno de los usos más útiles que tenemos con Squid, además de usarlo como caché es el de poder administrar el consumo de ancho de banda de los clientes que se conectan al mismo. Mediante Squid podemos establecer límites al ancho de banda mediante una de sus características de configuración denominada “Delay pools”. Establecemos una definición de regulación de ancho de banda, establecemos una ACL y asociamos la regulación a la ACL.

Para poder usar los Delay Pools debemos tener configuradas ACL’s, tanto sea por subred, grupo de equipos, destino, o lo que necesitamos controlar. Delay pools es la respuesta de Squid frente al control de ancho de banda y el traffic shaping (catalogación de tráfico). Esto se realiza limitando el rate que el Squid retorna los datos desde su cache.

Los Delay pools son en esencia "cubos de ancho de banda” (bandwidth buckets). La solicitud a una respuesta es demorada hasta que cierta cantidad de ancho de banda esté disponible desde un cubo. Squid llena con cierta cantidad de tráfico los cubos por cada segundo y los clientes del Cache consumen los datos llenados desde esos cubos.

El tamaño de un cubo determina cuánto límite de ancho de banda está disponible en un cliente. Si un cubo se encuentra lleno, un cliente puede descargar a máxima velocidad de la conexión disponible (sin limitación de rate) hasta que éste se vacíe. Después que se vacíe recibirá el límite de tráfico asignado.

Se requiere varios conceptos que Squid usa para el control de Delay Pools:

- Listas de control de acceso (Access rules)
- Clases de Delay Pools (Delay pool classes)
- Tipo de cubos (buckets)

2.3.8.3.1. Secuencia lógica:

1. Squid verifica en qué delay_access te encuentras.
2. Si concuerda con una, esta apunta hacia un Delay pool específico.
3. Cada Delay pool tiene una clase: 1 , 2 , 3.
4. La clase determina qué tipo de cubo estás usando. Squid tiene 3 tipos: Global (agregate), individual, red (network).

    - La *clase 1* tiene un único cubo Global (agregate), o sea que todas las solicitudes que entran en ese pool estarán compartiendo este ancho de bando.
    - La clase 2 tiene un cubo Global (agregate) y 256 cubos Individual (por cada usuario).
    - La clase 3 te permite definir un ancho de banda Global (agregate), un límite de 256 cubos por cada subred clase C (/24), y un límite de 65536 cubos por cada usuario accediendo.

    Por razones obvias se toma siempre el ancho de banda menor. Por ejemplo, considere la posibilidad de una clase 3 cubos agregate, network e Individual. Si tiene en Individual 20 KB, En network 30 KB, pero el agregate tiene 2 KB, el cliente recibirá sólo 2-KB.

2.3.8.3.2. Parámetros de configuración:

- **delay_pools:** Define cuantos Delay pools se van a utilizar.
    - delay_pools 5 (Define 5 Delay pools que serán configurados posteriormente).

- **delay_class:** Define la clase del Delay pool. Para evitar complicaciones es recomendable tener siempre un delay_class para cada Delay pool definido.
    - delay_class 1 3 (Define el Delay pool 1 que sea de tipo 3).
    - delay_class 5 2 (Define el Delay pool 5 que sea de tipo 2).

- **delay_parameters:** Este es el parámetro crítico en el cual se limita el ancho de banda. Para cada Delay Pool se debe definir: el fill rate (tráfico de llenado) y el tamaño máximo de cada cubo.
    - delay_parameters N rate/size [*rate/size* [*rate/size*]]

  El valor del rate está definido en bytes por segundo, y size en total de bytes. Si se divide el size entre el rate, Dispondrás el tiempo en segundos que el cubo se llenará si el cliente no está consumiendo.

  Una Pool de clase 1 que dispone de un solo cubo sería definida como sigue:

    - delay_class 2 1
    - delay_parameters 2 2000/8000

  Para un Pools de clase 2, El primer cubo es aggregate, y el segundo es un grupo de cubos individuales:

    - delay_class 4 2
    - delay_parameters 4 7000/15000 3000/4000

  De esta forma la red "toda la red" dispondrá en los primeros 15 Kb (tamaño del cubo) descarga a velocidad máxima sin restricción, después de haber descargado los primeros 15 Kb (se vació el cubo), descargará a 7 KB. Igualmente cada cliente solo podrá descargar rápidamente los primeros 4 Kb, después descargará establemente siempre a 3 Kb.

- **delay_initial_bucket_level:** Dice que tan lleno estará el cada cubo al iniciar el Squid. Se indica en porcentajes
    - delay_initial_bucket_level 75%

- **delay_access:** permite relacionarlo a una ACL específica. Es similar a las reglas de acceso de Squid, pero es necesario definir el número de Pool antes del Allow o Deny.
    - delay_access 1 deny gerentes
    - delay_access 1 allow mi_red
    - delay_access 5 allow mime_extensiones

  Squid define una lista de acceso separada para cada Delay Pool. Puede disponer de un Allow o un Deny. Si es Allow utiliza esa regla y no sigue buscando en las siguientes de ese Pool. Si es Deny, automáticamente cancela la búsqueda en ese Pool, pero seguirá buscando en las listas de acceso de los otros pools.

2.3.9. DESARROLLO DE APLICACIONES CON PHP PARA LA CREACIÓN DE PÁGINAS DINÁMICAS.

2.3.9.1. Introducción a PHP.

PHP es el lenguaje de lado servidor más extendido en la web. Nacido en 1994, se trata de un lenguaje de creación relativamente reciente. Es un lenguaje que ha tenido una gran aceptación en la comunidad de desarrolladores, debido a la potencia y simplicidad que lo caracterizan, así como al soporte generalizado en la mayoría de los servidores de hosting.

PHP nos permite embeber sus pequeños fragmentos de código dentro de la página HTML y realizar determinadas acciones de una forma fácil y eficaz, combinando lo que ya sabemos del desarrollo HTML. Es decir, con PHP escribimos scripts dentro del código HTML. Por otra parte, y es aquí donde reside su mayor interés con respecto a los lenguajes pensados para los CGI, PHP ofrece un sinfín de funciones para la explotación de bases de datos de una manera llana, sin complicaciones.

La principal función del PHP es permitir la interacción de la página web con el visitante que pudo haber realizado cambios en ella, y cada usuario que ingrese a la página podrá ver la información anteriormente modificada.

2.3.9.2. ¿Qué es PHP?

PHP es el acrónimo de Hipertext Preprocesor. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores.

Gráfico Nº 2.9: Esquema del funcionamiento de las páginas PHP.

Una vez que ya conocemos el concepto de lenguaje de programación de scripts del lado del servidor podemos hablar de PHP. PHP se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar, al igual que ocurre con el popular ASP de Microsoft, pero con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Cualquiera puede descargar a través de la página principal de PHP www.php.net y de manera gratuita, un módulo que hace que nuestro servidor web comprenda los scripts realizados en este lenguaje. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo.

PHP, en el caso de estar montado sobre un servidor Linux, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP.

Por último señalábamos la seguridad, en este punto también es importante el hecho de que en muchas ocasiones PHP se encuentra instalado sobre servidores Linux, que son de sobra conocidos como más veloces y seguros que el sistema operativo donde se ejecuta las ASP, Windows NT o 2000. Además, PHP permite configurar el servidor de modo que se permita o rechacen diferentes usos, lo que puede hacer al lenguaje más o menos seguro dependiendo de las necesidades de cada cual.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como PHP está desarrollado en política de código abierto, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores. Actualmente PHP se encuentra en su versión 5.4, que utiliza el motor Zend, desarrollado con mayor meditación para cubrir las necesidades de las aplicaciones web actuales.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red, por poner dos ejemplos.

Algunas de las más importantes capacidades de PHP son: compatibilidad con las bases de datos más comunes, como MySQL, mSQL, Oracle, SQLite, Informix, y ODBC, por ejemplo. Incluye funciones para el envío de correo electrónico, upload de archivos, crear dinámicamente en el servidor imágenes en formato GIF, incluso animadas y una lista interminable de utilidades adicionales.[5]

2.3.9.3. Características únicas de PHP.

- **Rendimiento:** Los scripts escritos en PHP se ejecutan más rápido que los escritos en otros lenguajes de creación de scripts; numerosos estudios comparativos independientes ponen este lenguaje por encima de sus competidores como JSP, ASP.NET y Perl. El motor de PHP 5.0 fue completamente rediseñado con un manejo óptimo de memoria para mejorar su rendimiento y es claramente más veloz que las versiones previas. Además, están disponibles aceleradores de terceros que pueden mejorar aun más el rendimiento y el tiempo de respuesta.

- **Portabilidad:** PHP está disponible para UNIX, Microsoft Windows, Mac OS y OS/2 y los programas escritos en PHP se pueden transportar de una plataforma a otra. Como resultado las aplicaciones PHP desarrolladas en Windows, por ejemplo, se ejecutaran en UNIX sin grandes contratiempos.

- **Fácil de Usar:** PHP es un lenguaje de programación extremadamente sofisticado. Su sintaxis es clara y consistente y viene con una documentación exhaustiva para más de 5000 funciones incluidas en la distribución principal.

[5] ÁLVAREZ Miguel Ángel, Director de Desarrollo Web.com, "¿Qué es PHP?" http://www.desarrolloweb.com/articulos/392.php

• **Código Libre:** PHP es un proyecto de código libre; el lenguaje es desarrollado por un grupo de programadores voluntarios distribuidos por todo el mundo, quienes ponen a disposición gratuita el código fuente a través de internet, y puede ser utilizado sin costo, sin pagos por licencia y sin necesidad de grandes inversiones en equipo de computo ni programas.

• **Soporte Comunitario:** Una de las mejores características de los lenguajes a los que da soporte una comunidad, como PHP, es el acceso que ofrece a la creatividad e imaginación de cientos de desarrolladores ubicados en diferentes partes del mundo.

• **Soporte a aplicaciones a terceros:** Una de las fortalezas históricas de PHP ha sido su soporte a una amplia gama de diferentes bases de datos, entre las cuales se incluyen MySql, Oracle y Microsoft SQL Server. PHP 5.3 soporta más de quince diferentes motores de bases de datos, e incluye una interfaz de programación de aplicaciones común para el acceso a base de datos. El soporte para XML facilita la lectura (y escritura) de documentos XML como si fueran estructuras de datos nativas de PHP.[6]

2.3.9.4. Ventajas y desventajas de PHP.

Ventajas:

PHP presenta múltiples ventajas frente a otros lenguajes de programación que necesariamente harán que este lenguaje se imponga como una alternativa para el desarrollo de todo tipo de aplicaciones.

- Es un lenguaje multiplataforma. Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Capacidad de difundir su potencial utilizando la enorme cantidad de módulos.

[6] VASWANI Vikram, "Fundamentos de PHP", Pág. 5-7.

- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones.

Desventajas:

- No posee una abstracción de base de datos estándar, sino bibliotecas especializadas para cada motor (a veces más de una para el mismo motor).
- No posee adecuado manejo de internacionalización, unicode, etc.
- Por su diseño dinámico no puede ser compilado y es muy difícil de optimizar.
- Por sus características favorece la creación de código desordenado y complejo de mantener.

Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aún estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable.

Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (o MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes.[7]

[7] Ing. TTITO C., Pablo Cesar, "Breve historia de PHP", http://www.monografias.com/trabajos66/introduccion-php/introduccion-php2.shtml.

2.3.9.5. Manejo de archivos en PHP.

Se nos vuelve necesaria la manipulación de archivos, para poder editarlos y configurar así CentOS, a fin de poder establecer la funcionalidad que deseamos del sistema. PHP nos ofrece una serie de funciones que facilitan esta tarea, para poder acceder a archivos, leer los mismos e incluso editarlos, varias de estas funciones son las siguientes, ver Tabla Nº 2.6:

<table>
<tr><th>Comando</th><th>Descripción</th></tr>
<tr><td>copy</td><td>Copia un archivo, su sintaxis es: copy($origen,$destino)</td></tr>
<tr><td>rename</td><td>Cambia el nombre de un archivo de $nombre1 a $nombre2, la sintaxis es: rename($antes,$despues)</td></tr>
<tr><td>unlink</td><td>Borra un archivo, y la sintaxis de esta funciones: unlink($archivo)</td></tr>
<tr><td>fopen</td><td>Abre un archivo y le asigna un identificador id. La sintaxis es la siguiente:
$id = fopen($archivo, $modo)
Donde el modo puede ser uno de los siguientes:
<table>
<tr><th>Comando</th><th>Descripción</th></tr>
<tr><td>‘r’</td><td>Sólo lectura</td></tr>
<tr><td>‘r+’</td><td>Lectura y escritura</td></tr>
<tr><td>‘w’</td><td>Sólo escritura</td></tr>
<tr><td>‘w+’</td><td>Lectura y escritura. Suprime el contenido anterior si se escribe. El archivo es creado si no existe.</td></tr>
<tr><td>‘a’</td><td>Sólo escritura. El archivo es creado si no existe y el puntero se coloca al final.</td></tr>
<tr><td>‘a+’</td><td>Lectura y escritura. El archivo es creado si no existe y el puntero se coloca al final.</td></tr>
</table></td></tr>
<tr><td>fgets</td><td>Lee una línea de un archivo hasta un número máximo de caracteres. La sintaxis es la siguiente: fgets($id,$max)</td></tr>
<tr><td>fwrite</td><td>Escribe una cadena dentro del archivo. La sintaxis es la siguiente: fwrite($id, $cadena)</td></tr>
<tr><td>fseek</td><td>Avanza o retrocede el puntero del archivo un cierto número de posiciones. La sintaxis es la siguiente: fseek($id,$posiciones)</td></tr>
<tr><td>feof</td><td>Comprueba si el puntero que lee el archivo ha llegado al final. La sintaxis es la siguiente: feof($id)</td></tr>
<tr><td>fpassthru</td><td>Lee completamente el archivo y lo muestra. La sintaxis es la siguiente: fpassthru($id)</td></tr>
<tr><td>fclose</td><td>Cierra el archivo abierto previamente. La sintaxis es la siguiente: fclose($id)</td></tr>
</table>

Tabla Nº 2.6: Comandos para manejo de archivos en PHP.

2.3.9.6. Manejo de Comandos Unix en PHP.

PHP, tiene varias funciones que nos permiten ejecutar comandos del Shell de Linux. Pero antes de seguir con la caracterización de las funciones que PHP posee, definiremos rápidamente lo que es un Shell.

Shell.- Es un intérprete de comandos, es un programa especializado para aceptar comandos ingresados por el usuario, traduciendo aquellos en programas para ejecutarse, ejecutando esos programas, y mostrándolos (o haciendo algo) con los resultados.

Diferentes Shells existen, que ofrecen diferentes características. Dos familias de Shell existen:

- Shell Bourn y sus variantes (sh, bash, ksh)
- Shell C y sus variantes (csh, tcsh).

Aunque muchas Shells tienen características comunes a otras, la manera en que las usan es única, de manera que (por ejemplo) las convenciones de la Shell Bourne usualmente no se aplican a las Shells C.

Bash.- es la Shell predeterminada para los usuarios de Linux. Es compatible con la tradicional Shell Bourne (sh), los scripts de hechos en sh funcionarán en bash, aunque hay algunas características específicas a bash que no funcionarán en Shells Bourne más viejas.

Otro de los aspectos importantes que se deben analizar, es que para la ejecución de comandos a través del browser, es que esta tarea es efectuada por el usuario apache. Este usuario no tiene todos los privilegios como si lo tiene "root", por lo que es requerido hablar de "sudo".

Sudo.- Es una herramienta de sistema que permite a los usuarios realizar la ejecución de mandatos como superusuario u otro usuario de acuerdo a como se especifique en el fichero */etc/sudoers*, donde se determina quien está autorizado. Los números de identidad de usuario y de grupo (UID: identificador de usuario y GID: identificador de grupos) reales y efectivas se establecen para igualar a aquellas del usuario objetivo como esté especificado en el fichero */etc/passwd*.

De modo predeterminado sudo requiere que los usuarios se autentiquen así mismos con su propia clave de acceso (nunca la clave de acceso de root). Una vez que el usuario se ha autenticado, el usuario podrá utilizar nuevamente sudo sin necesidad de volver a autenticarse durante 5 minutos, salvo que se especifique lo contrario en el fichero */etc/sudoers*. Si el usuario ejecuta el mandato *sudo -v* podrá refrescar éste periodo de tiempo sin necesidad de tener que ejecutar un mandato, en cuyo caso contrario expirará esta autenticación y será necesario volver a realizarla.

Si un usuario no listado en el fichero */etc/sudoers* trata de ejecutar un mandato a través de sudo, se registra la actividad en la bitácora de sistema (a través de syslogd) y se envía un mensaje de correo electrónico al administrador del sistema (root).

El fichero */etc/sudoers* se edita con el mandato visudo, herramienta que a través de vi permite realizar cambios y verificar sintaxis y errores. Si se trata de modificar directamente */etc/sudoers*, éste tiene permisos de solo lectura.

La sintaxis básica de una regla es:

```
[usuario,%grupo, NOMBRELISTA] [anfitrión] = (id de usuario) mandatos
```

En nuestro caso, queremos darle a apache varios permisos, por lo que tendríamos entonces que realizar el siguiente procedimiento:

- Bajo la sintaxis “defaults requiretty”, añadimos la línea:
  default logfile = /var/log/sudo.log
- Bajo la línea “root ALL = (ALL) ALL”, aumentamos los permisos que le daremos al usuario apache, un caso de ejemplo:
  apache ALL = (root) NOPASSWD: /bin/ls, /sbin/service

Luego de estas modificaciones, el usuario apache, está en capacidad de ejecutar por ejemplo, *ls* y puede iniciar o detener servicios.

Con la noción clara de lo que es un Shell y como darle permisos al usuario apache para que pueda ejecutar comando, veamos cuales son las funciones de PHP que nos permiten interactuar con el bash de Linux.

FUNCIONES PARA EJECUCIÓN DE COMANDOS

exec.- Ejecutar un programa externo. La sintaxis es la siguiente:

```
exec (string comando [, array &salida [, int &var_retorno]])
```

Donde los parámetros son:

- *comando*: El comando que será ejecutado.
- *salida*: Si el argumento salida está presente, entonces la matriz especificada será llenada con cada línea de la salida del comando. El espacio en blanco extra, como \n, no es incluido en esta matriz. Note que si la matriz ya contiene algunos elementos, exec() anexará sus resultados al final de la matriz. Si no desea que la función anexe los elementos, use unset() sobre la matriz antes de pasarla a exec().
- *var_retorno*: Si el argumento *var_retorno* está presente junto con el argumento salida, entonces el status de retorno del comando ejecutado será escrito en esta variable. El valor retornado por la función es la última línea de los resultados del comando.

shell_exec.- Ejecutar un comando mediante el intérprete de comandos y devolver la salida completa como una cadena. La sintaxis es la siguiente:

```
shell_exec (string cmd)
```

Donde el parámetro es:

- *cmd*: El comando que será ejecutado. El valor retornado por la función, será la salida misma del comando.

Con las funciones anteriores, podemos ejecutar comandos generales, pero debido a la necesidad de editar algunos scripts, necesitamos también de funciones que nos permitan manipular archivos. Para este cometido, veremos cuáles son las funciones de PHP que nos permitirán lograr este objetivo:

FUNCIONES PARA MANEJO DE ARCHIVOS

chmod(): Trata de cambiar los permisos del archivo especificado por el parámetro nombre_archivo a los permisos dados por modo. La sintaxis es la siguiente:

```
int chmod (string nombre_archivo, int modo)
```

Donde los parámetros son los siguientes:

- *nombre_archivo*: Con él se especifica el nombre del archivo que se trata de cambiar los permisos.
- *modo*: Consiste de tres componentes de valor octal que especifican las restricciones de acceso para el propietario, el grupo de usuarios al que pertenece el propietario del archivo, y todo el mundo, en ese orden. Uno de los componentes puede ser calculado al agregarle los permisos necesarios para ese usuario en específico, El número 1 significa que tiene permisos de ejecución, el número 2 significa que puede modificar el contenido del archivo, el número 4 significa que puede leer el contenido del archivo. Para asegurar que se hace la operación esperada se necesita anteponer un cero (0) como prefijo del parámetro modo.

fopen ().- Asocia un recurso con nombre, especificado por nombre_archivo, a una secuencia. Si nombre_archivo es de la forma "*esquema://...*", se asume que es una URL y PHP buscará por un gestor de protocolo (también conocido como envoltura) para tal esquema. Si no hay envolturas registradas para ese protocolo, PHP emitirá una noticia para ayudarle a rastrear problemas potenciales en su script, y luego continúa como si nombre_archivo indicara un archivo corriente. La sintaxis de es la siguiente:

```
fopen (string nombre_archivo, string modo [, bool
usar_ruta_inclusion [, resource contexto_z]])
```

Donde los parámetros son:

- *nombre_archivo*: Es el nombre del archivo que se desea abrir, puede ser la ruta de un archivo, o una URL.
- *modo*: Es el tipo de acceso que se tendrá al archivo especificado, puede tomar los siguientes valores:
  - **'r'** Apertura para sólo lectura; ubica el apuntador de archivo al comienzo del mismo.
  - **'r+'** Apertura para lectura y escritura; ubica el apuntador de archivo al comienzo del mismo.
  - **'w'** Apertura para sólo escritura; ubica el apuntador de archivo al comienzo de éste y lo trunca a una longitud de cero. Si el archivo no existe, intenta crearlo.
  - **'w+'** Apertura para lectura y escritura; ubica el apuntador de archivo al comienzo de éste y lo trunca a una longitud cero. Si el archivo no existe, intenta crearlo.
  - **'a'** Apertura para sólo escritura; ubica el apuntador de archivo al final del mismo. Si el archivo no existe, intenta crearlo.
  - **'a+'** Apertura para lectura y escritura; ubica el apuntador de archivo al final del mismo. Si el archivo no existe, intenta crearlo.
  - **'x'** Creación y apertura para sólo escritura; ubica el apuntador de archivo al comienzo de éste. Si el archivo ya existe, fopen() fallará devolviendo FALSE y generando un error de nivel E_WARNING.
  - **'x+'** Creación y apertura para lectura y escritura; ubica el apuntador de archivo al comienzo de éste. Si el archivo ya existe, fopen() fallará devolviendo FALSE y generando un error de nivel E_WARNING.
- *usar_ruta_inclusion*: puede definirse como '1' o TRUE si desea buscar por el archivo en include_path, también.

fwrite().-Escritura sobre archivos, segura con material binario. La sintaxis es la siguiente:

```
int fwrite (resource gestor, string cadena [, int longitud])
```

Donde los parámetros son:

- *gestor*: Es la fuente que contiene el archivo en el que se escribirá, es un almacén de datos.
- *cadena*: Es el contenido que se va a añadir en el archivo señalado por gestor.
- *longitud*: Cuando el argumento longitud es entregado, la escritura se detendrá después de que longitud bytes hayan sido escritos, o al alcanzar el final de cadena, aquello que ocurra primero.

2.3.9.7. Bases de datos.

Se define una base de datos como una serie de datos pertenecientes a un mismo contexto, organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular. Existen programas denominados sistema gestor de bases de datos, abreviado SGBD, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada.

2.3.9.7.1. Características de las bases de datos.

Entre las principales características de los sistemas de base de datos podemos mencionar:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.

- Consultas complejas optimizadas.
- Seguridad de acceso y auditoría.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

2.3.9.7.2. Ventajas de las Bases de Datos frente a los sistemas de ficheros.

Control sobre la redundancia de datos: Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. En los sistemas de bases de datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos.

Consistencia de datos: Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias.

Compartición de datos: En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados.

Mantenimiento de estándares: Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales.

Mejora en la integridad de datos: La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar.

Mejora en la seguridad: La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de

seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros.

Mejora en la accesibilidad a los datos: Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.

Mejora en la productividad: El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación.

Mejora en el mantenimiento: En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.

Aumento de la concurrencia: En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.

Mejora en los servicios de copias de seguridad: Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos. Sin embargo, los SGBD actuales funcionan de modo que se minimiza la cantidad de trabajo perdido cuando se produce un fallo.

2.3.9.7.3. Desventajas de las Bases de Datos.

Complejidad: Los SGBD son conjuntos de programas que pueden llegar a ser complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder realizar un buen uso de ellos.

Coste del equipamiento adicional: Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina que se dedique solamente al SGBD.

Vulnerable a los fallos: El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse. Es por ello que deben tenerse copias de seguridad (Backup).

2.3.9.7.4. Base de datos SQLite en PHP.

SQLite es una base de datos rápida y eficiente que ofrece una opción viable a MySQL, sobre todo para aplicaciones pequeñas y de tamaño mediano. Sin embargo, a diferencia de MySQL, que contiene una gran cantidad de componentes interrelacionados, SQLite está integrado en una sola biblioteca. También es significativamente más pequeña que MySQL, su línea de comandos pesa menos de 200 KB, y soporta todos los comandos SQL estándar con los que estamos familiarizados. MySQL y SQLite también difieren en sus políticas de licencia: a diferencia de MySQL, el código fuente de SQLite es completamente de dominio público, lo cual significa que los desarrolladores pueden utilizarlo y distribuirlo como les plazca.

Sin embargo el tamaño de SQLite disfraza su poder. La base de datos tiene una capacidad de soportar dos terabytes y puede tener un mejor rendimiento que MySQL en ciertas situaciones. En parte, esto se debe a razones estructurales: SQLite lee y escribe registros directamente en el disco y por lo mismo ocupa menos recursos

que MySQL, que opera sobre una arquitectura cliente – servidor que puede ser afectada por variables relacionada con el nivel de red.[8]

El motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host.

2.3.9.7.5. Extensión PDO para acceder a base de datos.

La extensión PHP Data Objects (PDO) define una interfaz ligera, para tener acceso a bases de datos en PHP. Cada controlador de base de datos que implementa la interfaz PDO puede exponer base de datos específicas como funciones de extensión regular. Tenga en cuenta que no puede realizar las funciones de base de datos utilizando la extensión PDO por sí mismo, debe utilizar un controlador PDO de base de datos específica para tener acceso a un servidor de base de datos.

PDO proporciona una capa de abstracción acceso a datos, que significa que, independientemente de la base de datos que está utilizando, se utiliza las mismas funciones para realizar consultas y obtener datos.

Funciones de SQLite PDO_SQLITE es un controlador que implementa la interfaz Objetos de Datos de PHP (PDO, siglas en inglés de PHP Data Objects) para habilitar el acceso a bases de datos de SQLite.

[8] VASWANI Vikram, "Fundamentos de PHP", Pág. 216-217

2.3.10. Gestor web Apache.

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo.

Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red, porque está diseñado para ser un servidor web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. Las diferentes plataformas y entornos, hacen que a menudo sean necesarias diferentes características o funcionalidades. Apache se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular. Este diseño permite a los administradores de sitios web elegir que características van a ser incluidas en el servidor seleccionando que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor.

2.3.10.1. Ventajas del servidor Apache.

- Modular
- Módulos de autenticación: mod_access, mod_auth y mod_digest.
- Open Source
- Multi-plataforma. Puede soportar de una forma más fácil y eficiente una amplia variedad de sistemas operativos.
- Extensible
- Popular (fácil conseguir ayuda/soporte) y Gratuito
- El servidor puede tipificarse mejor para las necesidades de cada sitio web.

- Soporte para los lenguajes perl, python, tcl y PHP.
- Soporte para SSL y TLS.
- Permite la configuración de mensajes de errores personalizados y negociación de contenido.

2.3.10.2. Instalación y configuración de Apache.

La instalación del servidor web apache es relativamente sencilla, solo debe teclear en terminal el siguiente comando como usuario root.

```
[root@servidorCentOS ~]# yum –y install httpd
```

La configuración del servidor web apache se realizara sobre dos ficheros distintos, uno de configuración general del servidor web apache y otro para indicarle al servidor apache los dominios virtuales que deben ser cargados al sistema. El fichero de configuración principal de apache lo encontramos en la siguiente ruta:

```
/etc/httpd/conf/
```

La carpeta donde deberán ser añadidos los ficheros de configuración de los dominios virtuales será en la siguiente ruta:

```
/etc/httpd/conf.d/
```

Cuando se instala Apache, se copia el archivo "httpd.conf", que está compuesto por tres bloques fundamentales, que bloques son:

- Parámetros globales
- Directivas de funcionamiento
- Hosts virtuales

2.3.10.2.1. Parámetros globales.

Algunos parámetros son propósito general, y otros son configurables de forma independiente para cada conjunto de directorios o de ficheros o incluso para un servidor virtual específico. En tales casos, estos parámetros se encuentran dentro de secciones en las que se indica el contexto de aplicación de dicho parámetro. Las secciones fundamentales son, ver Tabla Nº 2.7:

Parámetro	Descripción
<Directory>	Los parámetros que se encuentran dentro de la sección Directory sólo se aplican al directorio indicado y sus subdirectorios.
<DirectoryMatch>	Igual que Directory, aunque acepta expresiones regulares en el nombre del directorio.
<Files>	Los parámetros de configuración facilitan control de acceso a los ficheros mediante su nombre.
<FilesMatch>	Igual que Files, pero acepta en el nombre del fichero expresiones regulares.
<VirtualHost>	Los parámetros sólo se aplican a aquellas peticiones dirigidas a este host (nombre de servidor, dirección IP o puerto TCP).
<Proxy>	Sólo se aplican estos parámetros a aquellas peticiones de proxy (requiere que esté instalado "mod proxy") coincidentes con la especificación de URL.
<ProxyMatch>	Igual que proxy, pero acepta en la URL indicada el uso de expresiones regulares.

Tabla Nº 2.7: Parámetros globales de configuración de APACHE.

2.3.10.2.2. Directivas globales de configuración.

Algunas directivas de configuración nunca se aplican a las secciones antes mencionadas (directorios, etc.), sino que afectan al conjunto del servidor web. Las más destacables son, ver Tabla Nº 2.8:

Directiva	Descripción
ServerRoot	Especifica la localización del directorio raíz en el que se encuentra instalado el servidor web. Partiendo de este directorio, se encuentran los ficheros de configuración, etc.
Timeout	Número de segundos tras los cuales el servidor cierra la conexión.
KeepAlive	Especifica si se deben utilizar conexiones persistentes para atender las peticiones de un mismo usuario mediante la misma conexión TCP.
MaxKeepAliveRequest	El número máximo de peticiones permitidas durante una conexión persistente.
MinSpareServers	Define el número de procesos servidores hijo desocupados que no atienden una petición.
MaxSpareServers	Define el nº máximo de procesos servidor hijo desocupados que no manejan una petición. Si existieran más de lo que define la directiva, el proceso padre mataría los procesos que exceden.

StartServers	Número de procesos servidor hijo que serán creados cuando arranca Apache por primera vez.
Maxclients	Define el número de peticiones simultáneas que apache puede soportar. Como máximo se crean este nº de procesos servidores hijo.
MaxRequestPerChild	Define el nº de peticiones que cada proceso hijo tiene permitido procesar antes de morir.
Listen	Especifica el puerto en que se atenderán las peticiones. Por defecto el servidor "escucha" en el puerto 80 de TCP. Permite especificar las direcciones IP que se utilizarán (en caso de que el servidor tuviese más de una). Por defecto se utilizarán todas las disponibles.

Tabla Nº 2.8: Directivas globales de configuración de APACHE.

2.3.10.2.3. Directivas principales.

Hay algunas directivas que, generalmente, no suelen aparecer en las secciones anteriormente mencionadas (algunas de ellas no deben estar en ninguna sección, y es obligatorio que aparezcan en la sección principal), sino que se encuentran en la sección principal, ver Tabla Nº 2.8.

Directiva	Descripción
Port	Define el puerto en el cual escucha el servidor (0 - 65535). Hay que tener en cuenta la relación que tiene esta directiva con el fichero */etc/services* y que algunos puertos, especialmente los situados por debajo del 1024, están reservados para protocolos específicos. El puerto estándar para el protocolo HTTP es el 80. En nuestro caso usamos el 9870.
User/Group	Definen el usuario y el grupo con el que el servidor contestará las peticiones. Para poder utilizar esta directiva, el servidor standalone debe ejecutarse inicialmente como usuario root. El usuario no debería poseer privilegios que otorguen acceso a ficheros que no deseemos.
ServerAdmin	Especifica la dirección de correo electrónico del administrador. Esta dirección puede mostrarse en los mensajes de error a modo de dirección de contacto para que los usuarios notifiquen el error al administrador. No debe estar dentro de ninguna sección.
ServerName	Sirve para especificar el nombre y el puerto TCP que el Apache utiliza para identificarse. Se puede determinar de forma automática, pero se recomienda especificarlo. Si el servidor no tuviera un nombre DNS, es recomendable incluir su dirección IP. No debe incluirse dentro de ninguna sección. Su sintaxis es: ***ServerName nombre direccion: puerto*** como en: ***ServerName 192.168.10.1:9870***
DocumentRoot	Directorio raíz desde el cual se servirán los documentos. Por defecto es "htdocs", dentro de la carpeta de instalación de Apache. No debe aparecer dentro de ninguna sección, a excepción de la sección VirtualHost. Le corresponde una sección <Directory> en la cual se marcan los parámetros de configuración de este directorio.
Directory	<Directory></Directory> se utilizan para encerrar un grupo de directivas que se aplicarán al directorio en cuestión y sus sub-directorios. El parámetro directorio, puede ser una trayectoria completa o un meta carácter.

AllowOverride override	Cuando el servidor encuentra un fichero .htaccess (definido por la directiva AccessFileName) necesita conocer que directivas declaradas en este fichero pueden sobre escribir información de acceso. El parámetro override puede ser definido a None y en tal caso el servidor no leerá el fichero o puede ser definido a All, de forma que permitirá todas las directivas. El parámetro override también puede ser definido a: AuthConfig, FileInfo, Indexes, Limit, Options.
UserDir	Permite indicar a Apache que un subdirectorio dentro del directorio de trabajo de los diferentes usuarios del sistema sirva para que estos almacenen su página personal. Por ejemplo: ***UserDir publico*** hará que la página almacenada en el directorio del usuario "test", dentro del subdirectorio "público", sea accesible como: http://www.algo.com/~test/indice.html
DirectoryIndex	Especifica el fichero que Apache servirá por defecto para cada directorio en caso de que no se especifique ningún fichero concreto en la URL de la petición. Por defecto es "index.html". Es decir, si se solicita en la barra de direcciones del navegador: *www.algo.com* el servidor enviará por defecto *www.algo.com/index.html*. Es posible especificar más de un fichero y el orden con que se especifican los ficheros determinará la prioridad para determinar cuál se debe servir. Es posible encontrar la directiva fuera de cualquier sección o dentro de alguna de ellas.
Alias	Las directivas Alias y AliasMatch permiten la definición de accesos a directorios que están fuera del DocumentRoot. Su sintaxis es: Alias url directorio. Por ejemplo: ***Alias /docs /home/documentos*** Hará que una petición a http://www.algo.com/docs/manual se sirva desde /home/documentos/manual.
Location url	La directiva proporciona control de acceso por URL. Es similar a la directiva Directory.

Tabla Nº 2.9: Directivas principales de configuración de APACHE.

Cuando */etc/httpd/conf/httpd.conf* se encuentre configurado de modo tal que httpd escuche sobre puerto diferente a los puertos TCP predeterminados (80, 443, 488, 8008, 8009, o 8443), debe utilizarse el comando semanage port para agregar el nuevo número de puerto. Si */etc/httpd/conf/httpd.conf* está configurado para que httpd pueda escuchar sobre cualquier otro puerto no listado en http_port_t, entonces httpd no podrá iniciarse. En nuestro proyecto el puerto de escucha de la aplicación es el 9870, y lo agregamos a la lista de puertos validos con el siguiente comando ejecutado desde un terminal como usuario root:

```
[root@servidorCentOS ~]# semanage port -a -t http_port_t -p tcp 9870
```

2.3.10.2.4. Directivas de sección.

Casi todas las secciones de localización (Directory, Location, etc.) incluyen una serie de directivas en su configuración que permiten controlar el acceso al contenido, ver Tabla Nº 2.9.

Directiva	Descripción
Allow	Permite especificar quién tiene autorización para acceder a un recurso. Se pueden especificar direcciones IP, nombres de máquina, fragmentos del nombre o de la dirección o variables de la petición. Existe la palabra clave "all" que indica "todos los clientes".
Deny	Permite especificar a quién no permitimos el acceso a un recurso. Cuenta con las mismas opciones que Allow.
Order	Permite afinar el funcionamiento de las anteriores directivas: Allow y Deny. Existen 2 opciones: **Allow, Deny**. Por defecto se deniega el acceso y sólo podrán acceder aquellos clientes que cumplan las especificaciones de Allow y en cambio no cumplan las especificaciones de Deny. **Deny, Allow**. Por defecto se permite el acceso y sólo podrán entrar los clientes que no cumplan las especificaciones de Deny o sí cumplan las especificaciones de Allow.

Tabla Nº 2.10: Directivas de sección para la configuración de APACHE.

2.3.10.3. Servidores Virtuales.

Se conoce como servidor virtual a una partición dentro de un servidor que habilita varias máquinas virtuales dentro de dicha máquina por medio de varias tecnologías. Apache permite servir varios sitios web con un único servidor. Para ello permite la creación de dominios virtuales en función de diferentes direcciones IP o diferentes nombres por IP. Apache fue de los primeros servidores que soportó servidores virtuales sin necesidad de distinguir por IP, sino en función de nombre.

Esta capacidad simplifica enormemente la administración de los servidores, y supone un ahorro de direcciones IP, que normalmente son escasas. Los servidores virtuales que distinguen en función del nombre son perfectamente transparentes para el cliente, con la posible excepción de aquellos navegadores muy antiguos que no envíen la cabecera "Host:" con cada petición.

2.4. HIPÓTESIS Y VARIABLES.

2.4.1. HIPÓTESIS.

Con la implementación de una aplicación tipo web facilitaremos la configuración de un servidor proxy Squid para la administración del Internet en la Biblioteca Virtual de la Facultad de Administración, Finanzas e Informática.

2.4.2. VARIABLES.

2.4.2.1. Variable Independiente.

Implementación de una aplicación tipo web.

2.4.2.2. Variable Dependiente.

Facilitar la configuración del servidor proxy Squid.