Proyecto de Sistemas Informáticos

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

Memoria del proyecto de SSII TunnelBroker.

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

Índice de Ilustraciones

Resumen en castellano del proyecto

Debido al creciente uso que la humanidad hace de Internet y al desaprovechamiento de las direcciones IPv4 en los comienzos de la tecnología, dentro de pocos años se agotarán las 4294967296 direcciones con las que cuenta el protocolo. Para solucionar este problema, se ha ideado una nueva versión del protocolo con un espacio de direcciones muchísimo mayor, para hacernos una idea, con la nueva versión habrá 667134927874467426204 direcciones por metro cuadrado de la tierra.

Por desgracia, el nuevo protocolo no se está adoptando con la rapidez que sería deseable, ya que las grandes compañías de telecomunicaciones son reacias a cambiar un protocolo que funciona perfectamente.

Algunas empresas o usuarios optan por desplegar su red local utilizando la nueva versión del protocolo (IPv6) encontrándose con un serio problema si intentan comunicarse con el exterior de la red local.

La aplicación que se ha desarrollado en el marco de este proyecto, posibilita la conexión entre una isla IPv6 y la troncal (Backbone) utilizando la actual infraestructura IPv4, para ello, la aplicación crea un túnel entre una máquina de una isla y una máquina con conexión a la troncal de forma que el emisor encapsula los datagramas IPv6 dentro de datagramas IPv4 y el receptor los desencapsula, y si fuera necesario los reenvía al destino final.

English summary of the project

Due to the increasing use of the Internet by the humanity and the waste of IPv4 addresses in the early stages of the technology, in a few years, the 4294967296 addresses that the protocol dispose, will exhaust. In order to solve this problem, a new version of the protocol with a much greater space of addresses has been developed, to get the picture, with the new version there are going to be 667134927874467426204 addresses by square meter of the Earth.

Unfortunately, the new protocol is not being adopted with the rapidity that would be desirable, because the great companies of telecommunications are reluctant to change a protocol that works perfectly.

Some companies or users choose to unfold their local network using the new version of the protocol (IPv6) coming up against serious problems if they try to communicate outside of the local network.

The application that has been developed in the final career project, makes possible the connection of an IPv6 island and the Backbone using the present IPv4 infrastructure, for doing so, the application creates a tunnel between a machine of the island and a machine with a connection to the Backbone so that the emitter wraps the IPv6 datagrams within IPv4 datagrams and the receiver unwraps it, and if it's necessary, the receiver resends the IPv6 datagram to the final destiny.

Palabras clave

IPv6

IPv4 dinámica

TCP

Túneles SIT

Túnel broker

TunnelBrokerClient

TunnelBrokerRouter

TunnelBrokerServer

SSL

OpenSSL

Asignación de prefijos IPv6

User mode linux (UML)

Joomla!1.5.x

MySQL 5.0

Introducción al proyecto.

La necesidad de realizar el proyecto TunnelBroker surge de la propia evolución de internet. Debido al crecimiento exponencial que vive la red desde hace algunos años, se ha producido una escasez de direcciones IPv4 y previsiblemente, en unos pocos años, se agotarán por completo. Para solucionar el futuro problema, se ha creado una nueva versión del protocolo IP, el cual ha sido diseñado para resolver dicho problema de forma permanente. El protocolo adoptado es la versión 6 del Protocolo de Internet (IPv6) que, además de solucionar el problema de escasez de direcciones, mejora notablemente otras características de la versión 4.

Pero esta adopción ha traído también nuevos problemas: no existe compatibilidad entre ambas versiones y esa falta de compatibilidad ha hecho que la adopción de forma masiva del protocolo IPv6 por parte de las compañías proveedoras de internet se esté viendo muy ralentizada pese a la urgencia de la misma.

TunnelBroker es un sistema complejo que abarca campos muy diversos dentro del área de la informática y las telecomunicaciones.

Su principal objetivo es la configuración dinámica de túneles sobre una infraestructura IPv4 para comunicar redes locales que con una infraestructura IPv6.

El usuario que se registre y utilice el sistema TunnelBroker podrá configurar una isla IPv6 para que se mantenga conectada permanentemente al Backbone de IPv6. La interfaz web del sistema será el primer paso para comenzar a utilizar el mismo. Los usuarios deberán registrarse, y tras ello tendrán acceso a solicitar un prefijo de direcciones IPv6 con el que podrán configurar su isla. Después, las distintas aplicaciones TunnelBroker le ayudarán a conectar su isla IPv6 con el exterior.

El sistema TunnelBroker, está formado por tres programas y una interfaz web.

La aplicación TunnelBrokerClient permitirá configurar un terminal de la isla IPv6 para que a través de Internet IPv4 mantenga creado constantemente un túnel SIT con un encaminador del sistema que le de acceso al Backbone de IPv6. Dicho encaminador llevará instalada y configurada la aplicación TunnelBrokerRouter y serán los propios administradores del TunnelBroker los encargados de mantenerlos.

El sistema mantendrá monitorizado el túnel entre el cliente que sirve de pasarela a la isla y el encaminador que ofrece acceso al Backbone. Esta monitorización permitirá rehacer el túnel SIT en caso de que por algún motivo éste deje de funcionar (cambios de IPv4 en el cliente, errores en el proceso de creación del túnel, etc.).

Para mantener la monitorización y la gestión del sistema, se ha creado la aplicación TunnelBrokerServer la cual irá instalada en un servidor accesible a través de Internet IPv4. Esta aplicación gestiona los datos del sistema almacenados en una base de datos y comunica TBClients y TBRouters. TBServer es el encargado de asignar los encaminadores a los clientes que solicitan la creación de un nuevo túnel hacia el Backbone en función de la carga de clientes que estén

soportando en el momento de la petición.

TunnelBroker es, en definitiva, un granito de arena para promover el uso del Backbone de IPv6 y la temprana transición hacia la adopción masiva de dicho protocolo.

El presente documento detalla la estructura y desarrollo del proyecto Tunnel Broker.

El desarrollo de Tunnel Broker se ha realizado a lo largo del curso 2008-2009 por un equipo compuesto por tres personas bajo la tutela del director de proyecto acordado durante la prematrícula de la asignatura. Se han mantenido tutorías de forma continuada a lo largo de todo el curso tanto con el director del proyecto como a nivel interno para la coordinación del equipo. Estas reuniones comenzaron quincenales con el director del proyecto, pasando a ser semanales en los meses previos a la entrega. Las reuniones del equipo de trabajo se han mantenido semanalmente durante todo el desarrollo. El nivel de producción se ha mantenido constante durante todo el año por todo el equipo, excepto en febrero, que se redujo debido a los exámenes, y los dos últimos meses del curso académico, que aumentó ligeramente.

El proyecto está enmarcado en la asignatura Sistemas Informáticos de quinto curso de la carrera Ingeniería Informática de la Universidad Complutense de Madrid.

En las siguientes páginas se irán introduciendo y explicando:

- Los distintos componentes del sistema.
- Los motivos por los cuales han sido escogidas las tecnologías que utilizan estos.
- Si se ha planteado el uso de alguna otra alternativa y los motivos por los cuales ha sido rechazada.

Términos y abreviaturas.

UML: User-Mode-Linux.

IP: Internet Protocol.

TCP: Transmission Control Protocol.

MTU: Minimun Transfer Unit.

DHCP: Dynamic Host Configuration Protocol.

DNS: Domain Name System.

SSL: Secure Sockets Layer.

SQL: Structured Query Language.

Entorno del sistema

Máquinas virtuales, primeras opciones y estudios de rendimiento.

Para el desarrollo de este proyecto, no se disponía de una infraestructura amplia con máquinas físicas así que se decidió por unanimidad que se realizaría con máquinas virtuales.

Una máquina virtual es un software que emula a un ordenador y puede ejecutar programas como si fuese un ordenador físico.

Una característica esencial de las máquinas virtuales es que los procesos que ejecutan están limitados por los recursos y abstracciones proporcionados por ellas. Estos procesos no pueden escaparse de este "ordenador virtual".

Las máquinas virtuales se pueden clasificar en dos grandes categorías según su funcionalidad y su grado de equivalencia a una verdadera máquina.

- Máquinas virtuales de sistema (System Virtual Machine)
- Máquinas virtuales de proceso (Process Virtual Machine)

En el caso de este proyecto, el tipo de máquina utilizado es una máquina virtual de sistema. Este tipo de máquina virtual, también llamadas máquinas virtuales de hardware, permiten a la máquina física subyacente multiplexarse entre varias máquinas virtuales, cada una ejecutando su propio sistema operativo. A la capa de software que permite la virtualización se la llama monitor de máquina virtual o "hypervisor". Un monitor de máquina virtual puede ejecutarse o bien directamente sobre el hardware o bien sobre un sistema operativo ("host operating system").

A continuación se muestra un esquema del funcionamiento de una de las máquinas virtuales más utilizadas (VMWare):

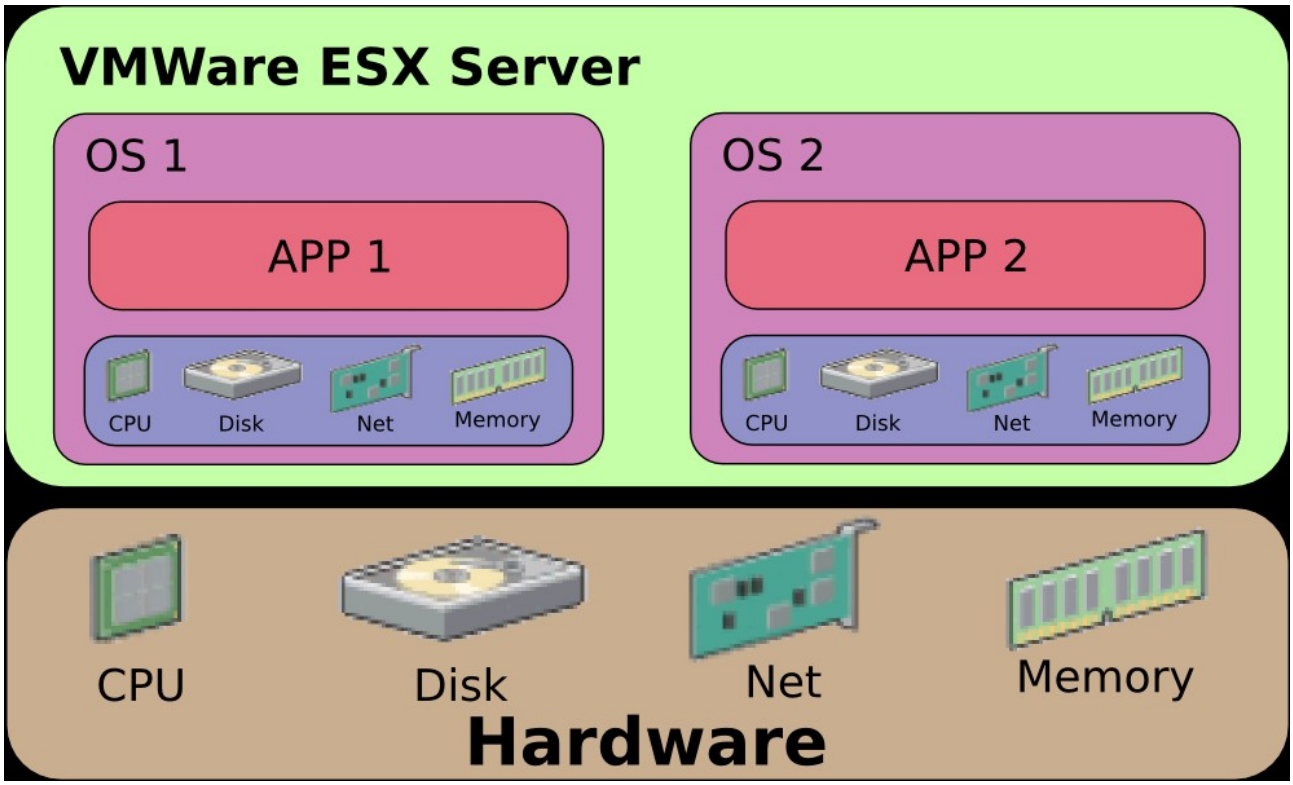


Imagen 1: Esquema de funcionamiento de VMWare

El uso de este tipo de máquina virtual hubiera y ha permitido entre otras cosas:

- Varios sistemas operativos distintos pueden coexistir sobre el mismo ordenador.
- La máquina virtual puede proporcionar una arquitectura de instrucciones (ISA) que sea algo distinta de la de la verdadera máquina. Es decir, se ha podido simular hardware.
- Varias máquinas virtuales (cada una con su propio sistema operativo llamado sistema operativo "invitado" o "guest"), se han utilizado para consolidar servidores.
- Dado que en la mayoría de los casos las máquinas para el desarrollo estaban siendo "sub-utilizadas" (gran capacidad en disco, memoria ram , en la mayoría de los casos se utiliza entre 30% a 60% de su capacidad). Al virtualizar la necesidad de nuevas máquinas físicas se omite y permite un ahorro en costes como energía, mantenimiento, espacio, etc.

Entre las diferentes máquinas virtuales a las que se podían acceder para su uso durante el desarrollo estuvieron las siguientes:

- User-Mode-Linux.
- VMWare.
- VirtualPC.

- VirtualBox.

Después de estudiar sus características y probar la mayoría de ellas se llegó a las siguientes conclusiones.

VirtualPC emula una plataforma x86, Vmware la virtualiza, de forma que la mayor parte de las instrucciones en VMware se ejecutan directamente sobre el hardware físico, mientras que en el caso de Virtual PC se traducen en llamadas al sistema operativo que se ejecuta en el sistema físico. VMware permite ejecutar (simular) varios ordenadores (sistemas operativos) dentro de un mismo hardware de manera simultánea, permitiendo así el mayor aprovechamiento de recursos. No obstante, y al ser una capa intermedia entre el sistema físico y el sistema operativo que funciona en el hardware emulado, la velocidad de ejecución de este último es menor.

Esto se percibió a la hora de aumentar el número de máquinas virtuales corriendo sobre una misma máquina física, ya que se ralentizaba notablemente.

En cuanto a VirtualPC, llegó un momento en el que se descartó ya que debía correr sobre un Windows y esto suponía añadir una nueva"capa" entre varias máquinas.

Comparando VirtualBox con otras aplicaciones privadas de virtualizacion, como VMware Workstation o Microsoft Virtual PC, VirtualBox carece de algunas funcionalidades, pero provee de otras como la ejecución de máquinas virtuales de forma remota, por medio del Remote Desktop Protocol (RDP), soporte iSCSI. En cuanto a la emulación de hardware, los discos duros de los sistemas invitados son almacenados en los sistemas anfitriones como archivos individuales en un contenedor llamado Virtual Disk Image, incompatible con los demás software de virtualización. De este modo, también se descartó VirtualBox.

Finalmente, se probó con User-Mode-Linux que destacó por su "ligereza" a la hora de correr múltiples máquinas virtuales, llegando a hacer pruebas con seis máquinas sin problemas.

Las decisión final sobre el uso de Vmware o User-Mode-Linux fue clara, User-Mode-Linux.

User Mode Linux.

De aquí en adelante UML. Para el desarrollo de las aplicaciones que exige el proyecto, se ha montado un sistema de máquinas virtuales sobre UML. Sobre ellas, se han hecho todas las pruebas como si de máquinas físicas se tratase.

User-Mode-Linux, instalado sobre un Linux, es un método seguro para ejecutar versiones de Linux y sus procesos. Ejecutar software que puede contener errores, experimentar con nuevos núcleos y en la parte interna de Linux, todo esto, sin arriesgar la instalación principal de Linux.

User-Mode-Linux proporciona una máquina virtual que puede tener más recursos hardware y software virtuales que la máquina actual, la máquina física. El almacenamiento en disco para la máquina virtual está contenido completamente en un único fichero en la máquina física. Se puede asignar a la máquina virtual el acceso al hardware que solamente se desee. Con los permisos

correctamente configurados, no se puede hacer nada en la máquina virtual que pueda dañar a la máquina física real o a su software.

Algunas de las aplicaciones para las que es utilizado UML son:

- Alojamiento de servidores virtuales.
- Núcleo de desarrollo.
- Experimentos con nuevos kernels y distribuciones.
- Educación.
- Sandbox: en seguridad computacional, sandbox es un mecanismo de seguridad para separar la ejecución de programas. Con frecuencia, es utilizado para la ejecución de código no probado, no confiable o no verificado de los programas de terceros, proveedores o usuarios no confiables.

Al iniciar UML, se creará un fichero cuyo tamaño es el mismo que el tamaño de disco de UML. Un problema que puede surgir al utilizar varias máquinas virtuales que utilizan la misma imagen de disco, es la duplicación innecesaria de datos en la memoria de la máquina física. Si se tienen N UMLs arrancadas desde imágenes similares, los datos de la misma imagen están cargadas 2 * N veces en la memoria de la máquina física, N veces en la caché de la máquina física y N veces en la página de cachés de UML.

Arrancar estas UMLs desde la misma imagen con ficheros COW reducirá el número de copias en la página de caché del host a 1, pero seguirá habiendo N copias en la página de caché de UML, una por cada UML.

Root_fs.

Es el fichero que contiene el sistema de ficheros de la imagen de Linux que se ejecutará en cada una de las máquinas virtuales. Los ficheros COW hacen referencia a éste fichero, que contiene las características comunes a todas las máquinas UML.

Si éste fichero es modificado tras crear varios COW, se deberán volver a crear de nuevo.

Ficheros COW.

Estos ficheros contienen las particularidades de cada uno de los UML.

Swap.

Es el fichero de intercambio perteneciente a cada una de las máquinas virtuales. Éste no puede ser compartido y deberá existir uno por cada máquina virtual.

Internet Protocol.

El Protocolo de Internet es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.

Los datos en una red basada en IP son enviados en bloques conocidos como paquetes o datagramas (en el protocolo IP estos términos se suelen usar indistintamente). En particular, en IP no se necesita ninguna configuración antes de que un equipo intente enviar paquetes a otro con el que no se había comunicado antes.

El Protocolo de Internet provee un servicio de datagramas no fiable. IP no provee ningún mecanismo para determinar si un paquete alcanza o no su destino y únicamente proporciona seguridad (mediante checksums o sumas de comprobación) de sus cabeceras y no de los datos transmitidos. Si se necesita fiabilidad, ésta es proporcionada por los protocolos de la capa de transporte, como TCP.

Si la información a transmitir ("datagramas") supera la unidad de tamaño máximo (MTU, que viene fijada por la tecnología) en el tramo de red por el que va a circular podrá ser dividida en paquetes más pequeños, y reensamblada luego cuando sea necesario. Estos fragmentos podrán ir cada uno por un camino diferente dependiendo de como estén de congestionadas las rutas en cada momento.

Las cabeceras IP contienen las direcciones de las máquinas de origen y destino (direcciones IP), direcciones que serán usadas por los conmutadores de paquetes (switches) y los encaminadores (routers) para decidir el tramo de red por el que reenviarán los paquetes.

El IP es el elemento común en la Internet de hoy. El actual y más popular protocolo de red es IPv4. IPv6 es el sucesor propuesto de IPv4; poco a poco Internet está agotando las direcciones disponibles por lo que IPv6 utiliza direcciones de fuente y destino de 128 bits (lo cual asigna a cada milímetro cuadrado de la superficie de la Tierra la colosal cifra de 670.000 millones de direcciones IP), muchas más direcciones que las que provee IPv4 con 32 bits.

Las versiones de la 0 a la 3 están reservadas o no fueron usadas. La versión 5 fue usada para un protocolo experimental. Otros números han sido asignados, usualmente para protocolos experimentales, pero no han sido muy extendidos.

Dirección IP.

Una dirección IP es un número que identifica de manera lógica y jerárquicamente a una interfaz de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo de Internet, que corresponde al nivel de red o nivel 3 del modelo de referencia OSI. Dicho número no se ha de confundir con la dirección MAC que es un número físico que es asignado a la tarjeta o dispositivo de red (viene impuesta por el fabricante), mientras que la dirección IP se puede cambiar.

Es obligatorio que un usuario que se conecta desde su hogar a Internet utilice una dirección IP. Esta dirección puede cambiar al reconectar. A ésta forma de asignación de la dirección IP se denomina

dirección IP dinámica (normalmente se abrevia como IP dinámica).

Los sitios de Internet que por su naturaleza necesitan estar permanentemente conectados, generalmente tienen una dirección IP fija (se aplica la misma reducción por IP fija o IP estática), es decir, no cambia con el tiempo. Los servidores de correo, dns, ftp públicos, servidores web necesariamente deben contar con una dirección IP fija o estática, ya que de esta forma se facilita su ubicación.

Existen protocolos para asignar direcciones IP dinámicas, como por ejemplo DHCP (Dynamic Host Configuration Protocol).

IPv4 (RFC0791)

IPv4 es la versión 4 del Protocolo IP (Internet Protocol).

IPv4 usa direcciones de 32 bits, limitándola a 2^{32} = 4.294.967.296 direcciones únicas, muchas de las cuales están dedicadas a redes locales (LANs). Por el crecimiento enorme que ha tenido Internet, combinado con el hecho de que hay desperdicio de direcciones en muchos casos, ya hace varios años se vio que escaseaban las direcciones IPv4.

Esta limitación ayudó a estimular el impulso hacia IPv6, que esta actualmente en las primeras fases de implantación, y se espera que termine reemplazando a IPv4.

Cabecera IPv4.

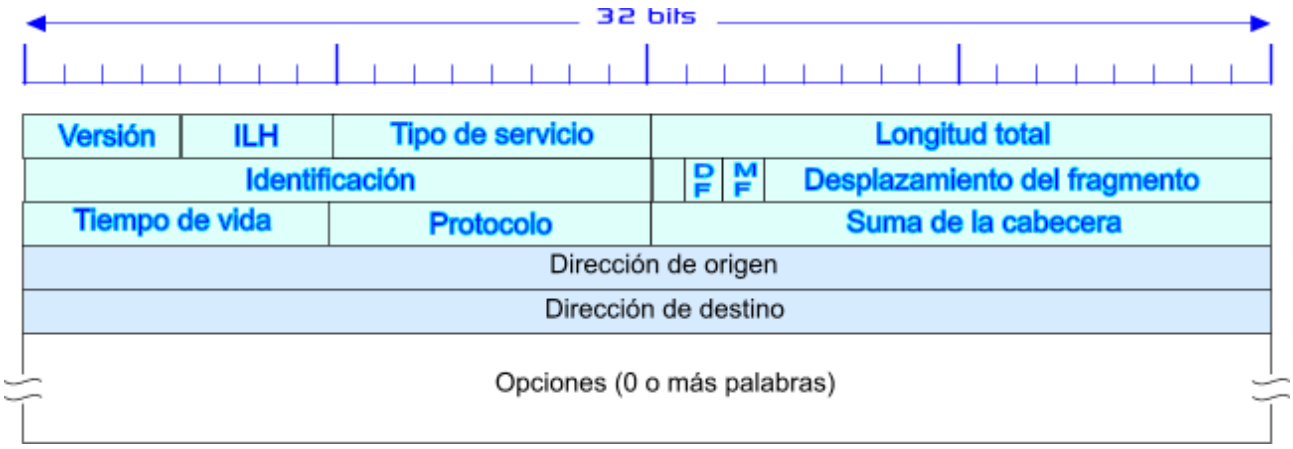


Imagen 2: Cabecera IPv4

Versión: 4 bits

Siempre vale lo mismo (0100). Este campo describe el formato de la cabecera utilizada. En la tabla se describe la versión 4.

Tamaño Cabecera (IHL): 4 bits

Longitud de la cabecera, en palabras de 32 bits. Su valor mínimo es de 5 para una cabecera correcta,

y el máximo de 15.

Tipo de Servicio: 8 bits

Indica una serie de parámetros sobre la calidad de servicio deseada durante el tránsito por una red. Algunas redes ofrecen prioridades de servicios, considerando determinado tipo de paquetes "más importantes" que otros (en particular estas redes sólo admiten los paquetes con prioridad alta en momentos de sobrecarga). Estos 8 bits se agrupan de la siguiente manera. Los 5 bits de menos peso son independientes e indican características del servicio:

Bit 0: sin uso, debe permanecer en 0.
Bit 1: 1 costo mínimo, 0 costo normal.
Bit 2: 1 máxima fiabilidad, 0 fiabilidad normal.
Bit 3: 1 máximo rendimiento, 0 rendimiento normal.
Bit 4: 1 mínimo retardo, 0 retardo normal.

Los 3 bits restantes están relacionados con la precedencia de los mensajes, un indicador adjunto que indica el nivel de urgencia basado en el sistema militar de precedencia de la CCEB, un organización de comunicaciones electrónicas militares formada por 5 naciones. La urgencia que estos estados representan aumenta a medida que el número formado por estos 3 bits lo hace, y responden a los siguientes nombres.

000: De rutina.
001: Prioritario.
010: Inmediato.
011: Relámpago.
100: Invalidación relámpago.
101: Procesando llamada crítica y de emergencia.
110: Control de trabajo de Internet.
111: Control de red.

Longitud Total: 16 bits

Es el tamaño total, en octetos, del datagrama, incluyendo el tamaño de la cabecera y el de los datos. El tamaño máximo de los datagramas usados normalmente es de 576 octetos (64 de cabeceras y 512 de datos). Una máquina no debería enviar datagramas mayores a no ser que tenga la certeza de que van a ser aceptados por la máquina destino.

En caso de fragmentación este campo contendrá el tamaño del fragmento, no el del datagrama original.

Identificador: 16 bits

Identificador único del datagrama. Se utilizará, en caso de que el datagrama deba ser fragmentado, para poder distinguir los fragmentos de un datagrama de los de otro. El creador del datagrama debe asegurar un valor único para la pareja origen-destino y el tipo de protocolo durante el tiempo que el datagrama pueda estar activo en la red.

Indicadores: 3 bits

Actualmente utilizado sólo para especificar valores relativos a la fragmentación de paquetes:

bit 0: Reservado; debe ser 0
bit 1: 0 = Divisible, 1 = No Divisible (DF)
bit 2: 0 = Último Fragmento, 1 = Fragmento Intermedio (le siguen más fragmentos) (MF)
La indicación de que un paquete es indivisible debe ser tenida en cuenta bajo cualquier circunstancia. Si el paquete necesitara ser fragmentado, no se enviará.

Posición de Fragmento: 13 bits

En paquetes fragmentados indica la posición, en múltiplos de 8 bytes, que ocupa el paquete actual dentro del datagrama original. El primer paquete de una serie de fragmentos contendrá en este campo el valor 0.

Tiempo de Vida (TTL): 8 bits

Indica el máximo número de encaminadores que un paquete puede atravesar. Cada vez que algún nodo procesa este paquete disminuye su valor en, como mínimo, un encaminador. Cuando llegue a ser 0, el paquete no será reenviado.

Protocolo: 8 bits

Indica el protocolo de siguiente nivel utilizado en la parte de datos del datagrama.

Suma de Control de Cabecera: 16 bits

Suma de Control de cabecera. Se recalcula cada vez que algún nodo cambia alguno de sus campos (por ejemplo, el Tiempo de Vida). El método de cálculo (intencionadamente simple) consiste en sumar el complemento a 1 de cada palabra de 16 bits de la cabecera y hacer el complemento a 1 del valor resultante.

Dirección IP de origen: 32 bits

Dirección IP de destino: 32 bits

Opciones: Variable

Aunque no es obligatoria la utilización de este campo, cualquier nodo debe ser capaz de interpretarlo.

IPv6 (RFC2460)

El protocolo IPv6 es una nueva versión de IP (Internet Protocol), diseñada para reemplazar a la versión 4 (IPv4). El nuevo estándar mejorará el servicio globalmente; por ejemplo, proporcionará a futuras celdas telefónicas y dispositivos móviles con sus direcciones propias y permanentes. Al día de hoy se calcula que las dos terceras partes de las direcciones que ofrece IPv4 ya están asignadas.

IPv6 admite 340.282.366.920.938.463.463.374.607.431.768.211.456 (2^{128} o 340 sextillones)

direcciones —cerca de 3,4 × 10^{20} (340 trillones) direcciones por cada pulgada cuadrada (6,7 × 1017 o 670 mil billones direcciones/mm2) de la superficie de La Tierra.

Cabecera IPv6

Imagen 3: Cabecera IPv6

Los campos renombrados respecto a IPv4 son:

Longitud de la carga útil (en IPv4, "Longitud total").

En definitiva, es la longitud de los propios datos, y que puede ser de hasta 65535 bytes. Tiene una longitud de 16 bits. (2 bytes).

Siguiente cabecera (en IPv4 "protocolo").

Dado que en lugar de usar cabeceras de longitud variables se emplean sucesivas cabeceras encadenadas, de ahí que desaparezca el campo "opciones". Tiene una longitud de 8 bits.

Límite de saltos (En IPv4 "Tiempo de vida").

Establece el límite de saltos. Tiene una longitud de 8 bits.

Los nuevos campos son :

Clase de Tráfico (Traffic Class).

También denominado Prioridad (Priority), o simplemente Clase (Class) . Podría ser más o menos equivalente a TOS en IPv4. Tiene una longitud de 8 bits. (1 byte).

Etiqueta de flujo (Flow Label).

Sirve para permitir tráfico con requisitos de tiempo real. Tiene una longitud de 20 bits.

Estos dos campos son los que nos permiten una de las características fundamentales e intrínsecas de IPv6: Calidad de Servicio (QoS) y Clase de Servicio (CoS), y en definitiva un poderoso mecanismo de control de flujo, de asignación de prioridades diferenciadas según los tipos de servicio.

La longitud de esta cabecera es de 40 bytes, el doble que en IPv4, pero con muchas ventajas al haberse eliminado campos redundantes. Debido a que la longitud de la cabecera es fija, implica numerosas ventajas ya que facilita el procesado en encaminadores y conmutadores. Los nuevos procesadores y microcontroladores de 64 bits pueden procesar de forma más eficazmente este tipo de cabecera, ya que los campos están alineados a 64 bits.

Shell Scripts.

Los shell scripts son programas escritos con comandos UNIX y son equivalentes a los batch de DOS aunque mucho más potentes, pues admiten ejecuciones en segundo plano y tienen un conjunto de expresiones mucho más amplio.

Una de las ventajas que presentan los shell scripts es que pueden ser portadas de una máquina UNIX a otra sin problemas, sin necesidad de retocar nada, salvo que se utilicen llamadas a programas muy concretos específicos de una versión de UNIX. Otra ventaja es la facilidad de lectura e interpretación.

El principal inconveniente que presentan respecto a los programas compilados es la lentitud de ejecución, que se puede paliar usando las utilidades built-in incluidas en el propio kernel en lugar de llamar a comandos externos que han de ser leídos de disco. Otro inconveniente es que el código resulta visible a cualquier usuario que lo pueda ejecutar.

Los comentarios se insertan anteponiendo el carácter "#" al comentario, que se extenderá hasta el final de la línea. Se colocan comentarios de documentación en diferentes partes del script para mejorar la comprensión y facilitar el mantenimiento. Un caso especial es el uso de "#" en la primera línea, seguido del carácter admiración y el path de la subshell, para indicar el intérprete con que se ejecutará el script (en nuestro caso bash):

#!/bin/bash

Es interesante saber que muchos comandos devuelven un valor después de ejecutarse, y que este valor indicará si la ejecución ha sido buena o si ha habido algún fallo y qué tipo de fallo se ha producido.

Para poder ejecutar un archivo de comandos es necesario que tenga activados, al menos, los permisos de lectura y ejecución.

De cara al ámbito del proyecto, se ha generado una jerarquía en árbol en la cual, el usuario debe hacer una llamada inicial a uno de los script y, a partir de ahí, se genera una secuencia de llamadas que hace que se ejecute el conjunto completo. El motivo por el que se usan en el proyecto es porque

permiten automatizar una configuración compleja que realizándose a mano, supondría una pérdida de tiempo e ineficacia al tener que ejecutarla en varias ocasiones. De este otro modo, con un sólo script., se desencadena una serie de llamadas a otros scripts y la configuración se lleva a cabo sin errores y en una cantidad de tiempo mucho menor.

A continuación se muestra el árbol.

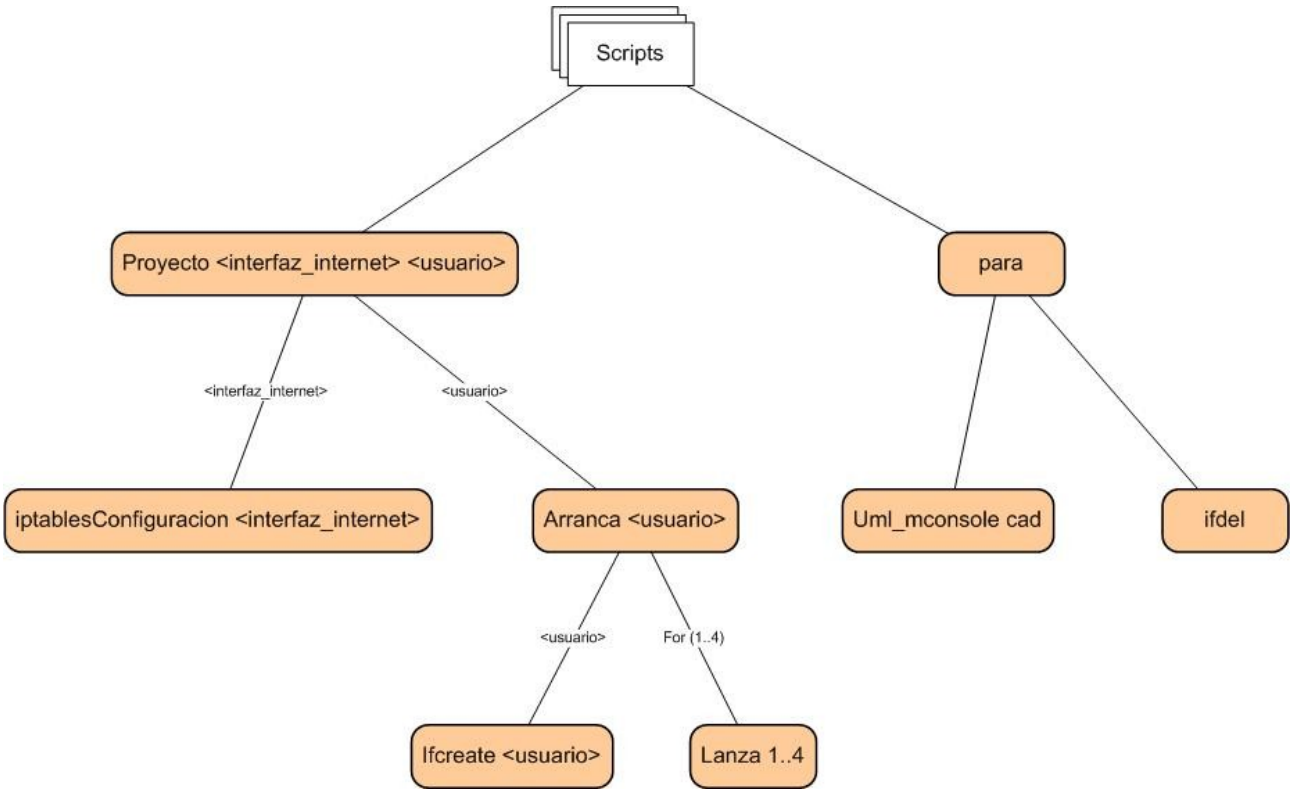


Imagen 4: Jerarquía de los scripts del desarrollo.

Túneles SIT.

De entre los túneles disponibles (ipip, sit, gre, isatap) se ha elegido el túnel sit porque es el que más se adecúa al objetivo del proyecto, comunicar dos islas IPv6 a través de IPv4.

Para empezar, SIT (significa Simple Internet Transition) significa transición simple de internet.

El modo SIT consiste en tunelizar IPv6 sobre IPv4, pero, ¿qué significa tunelizar?.

Tunelizar significa escapsular datos desde un tipo de protocolo (IPv6) a otro tipo de protocolo (IPv4) y enviándo éste sobre un canal que entiende el protocolo sobre el que se encapsula (IPv4).

SIT dispone de un driver que implementa la encapsulación de IPv6 dentro de paquetes IPv4.

Para su uso, se debe marcar la opción mode de la llamada ip tunnel a 'sit'.

Construcción de nuestros túneles IPv4.

Primero hagamos un túnel IPv4:

Digamos que tenemos 3 redes: las redes internas 1 y 2, y una red intermedia 0 (por ejemplo, Internet).

De manera que tenemos la red 1:

```
network 192.168.1.0
netmask 255.255.255.0
encaminador  192.168.1.1
```

El encaminador tiene la dirección 192.168.0.1 en la red 0. Llamaremos a esta red interfazHacia2.

Y la red 2:

```
network 192.168.2.0
netmask 255.255.255.0
encaminador  192.168.2.1
```

El encaminador tiene la dirección 192.168.0.3 en la red 0. Llamemos a esta red interfazHacia1.

Hasta donde concierne a la red 0, asumiremos que dejará pasar los paquetes enviados de 1 a 2 y viceversa.

En el encaminador de la red 1, haremos lo siguiente:

```
ip tunnel add interfazHacia2 mode sit remote 192.168.0.3 local 192.168.1.1 ttl
255
ip link set interfazHacia2 up
ip addr add 192.168.1.1 dev  interfazHacia2
ip route add 192.168.2.0/24 dev interfazHacia2
```

Miremos esto con más atención. En la línea 1, hemos añadido un dispositivo de túnel, y le hemos llamado interfazHacia2 (bastante obvio porque es a donde queremos llegar). Más aún, le hemos dicho que use el protocolo SIT (mode sit), que la dirección remota es 192.168.0.3 (el encaminador en el otro extremo), que nuestros paquetes de túnel deberían ser originados por 192.168.0.1 (lo que permite a nuestro encaminador tener varias direcciones IP en la red 0 y decidir cual usar para el tunelizado) y que el campo TTL del paquete debería establecerse en 255 (ttl 255).

La segunda línea habilita el dispositivo.

En la tercera línea le hemos dado a la recién nacida interfaz interfazHacia2 la dirección 192.168.1.1. Esto está bien para redes pequeñas, pero cuando empiece una expedición de zapadores (MUCHOS túneles), quizá debiera considerar usar otro rango de IP para las interfaces de túneles (en este ejemplo, podría usar 192.168.3.0).

En la cuarta línea hemos establecido la ruta hacia la red 2.

Veamos el encaminador de la red 2.

```
ip tunnel add interfazHacia1 mode sit remote 192.168.0.1 local 192.168.0.3 ttl
255
ip link set interfazHacia1 up
ip addr add 192.168.2.3 dev  interfazHacia1
ip route add 192.168.1.0/24 dev  interfazHacia1
```

Y cuando vaya a eliminar el túnel del encaminador 1:

```
ip link set interfazHacia2 down
ip tunnel del  interfazHacia2
```

Por supuesto, puede cambiar interfazHacia2 por interfazHacia1 para el encaminador 2.

Encaminador.

Un encaminador es un dispositivo de interconexión de redes informáticas que permite asegurar el encaminamiento de paquetes entre redes o determinar la ruta que debe tomar el paquete de datos.

Cuando un usuario accede, por ejemplo, a una URL, el cliente web (navegador) consulta al servidor de nombre de dominio, el cual le indica la dirección IP del equipo deseado.

La estación de trabajo envía la solicitud al encaminador más cercano, es decir, a la pasarela predeterminada de la red en la que se encuentra. Este encaminador determinará así el siguiente equipo al que se le enviarán los datos para poder escoger la mejor ruta posible. Para hacerlo, el encaminador cuenta con tablas de encaminamiento actualizadas, que son verdaderos mapas de los itinerarios que pueden seguirse para llegar a la dirección de destino. Existen numerosos protocolos dedicados a esta tarea.

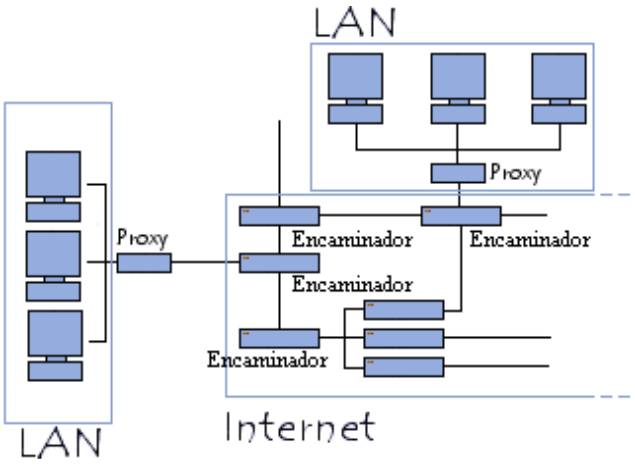


Imagen 5: Ejemplo de una red con encaminadores.

Además de su función de encaminar, los encaminadores también se utilizan para manipular los datos que circulan en forma de datagramas, para que puedan pasar de un tipo de red a otra. Como no todas las redes pueden manejar el mismo tamaño de paquetes de datos, los encaminadores deben fragmentar los paquetes de datos para que puedan viajar libremente.

Red.

Una red de computadoras (también llamada red de ordenadores o Red informática) es un conjunto de equipos (computadoras y/o dispositivos) conectados por medio de cables, señales, ondas o cualquier otro método de transporte de datos, que comparten información (archivos), recursos (CD-ROM, impresoras, etc.) y servicios (acceso a internet, e-mail, chat, juegos), etc.

Para simplificar la comunicación entre programas (aplicaciones) de distintos equipos, se definió el Modelo OSI por la ISO, el cual especifica 7 distintas capas de abstracción. Con ello, cada capa desarrolla una función específica con un alcance definido.

Host.

Se define como una máquina conectada a una red de ordenadores y que tiene una dirección ip asociada. Es una dirección ip única que se le da a un dispositivo conectado a una red informática. Puede ser un ordenador, un servidor de archivos, un dispositivo de almacenamiento por red, una máquina de fax, impresora, etc.

Iptables.

Es el componente más popular construido sobre Netfilter. Iptables es una herramientas de cortafuegos que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPv4 o mantener registros de log. El proyecto ofrecía compatibilidad hacia atrás con ipchains hasta hace relativamente poco, aunque hoy día dicho soporte ya ha sido retirado al considerarse una herramienta obsoleta.

Netfilter es un framework disponible en el núcleo Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento. Netfilter es también el nombre que recibe el proyecto que se encarga de ofrecer herramientas libres para cortafuegos basados en Linux.

El proyecto Netfilter no sólo ofrece componentes disponibles como módulos del núcleo sino que también ofrece herramientas de espacio de usuario y librerías.

Iptables es el nombre de la herramienta de espacio de usuario mediante la cual el administrador puede definir políticas de filtrado del tráfico que circula por la red. El nombre iptables se utiliza frecuentemente de forma errónea para referirse a toda la infraestructura ofrecida por el proyecto Netfilter. Sin embargo, el proyecto ofrece otros subsistemas independientes de iptables tales como el

connection tracking system o sistema de seguimiento de conexiones, que permite encolar paquetes para que sean tratados desde espacio de usuario. Iptables es un software disponible en prácticamente todas las distribuciones de Linux actuales.

Tipos de tablas

Hay tres tablas ya incorporadas, cada una de las cuales contiene ciertas cadenas predefinidas. Es posible crear nuevas tablas mediante módulos de extensión. El administrador puede crear y eliminar cadenas definidas por usuarios dentro de cualquier tabla. Inicialmente, todas las cadenas están vacías y tienen una política de destino que permite que todos los paquetes pasen sin ser bloqueados o alterados.

- **filter table** (Tabla de filtros, que ha sido la que hemos modificado): Esta tabla es la responsable del filtrado (es decir, de bloquear o permitir que un paquete continúe su camino). Todos los paquetes pasan a través de la tabla de filtros. Contiene las siguientes cadenas predefinidas y cualquier paquete pasará por una de ellas:
  - INPUT chain (Cadena de ENTRADA): Todos los paquetes destinados a este sistema atraviesan esta cadena (y por esto se la llama algunas veces LOCAL_INPUT o ENTRADA_LOCAL)
  - OUTPUT chain (Cadena de SALIDA): Todos los paquetes creados por este sistema atraviesan esta cadena (a la que también se la conoce como LOCAL_OUTPUT o SALIDA_LOCAL)
  - FORWARD chain (Cadena de REDIRECCIÓN): Todos los paquetes que meramente pasan por este sistema para ser encaminados a su destino recorren esta cadena
- **nat table** (Tabla de traducción de direcciones de red): Esta tabla es la responsable de configurar las reglas de reescritura de direcciones o de puertos de los paquetes. El primer paquete en cualquier conexión pasa a través de esta tabla; los veredictos determinan como van a reescribirse todos los paquetes de esa conexión.
- **mangle table** (Tabla de planchado): Esta tabla es la responsable de ajustar las opciones de los paquetes, como por ejemplo la calidad de servicio. Todos los paquetes pasan por esta tabla.

Forwarding.

Forwarding es una transmisión de los paquetes a través de un segmento de la red hacia otro a través de los nodos (máquinas físicas) en una red de ordenadores.

Para que esto funcione se debe tener activado el forwarding para que se transmitan.

Sysctl -w net.ipv4.conf.all.forwarding = 1.

Sysctl -w net.ipv6.conf.all.forwarding = 1.

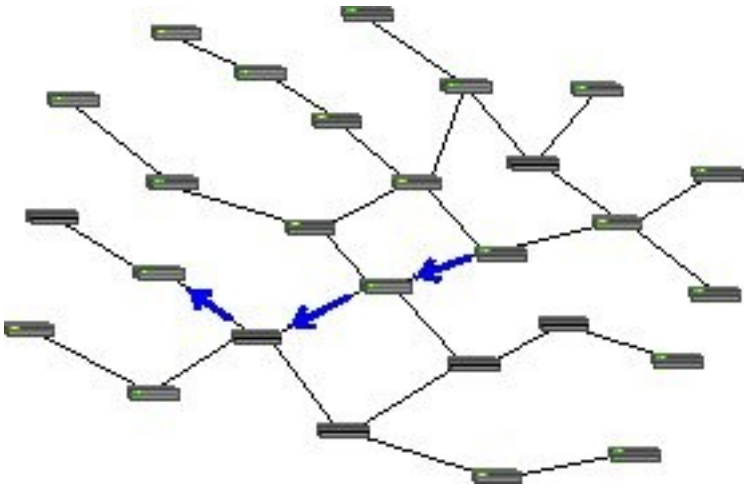


Imagen 6: Ejemplo de forwarding en una red.

Bridge (Puente de Red).

Un puente o bridge es un dispositivo de interconexión de redes de ordenadores que opera en la capa 2 (nivel de enlace de datos) del modelo OSI. Este interconecta dos segmentos de red (o divide una red en segmentos) haciendo el pasaje de datos de una red hacia otra, con base en la dirección física de destino de cada paquete.

Un bridge conecta dos segmentos de red como una sola red usando el mismo protocolo de establecimiento de red.

Funciona a través de una tabla de direcciones MAC detectadas en cada segmento a que está conectado. Cuando detecta que un nodo de uno de los segmentos está intentando transmitir datos a un nodo del otro, el bridge copia la trama para la otra subred. Por utilizar este mecanismo de aprendizaje automático, los bridges no necesitan configuración manual.

La principal diferencia entre un bridge y un hub es que el segundo pasa cualquier trama con cualquier destino para todos los otros nodos conectados, en cambio el primero sólo pasa las tramas pertenecientes a cada segmento. Esta característica mejora el rendimiento de las redes al disminuir el tráfico inútil.

Para hacer el bridging o interconexión de más de 2 redes, se utilizan los switches.

Se distinguen dos tipos de bridge:

- Locales: sirven para enlazar directamente dos redes físicamente cercanas.
- Remotos o de área extensa: se conectan en parejas, enlazando dos o más redes locales, formando una red de área extensa, a través de líneas telefónicas.

Estructura.

Sistema de ficheros.

El sistema de ficheros queda configurado de la siguiente forma a partir de un directorio Home.

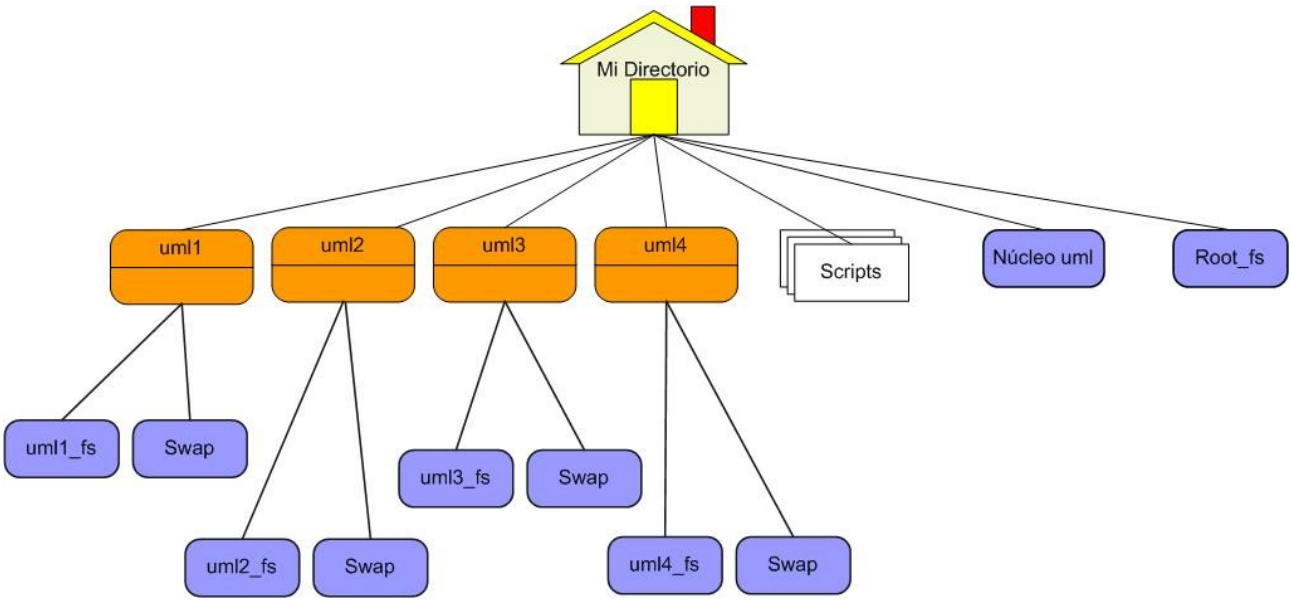


Imagen 7: Estructura de los ficheros y directorios para el desarrollo.

Sistema emulado.

El sistema de máquinas que se ha diseñado y construido es el siguiente.

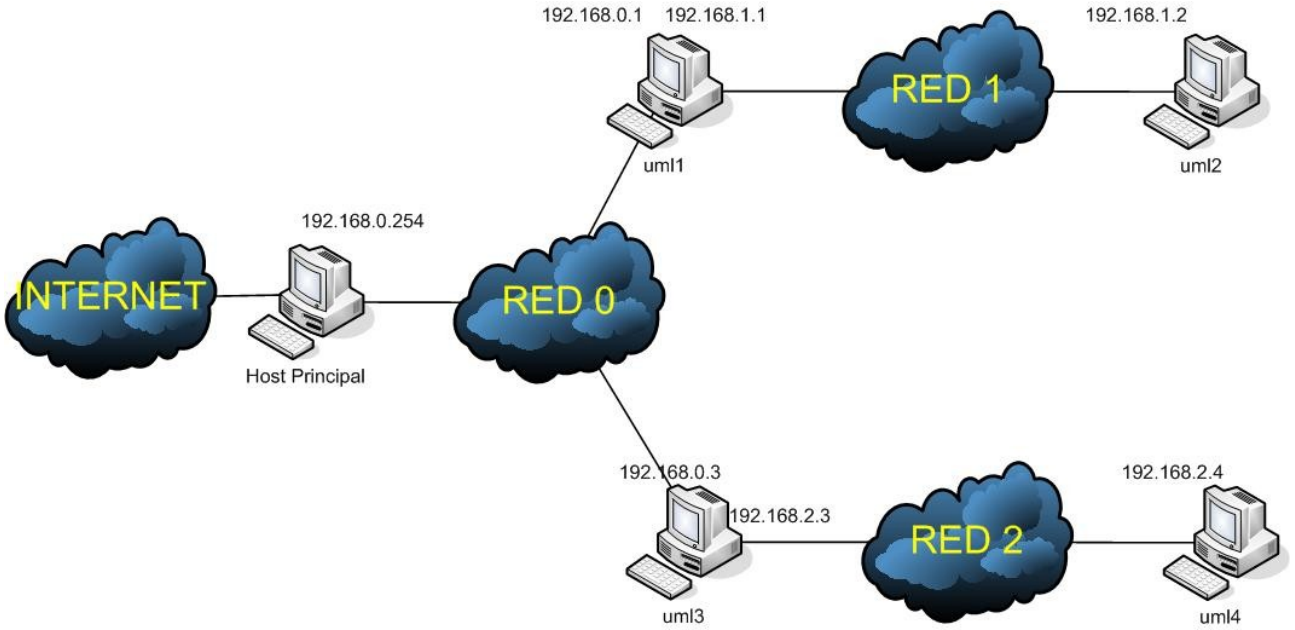


Imagen 8: Sistema emulado mediante máquinas UML.

Para las primeras pruebas en la red interna se usa IPv4. Luego se cambiará por IPv6.

A gran escala y teniendo en cuenta máquinas físicas reales y no UMLs, será un sistema similar al siguiente:

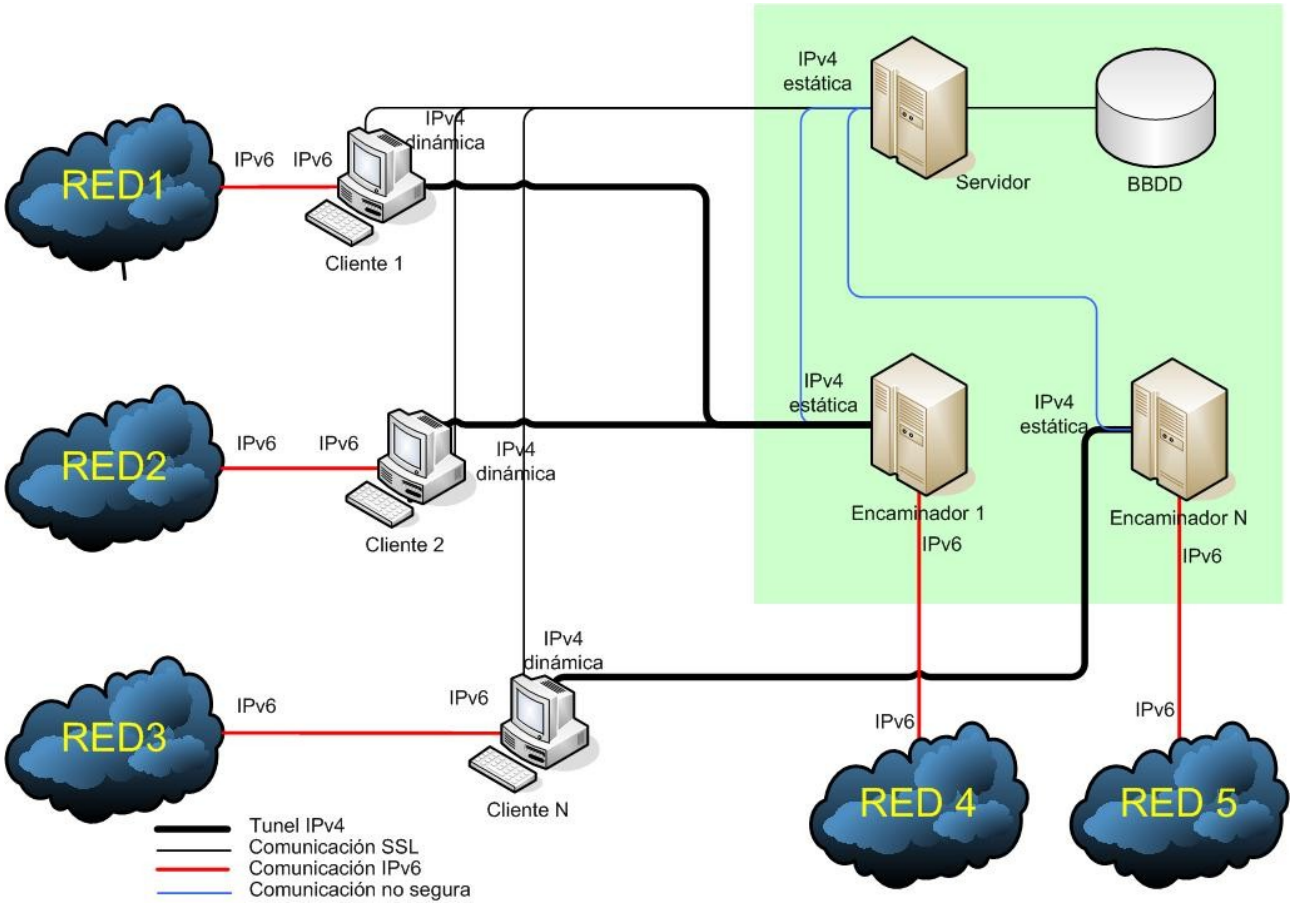


Imagen 9: Sistema real para la ejecución del proyecto.

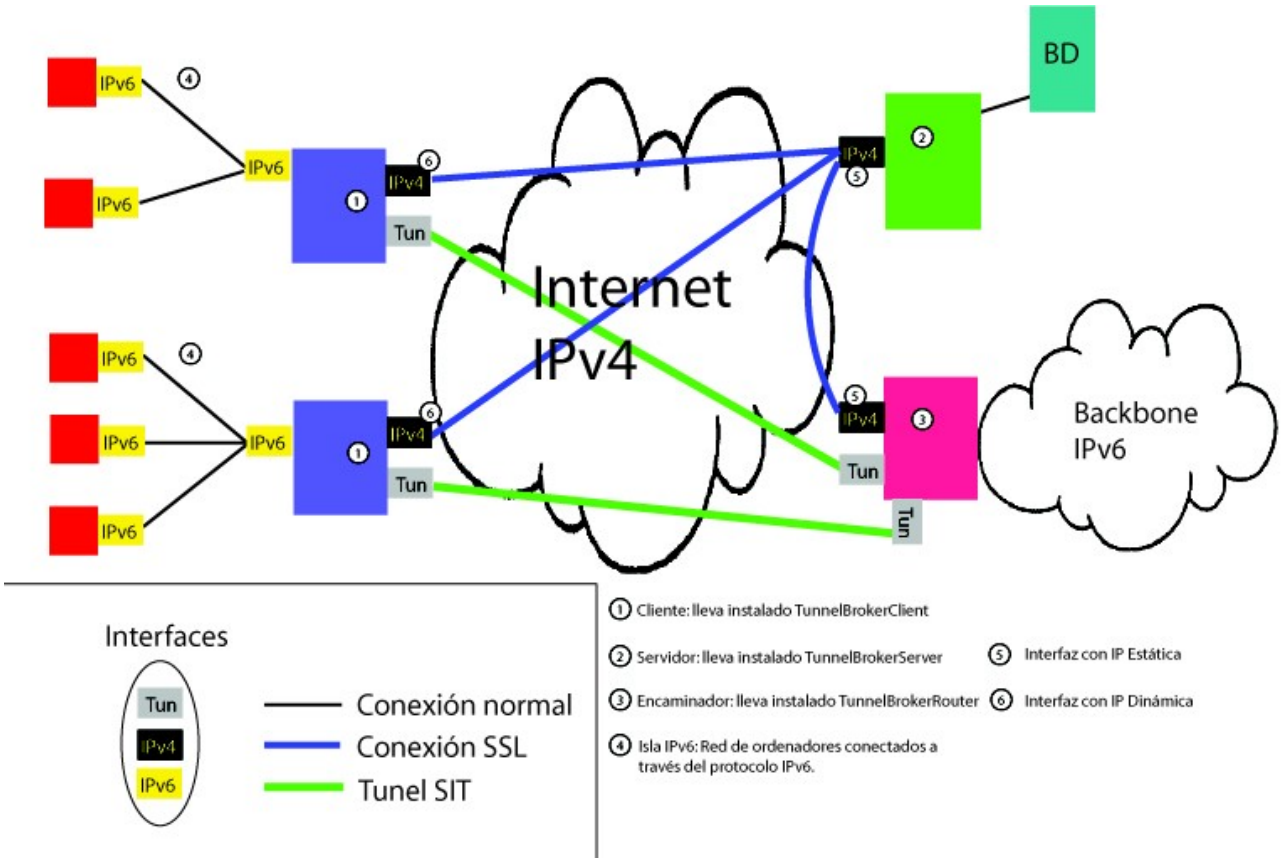


Imagen 10: Esquemático del sistema real.

Interfaz de Gestión

Introducción a los requisitos de la interfaz.

Los requisitos del proyecto exigen que un usuario cualquiera se pudiera dar de alta en el servicio de túneles a través de una interfaz web, por otro lado, no se especificaba cómo debía ser la interfaz que gestionase el resto de funcionalidades. Por ello, se abren múltiples posibilidades, ya que se puede solucionar el problema de muchas formas. Algunas de ellas son:

- Implementar una interfaz gráfica en C++ usando la librería Qt que se conectase remotamente a la base de datos ubicada en el servidor.
- Implementar una interfaz gráfica en C++ usando la librería GTK que se conectase remotamente a la base de datos ubicada en el servidor.
- Implementar la interfaz mediante una página web.
- Implementar la interfaz mediante una página web utilizando un framework.
- No implementar una interfaz gráfica y sí una tipo consola.

La primera opción en descartarse fue la de no implementar una interfaz gráfica, ya que perjudica notablemente la usabilidad de la aplicación.

Las 2 primeras opciones se descartaron por varias razones:

- La conexión remota a una base de datos ralentiza bastante el funcionamiento normal de la aplicación cada vez que es necesario realizar una consulta.
- Mediante una página web, todo el procesamiento y consultas a la base de datos se hacen en la máquina local, de otro modo, se deberían habilitar conexiones remotas a la base de datos, lo cual es menos seguro.
- Mediante una página web, siempre se puede acceder desde cualquier parte del mundo simplemente utilizando un navegador, mientras que con una aplicación compilada, hay que asegurarse de disponer de un sistema compatible, instalarla y configurarla antes de poder conectarse.
- Es preferible comunicarse con máquinas y seres vivos utilizando estándares abiertos ampliamente adoptados como son los lenguajes de marcado HTML/XHTML.

Por otro lado, implementar la gestión mediante una interfaz web presenta 2 desventajas respecto a la implementación en un programa local:

- Implica la instalación de un servidor web, desventaja que no es tal ya que, como se ha comentado anteriormente, unos de los requisitos especificaba que cualquier usuario debía

poder darse de alta a través de una interfaz web, con lo que el servidor web iba a estar instalado de todas formas.

- Aunque todo el procesamiento y consultas a la base de datos se hacen en la máquina local, transmitir un documento HTML/XHTML puede ser más costoso que una consulta remota a la base de datos, dependerá de la funcionalidad pedida.

Elección del framework

Una vez descartadas todas las opciones que no supusieran realizar una página web, se debía tomar una decisión sobre el modo de generar la página web. Las soluciones contempladas fueron:

- Realizar toda la interfaz web en PHP
- Utilizar el framework Codeigniter
- Utilizar el framework Joomla!1.5.x

La primera opción se descartó desde un principio ya que en un proyecto no se pide reinventar la rueda, al contrario, a lo largo de la carrera se ha insistido hasta la saciedad en la importancia de la reutilización del código.

Codeigniter es un framework de PHP, que posibilita el desarrollo de webs de forma muy rápida, no te da ninguna funcionalidad completamente implementada, simplemente provee de clases y funciones que habitualmente se utilizan al desarrollar una página web. Otra ventaja de este framework es que ha sido desarrollado para que su código sea lo más rápido posible, aunque hoy en día, debido a los grandes avances en la capacidad de procesamiento, no es un factor determinante.

Joomla! además de un framework escrito en PHP, es un gestor de contenidos, es decir, es una aplicación que permite a los usuarios contribuir con artículos y a los administradores gestionar todo ese contenido generado por los usuarios. Además, ha sido diseñado para poder extender su funcionalidad mediante su API, con lo que Joomla!, a diferencia de Codeigniter, sí proporciona funcionalidad ya implementada, como por ejemplo:

- Registro y gestión de los usuarios.
- Gestión de conexiones seguras mediante SSL.
- Soporte para múltiples idiomas (está traducido a 81 idiomas).
- Configuración general del servidor, soporte para caché, y muchas más.

Las ventajas y desventajas de desarrollar la aplicación con Joomla! son:

- Escrito en PHP.
- Amplia funcionalidad ya implementada.
- Funcionalidad fácilmente extensible mediante componentes de terceros.

- Permite añadir y modificar el contenido sin tener que editar código de programación.
- Permite modificar el aspecto de forma dinámica.
- Respaldado por una comunidad muy activa.
- Ampliamente probado.
- Open source (licencia GPL).
- API compleja debido a su amplia funcionalidad.

Debido a la cantidad de ventajas que presentaba desarrollar la aplicación web usando Joomla!, se decidió utilizarlo, aunque, después se descubrió que extender la funcionalidad exigía un conocimiento profundo de su API.

Diseño de la aplicación web.

Joomla! está diseñado para tener dos partes bien diferenciadas:

- Sitio (front end)
- Administración (back end)

El sitio (front end) es la parte de la web a la que los clientes podrán acceder, mientras que la administración (back end), sólo accesible a los administradores, permite gestionar todos los pormenores del programa. Ambas partes están tan diferenciadas que los clientes nunca sabrán de la existencia de la administración (back end), de hecho, no hay enlaces entre ambos.

Joomla! permite tres tipos de extensiones:

- Módulos
- Plugins
- Componentes

La diferencia entre ellos reside en los permisos que tiene cada uno para acceder a determinada información de la base de datos y determinadas funciones. La única extensión que permite extender la funcionalidad de la administración (back end) son las componentes. Por ello, en el marco de este proyecto, se ha desarrollado una componente para Joomla!.

La API de Joomla! implementa el patrón MVC (modelo-vista-controlador). Dicho patrón permite independizar la capa de integración y de negocio de la capa de presentación favoreciendo el bajo acoplamiento entre las distintas capas. Joomla! no sólo utiliza dicho patrón, sino que obliga a utilizarlo en todas las extensiones.

En la capa de integración, permite usar el patrón active record, el cual facilita enormemente las cuatro operaciones básicas de manejo de entidades en las BBDD, lo que en inglés se denomina

CRUD (create read update delete). Este patrón de diseño requiere crear una clase por cada entidad, de forma que debe tener un atributo por cada columna de la tabla en cuestión. Active record permite realizar alguna de las cuatro operaciones mencionadas sin tener que escribir ninguna sentencia SQL.

Funcionalidad de la componente

Las componentes, al igual que Joomla!, están divididas en 2 partes, la administración y el sitio. Las componentes que no necesiten mostrar o procesar información a los clientes no tendrán que implementar el sitio, aunque no ha sido nuestro caso.

Funcionalidad de la administración

En la administración de la componente se pueden gestionar todas las entidades del proyecto, es decir:

- Encaminadores
- Prefijos
- Conexiones
- Clientes

Encaminadores

La entidad encaminador se compone del nombre del encaminador, la IPv4 , el puerto y el número máximo de conexiones.

Uno de los requisitos del proyecto especificaba que el encaminador podía estar en una máquina diferente a la del servidor y, además, podía haber más de un encaminador. La componente permite:

- Dar de alta encaminadores (tantos como se quiera)
- Borrar encaminadores
- Actualizar los datos de un encaminador
- Obtener los datos de un encaminador
- Listar todos los encaminadores dados de alta

Se ha añadido a la entidad encaminador un parámetro no especificado en los requisitos, para permitir un número máximo de conexiones en los mismos. De esta forma, el administrador puede adaptar el número de conexiones a las características propias del hardware del encaminador y así conseguir un rendimiento óptimo. Otra posible configuración que puede interesar al administrador sería crear una red de encaminadores en la que hubiese algún o algunos encaminadores menos saturados para conexiones preferentes y de esta forma obtener mayor velocidad para dichas

conexiones.

Prefijos

La entidad prefijo se compone del prefijo IPv6, la longitud del prefijo IPv6 y el tiempo de vida en días del mismo.

La componente permite realizar las siguientes operaciones sobre la entidad prefijos:

- Generar un subprefijo IPv6 a partir del prefijo IPv6 otorgado a la web
- Borrar prefijos
- Actualizar los datos del prefijo
- Obtener los datos de un prefijo
- Listar todos los prefijos dados de alta

Los sistemas gestores de bases de datos actuales no soportan un tipo de columna tipo IPv6, por ello, es necesario averiguar la mejor forma de guardar los prefijos IPv6. A priori, existen las siguientes posibilidades para guardar un prefijo en una tabla:

- Una cadena de caracteres (VARCHAR)
- Un número binario de 128 bits
- Un número decimal de 39 dígitos

Todas las opciones anteriores son malas soluciones, las 2 últimas porque Mysql no soporta números tan grandes y la primera porque hacer búsquedas de cadenas de caracteres es muy poco eficiente.

La solución a este problema pasaba por guardar un prefijo en más de una columna:

- Ocho columnas decimales, una cada por cada grupo en la notación hexadecimal
- Dos columnas BIGINT(20), una correspondiente a los bits 0-63 y otra para los bits 64-127.

Al final se ha elegido la segunda opción ya que era la solución que aparentemente más se había adoptado hasta ahora y porque había librerías beta en PHP para transformar un prefijo IPv6 en notación hexadecimal a dos números enteros correspondientes a los bits mencionados anteriormente.

Dicha librería, era una versión beta ya que no funcionaba en todos los casos, y por ello, se ha tenido que rehacer para que funcionase con números enteros grandes. Hay que tener en cuenta que ningún lenguaje actual soporta de forma nativa números enteros tan grandes como los necesarios para poder manejar los prefijos IPv6. Por ejemplo, el número entero máximo permitido en PHP es 2^64 -1 (18446744073709551615) mientras que es necesario poder manejar números enteros de hasta 2^128 -1 (340282366920938463463374607431768211455). El problema de la librería consistía en que intentaba realizar operaciones matemáticas con números demasiado grandes para el tipo nativo

de PHP. Para solucionarlo, se han utilizado números de precisión arbitraria, es decir, números que se guardan como cadenas de caracteres y, por tanto, permiten longitudes muchísimo mayores (tanto como la longitud máxima de una cadena de caracteres).

En PHP existe una librería llamada *bcmath* que permite realizar operaciones matemáticas básicas con números de precisión arbitraria, éstas son:

- Suma
- Resta
- Multiplicación
- División
- Comparación

Hay que reseñar que las operaciones con números de precisión arbitraria son mucho más lentas que las operaciones con tipos nativos del lenguaje, por ello, es recomendable utilizarlas sólo cuando sean imprescindibles.

Asignación de Prefijos

Una de las principales tareas de un túnel broker es otorgar prefijos IPv6 contenidos en el propio prefijo del túnel broker a los clientes que lo deseen. Hay múltiples maneras de realizar dicha tarea, se ha optado por resolverlo de una manera similar a como lo hacen los sistemas operativos a la hora de asignar direcciones de memoria a los procesos. Hay que tener en cuenta que los prefijos siempre deben ser potencias de 2, lo que representa una diferencia respecto a las direcciones de memoria.

Hay 3 formas de asignar los prefijos:

Primer ajuste: asigna el prefijo utilizando las direcciones del primer hueco encontrado que sea suficientemente grande.

Mejor ajuste: busca un hueco del tamaño exacto al pedido, si no lo hubiera, asigna el prefijo utilizando las direcciones del hueco con tamaño más cercano al pedido que sea suficientemente grande.

Peor ajuste: busca el hueco más grande, y asigna el prefijo utilizando las direcciones de ese hueco si fuera suficiente grande.

Se ha realizado una comparativa del rendimiento de los diferentes algoritmos en la sección llamada *Comparativa de los algoritmos de asignación de prefijos*.

Conexiones

La entidad conexiones permite asociar un cliente y su IPv4 a un encaminador. Sólo se permite un prefijo por cliente y por ello, una sola conexión por cliente.

La componente permite realizar las siguientes operaciones sobre la entidad conexiones:

- Dar de alta una conexión
- Borrar una conexión
- Actualizar los datos de una conexión
- Obtener los datos de una conexión
- Listar todas las conexiones dadas de alta

Hay que resaltar que el administrador no debería manipular las conexiones ya que ése es el cometido del TunnelBrokerServer (programa del servidor). Por ejemplo, si se borra una conexión de las base de datos mediante la interfaz web, habría que borrar manualmente los túneles tanto del cliente como del encaminador. Estas operaciones se ofrecen por completitud y para que el administrador pueda utilizarlas en casos muy particulares.

Clientes

La entidad clientes sirve para identificar al usuario tanto en la administración (back end) como en el sitio (front end). Esta entidad la proporciona Joomla! ya que internamente ya existía la funcionalidad para registrar un usuario y gestionarlo. Por tanto, ésta entidad ha sido reutilizada sin cambios.

La componente permite realizar las siguientes operaciones sobre la entidad clientes:

- Dar de alta un cliente
- Borrar un cliente
- Bloquear un cliente
- Actualizar los datos de un cliente
- Obtener los datos de un cliente
- Listar todos los clientes dados de alta

Aunque los conceptos de usuario y cliente pueden parecer similares, tienen un matiz importante y diferenciador ya que un usuario puede ser cliente o administrador, mientras que un cliente no puede ser un administrador. Joomla! provee una jerarquía de usuarios asignando diferentes privilegios a cada uno de ellos. Es por ésta razón por la que si un cliente accede a la administración (back end) e intenta iniciar la sesión se le denegará el acceso al no tener los permisos suficientes.

La clasificación de los tipos de usuarios que hay en Joomla! es la siguiente:

Usuarios del Sitio (front end):

- **Invitado (Guest)**: Un invitado es sencillamente un usuario de Joomla! que ha navegado hasta

encontrar su sitio web. Dependiendo de cómo haya configurado el sitio el administrador, los invitados podrán navegar libremente por todo el contenido o tener restringido el acceso a cierto tipo de contenidos reservados para usuarios registrados.

- **Registrado (Registered)**: Un usuario registrado no puede crear, editar o publicar contenido en un sitio Joomla!. Puede enviar nuevos enlaces web para ser publicados y puede tener acceso a contenidos restringidos que no están disponibles para los invitados.
- **Autor (Author)**: Los autores pueden crear contenido, indicar ciertos aspectos de cómo se presentará el contenido y especificar la fecha en la que se publicará el material. El contenido creado por ellos no estará publicado hasta que un supervisor dé el visto bueno.
- **Editor (Editor)**: Un editor tiene todas las posibilidades de un autor, y además la capacidad de editar el contenido de sus propios artículos y los de cualquier otro autor.
- **Supervisor (Publisher)**: Los supervisores pueden ejecutar todas las tareas de los autores y editores, y además tienen la capacidad de publicar un artículo.

Usuarios de la administración (back end):

- **Gestor (Manager)**: Un gestor puede ser visto como un supervisor con acceso al panel de administración del back end. El gestor tiene acceso, en el panel del administrador, a todos los controles asociados al contenido, pero no tiene capacidad para cambiar las plantillas, alterar el diseño de las páginas o añadir o eliminar extensiones de Joomla!. Los mánager tampoco tienen autoridad para añadir usuarios o alterar los perfiles de usuarios existentes.
- **Administrador (Administrator)**: Los administradores tienen un rango de acceso más amplio que los gestores. Además de todas las actividades relacionadas con el contenido que puede ejecutar un gestor, los administradores pueden añadir o eliminar extensiones al sitio web, cambiar plantillas o alterar el diseño de las páginas, e incluso alterar los perfiles de usuario a un nivel igual o inferior al suyo. Lo que no pueden hacer los administradores es editar los perfiles de súper administradores o cambiar ciertas características globales del sitio web. De hecho, ni siquiera verán los usuarios de tipo súper administrador en el administrador de usuarios.
- **Súper administrador (Super Administrator)**: Los súper administradores tienen el mismo poder que un root en un sistema tradicional Linux y disponen de posibilidades ilimitadas para ejecutar todas las funciones administrativas de Joomla!. Sólo los súper administradores tienen la capacidad de crear nuevos usuarios con permisos de súper administrador o asignar éste permiso a usuarios ya existentes.

Funcionalidad del sitio (front end)

El sitio permite gestionar al cliente la entidad prefijo, es decir, le permite:

- Obtener un prefijo si no lo ha hecho antes
- Modificar el tiempo de vida del prefijo que se le haya asignado
- Borrar el prefijo que se le haya asignado

Para poder acceder a la interfaz de gestión del cliente, el usuario debe ser como mínimo un usuario de tipo registrado, es decir no puede ser un invitado.

Además de la interfaz de gestión, el sitio tiene distintas secciones para que el usuario pueda descargarse el TunnelBrokerClient (el programa cliente) y la documentación necesaria.

Comparativa de los algoritmos de asignación de prefijos

¿Cómo se han realizado las pruebas?

Se ha creado un pequeño programa de pruebas, cuyo funcionamiento es el siguiente:
- Selecciona de forma aleatoria una longitud para un prefijo entre 3 posibles candidatos.
- Intenta generar un prefijo de dicha longitud con la estrategia escogida.
- Si lo genera, vuelve a comenzar el proceso
- Si no lo genera, es porque no hay espacio para un prefijo de dicha longitud. Escoge una longitud menor, de entre las 3 posibles, y sigue generando prefijos hasta que no quepa ninguno.

Para realizar las pruebas se han escogido los siguientes valores para los parámetros:
- Espacio de direcciones: 2^24 direcciones (16777216 direcciones), es decir, un prefijo de longitud 104.
- Prefijos generados de longitud aleatoria con valores posibles de 120, 118 ó 108.

Resultados

Primer ajuste

- Número de prefijos generados: 484.
- Tiempo medio para generar un prefijo: 0,01632 microsegundos
- Tiempo total de la ejecución del script: 9,0138 segundos
- Tiempo total ejecución del algoritmo Primer ajuste: 7,899160 microsegundos
- Número de huecos generados: 0 huecos

Mejor ajuste

- Número de prefijos generados: 499.
- Tiempo medio para generar un prefijo: 0,016937 microsegundos
- Tiempo total de la ejecución del script: 9,5942 segundos

- Tiempo total ejecución del algoritmo mejor ajuste: 8,45169 microsegundos
- Número de huecos generados: 0 huecos

Peor ajuste

- Número de prefijos generados: 2839.
- Tiempo medio para generar un prefijo: 0,10464 microsegundos
- Tiempo total de la ejecución del script: 304,2983 segundos
- Tiempo total ejecución del algoritmo peor ajuste: 297,097 microsegundos
- Número de huecos generados: 0 huecos

Gráficas comparativas

Utilizando los datos mostrados en la sección resultados, se han elaborado unas gráficas para poder comparar los resultados más fácilmente.

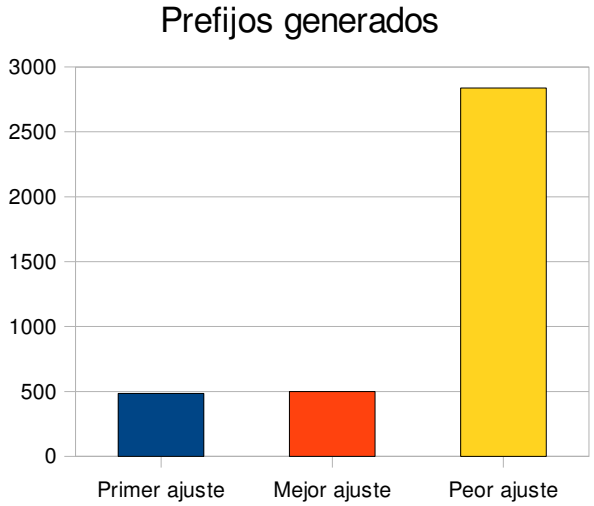


Imagen 11: Gráfica con el número de prefijos generados con cada algoritmo.

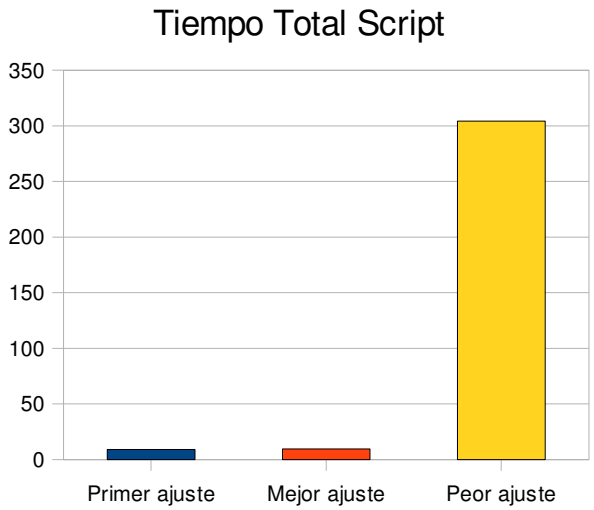


Imagen 12: Gráfica con el tiempo total empleado para ejecutar el script completo.

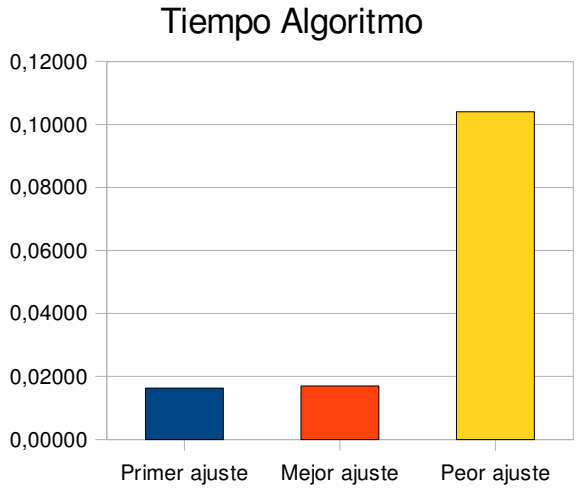


Imagen 13: Gráfica con el tiempo medio empleado para ejecutar el algoritmo para generar un prefijo.

Explicación de los resultados

Peor ajuste

Es sin duda, la peor elección. Es el algoritmo que más prefijos ha generado, lo que sólo se explica debido a que favorece la fragmentación de huecos pequeños, es decir, es el algoritmo que menos prefijos de mayor tamaño ha generado y el que más prefijos pequeños ha generado, de ahí el número tan elevado de prefijos asignados. Este hecho es más significativo de lo que pudiera parecer, ya que, como el coste del algoritmo depende del número de prefijos, a mayor número de prefijos, mayor

coste total.

Primer ajuste.

Es el algoritmo más rápido de los 3, seguido muy de cerca de Mejor ajuste, a costa de favorecer un poco la fragmentación.

Mejor ajuste

Su coste es muy parecido a primer ajuste, con la diferencia de que nos aseguramos una menor fragmentación. Las pruebas indican que es sólo un 3,73% más lento que Primer ajuste.

Conclusión

Los resultados han esclarecido sin lugar a dudas los pros y los contras de cada algoritmo, revelando que la mejor elección es sin duda Mejor ajuste, ya que es sólo un poco mas lento que primer ajuste, pero con una menor fragmentación. La diferencia de coste con respecto a primer ajuste se presenta muy pequeña si se tiene en cuenta la contrapartida que se gana ya que en un entorno de producción el coste extra de Mejor ajuste sería inapreciable para el usuario.
Resulta curioso el mal resultado de Peor ajuste, no se esperaba que la fragmentación en huecos pequeños perjudicará tanto el coste final, los resultados son claros y no debería utilizarse en ninguna circunstancia.
Las pruebas realizadas no representan un modelo fiel del mundo real, ya que no se suele asignar prefijos de cualquier tamaño sin parar hasta que no se pueda asignar ninguno más, por ello, las pruebas no son un fiel reflejo de un entorno de producción y sólo unas pruebas orientativas.
Se podía haber hecho una prueba adicional que consistiría en ir asignando prefijos hasta que uno no cupiese, en ese punto se deberían contar el número de prefijos generados de cada longitud, el número de huecos y el tiempo total. Esta prueba también sería un prueba orientativa que no penalizaría tanto al algoritmo peor ajuste, ya que no generaría tantos prefijos pequeños.

Aplicaciones TunnelBroker (TB)

Introducción

El sistema TunnelBroker permite la conexión automática entre redes independientes que funcionan sobre el protocolo IPv6. Para conseguir dicha conexión una red conectada por IPv6 que compone una isla independiente, deberá tener configurado un terminal que le de acceso a través de un túnel a otro terminal que hará el papel de encaminador hacia algún backbone de internet IPv6.
Una vez que el usuario se ha dado de alta en el sistema a través del interfaz web ofrecida a tal fin, y ha obtenido un prefijo IPv6 para la configuración de su isla, podrá instalar en un terminal de su red la aplicación TB-C para que el sistema le ofrezca acceso automático a algún backbone de internet IPv6 a través de túneles con los encaminadores propios del mismo.

El funcionamiento básico del sistema de conexión se basa en tres componentes:

- **TunnelBrokerClient (TB-C)**: Se instala en la red configurada por el usuario. El terminal donde va instalado TB-C es uno de los extremos del túnel a través del cual viajarán los paquetes IPv6 encapsulados en paquetes IPv4 para aprovechar la infraestructura del internet actual basado en IPv4. TB-C se comunica con TB-S para crear y mantener túneles con los encaminadores que le proporcionan acceso a Internet IPv6.
- **TunnelBrokerServer (TB-S)**: Instalado en una máquina accesible a través de Internet IPv4 y con dirección IPv4 estática. La aplicación TB-S mantiene un control de las conexiones existentes entre clientes y encaminadores. Mediante el envío de keep-alives las aplicaciones TB-C mantienen viva la conexión con el encaminador, además de proveer la información de su dirección IPv4 por si esta hubiese sido modificada.
- **TunnelBrokerRouter (TB-R)**: El encaminador se encarga de hacer, rehacer y deshacer los túneles con los clientes que le haya asignado la aplicación TB-S. El encaminador estará siempre a la escucha por si la aplicación TB-S le comunica algún cambio en los túneles creados actualmente o por si le comunica la necesidad de crear un nuevo túnel con un cliente.

Comunicaciones entre componentes

Introducción

La seguridad de las comunicaciones entre los tres componentes del sistema es un requisito indispensable. Una comunicación insegura que permitiese el robo de información por un tercero utilizando un sniffer o una comunicación en la que se pudiese suplantar la identidad de uno de los actores impiden una implantación mínimamente realista del sistema.

Comunicación básica

El protocolo TCP

Protocolo perteneciente a la capa 4 del modelo OSI. Se encuentra entre el protocolo IP y, en este caso, la capa de conexión segura SSL.

El protocolo de internet (IP) no es orientado a conexión, y además, no garantiza que los paquetes lleguen a su destino, o de llegar, que lleguen con todos los datos correctos. Los protocolos de la capa de transporte, como TCP, son los encargados de hacer la comunicación sobre el protocolo de internet fiable.

TCP maneja las conexiones de extremo a extremo. Añade toda la funcionalidad necesaria para permitir una conexión fiable entre los dos extremos a comunicar. Dicha comunicación estará libre de errores, será full-duplex y además, TCP se encargará del control de flujo y del envío de reconocimientos para que cualquiera de los extremo sepa que el otro ha recibido correctamente la información.

La especificación del protocolo TCP data de 1981, se encuentra en la RFC 793.

Sockets

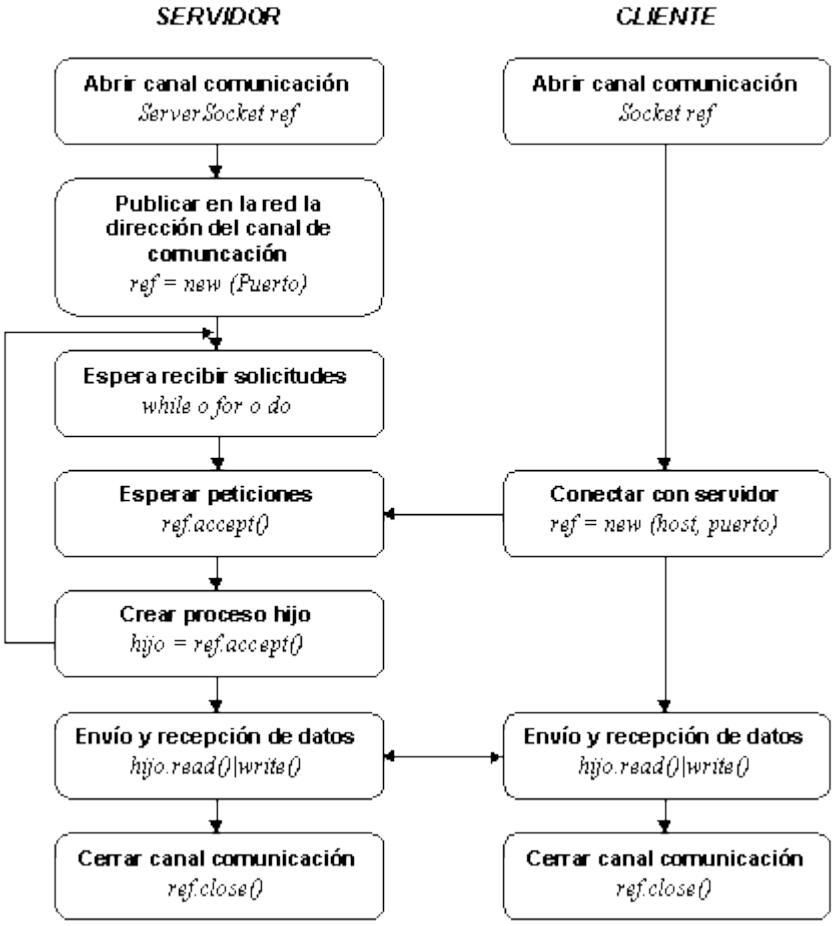
Un socket es un punto de comunicación por el cual dos procesos pueden intercambiarse cualquier flujo de datos.

El concepto de socket se define en base a tres conceptos:

- Un protocolo de comunicaciones, que permite el intercambio de octetos.
- Una dirección del Protocolo de Red (Dirección IP, si se utiliza el protocolo TCP/IP).
- Un número de puerto, que identifica a un programa dentro de una computadora.

Los sockets permiten implementar una arquitectura cliente-servidor. La comunicación ha de ser

iniciada por uno de los procesos (el proceso cliente). El segundo proceso espera a que otro inicie la comunicación, el programa servidor. En la siguiente imagen se muestra un ejemplo de cliente-servidor implementado con sockets:



Un socket es un descriptor existente en el cliente y en el servidor, que sirve en última instancia para que el programa servidor y el cliente lean y escriban la información. Esta información será la transmitida por las diferentes capas de red.

Se ha escogido sockets TCP por las siguientes características:

- Orientado a conexión.
- Se garantiza la transmisión de todos los octetos sin errores ni omisiones.
- Se garantiza que todo octeto llegará a su destino en el mismo orden en que se ha transmitido.
- Existencia de la Berkeley Sockets API: Es el estándar de facto para Sockets en linux. Una implementación ampliamente extendida, probada y documentada.

Estas propiedades son muy importantes para garantizar la corrección de los programas que tratan la información.

El problema que presenta el uso simple de sockets es que, si bien garantizan la transmisión, no garantizan la identidad de los extremos ni la seguridad de los datos que transmiten.

Posibles vulnerabilidades

Man in the middle

Ataque consistente en leer y también modificar la comunicación entre dos extremos sin que estos tengan constancia de dicho ataque.

En el sistema supone un riesgo por:

- TunelBrokerClient se identifica en la aplicación TunelBrokerServer enviando dentro del mensaje el nombre de usuario y contraseña con el que está registrado en el sistema. Dicha información en manos de un tercero malintencionado le da permisos para poder acceder a toda la información del usuario a través del interfaz web.
- Otro motivo de riesgo en el sistema es la posibilidad de que un tercero malintencionado envíe mensajes falsos de creación o destrucción de túneles.

En definitiva un ataque “man in the middle” permite a cualquier individuo suplantar la identidad del usuario en el sistema, ya sea a nivel de administración como durante el funcionamiento de una conexión.

Las aplicaciones TB impiden este tipo de ataques utilizando el protocolo SSL en la capa de aplicación del modelo OSI. De esta forma todos los componentes se comunicarán entre ellos de forma cifrada, y aquellos que lo requieran comprobarán la identidad de otros a través de certificados firmados.

Denegación de servicio

Ataque consistente en impedir que un recurso en una red sea inaccesible a los usuarios que pretenden utilizarlo.

Existen numerosas formas de producir este tipo de ataques, muchas de las cuales solo podrán evitarlas o corregirlas los mismos administradores de la red.

Si, por ejemplo, se efectúa un ataque contra el servidor que aloja la aplicación TunelBrokerServer, se paralizaría la creación y destrucción de túneles. Aún así, las conexiones actualmente creadas se mantendrían funcionando correctamente.

Reenvío de datos cifrados

Otro ataque que podría tener lugar es la captura de datos cifrados, y el reenvío de los mismos tiempo después.

Si una aplicación TunelBrokerClient envía un mensaje de creación de túnel con una IPv4 y tiempo después cambia dicha IP, el cliente enviará la nueva IP en el próximo mensaje keep-alive. Si un tercero hubiese capturado el primer mensaje con la IP antigua, aun estando cifrado y no pudiendo descifrarlo, podría reenviarlo al servidor TunelBroker alterando la correcta creación de la conexión.

Esta situación no podrá darse, puesto que TB-C y TB-S, gracias al protocolo SSL, cifran de forma distinta cada conexión que establecen entre ellos, de tal modo que un mensaje cifrado en una conexión, no valdrá tiempo después puesto que será necesario rehacer el handshake.

Comunicación segura

Las comunicaciones entre los distintos componentes del sistema, como ya se ha mencionado anteriormente, utilizan el protocolo SSL. La utilización de dicho protocolo permite asegurar tanto la seguridad como, en el caso de requerirlo, la autenticación de los componentes.

Protocolo TCP y la capa de conexión segura SSL

SSL encaja perfectamente en el diseño planteado para las aplicaciones implementadas en el proyecto. Las aplicaciones que se han desarrollado utilizan la arquitectura cliente-servidor para el intercambio de datos. Dicha arquitectura se ha basado completamente en la utilización de sockets implementados con el protocolo TCP: orientado a conexión, que garantiza de esta forma la transmisión y el orden de llegada de los datos.

El uso del protocolo UDP se desestimó desde un principio por no aportar características que se han considerado fundamentales. La necesidad de asegurar la correcta recepción de los mensajes intercambiados entre aplicaciones hace que la garantía que ofrece la forma de funcionar de TCP resulte más adecuada. Además, la capa de conexión segura SSL exige tener la comunicación orientada a conexión, con lo que la elección del protocolo TCP sobre UDP es clara.

Protocolos de seguridad

La implementación de la seguridad en la capa de transporte no es el único sistema posible para asegurar y cifrar las comunicaciones. También existen protocolos que funcionan sobre la capa de red. Por ejemplo IpSec es un conjunto de protocolos que actúan sobre dicha capa.

Si bien la utilización de protocolos en la capa de red como IpSec habría ahorrado la gestión y uso de la capa de seguridad en las aplicaciones TB, por otro lado también supone un problema, puesto que exige más requisitos en los sistemas que alojan las aplicaciones. IpSec obliga a que dichos sistemas tengan una "pila de red" IpSec habilitada para poder utilizar la seguridad en el protocolo IPv4, ya

que el uso de IpSec en esta versión es opcional, mientras que en la versión 6 se incluye el uso de IpSec por defecto.

Así pues, dado que no se ha querido exigir como requisito de los sistemas donde se ejecutarán las aplicaciones TB la compatibilidad con IpSec, se ha decidido utilizar los protocolos de seguridad existente sobre la capa de transporte. En concreto el protocolo SSL (TLS en su versión más actual).

Secure Socket Layer (SSL)

El protocolo SSL es un sistema diseñado y propuesto por Netscape Communications Corporation. Se encuentra en la pila OSI entre los niveles de TCP/IP y de los protocolos de la capa de aplicación: HTTP, FTP, SMTP, etc. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de sesión es la que se utiliza para cifrar los datos que vienen del y van al servidor seguro. Se genera una clave de sesión distinta para cada transacción, lo cual permite que aunque sea reventada por un atacante en una transacción dada, no sirva para descifrar futuras transacciones. MD5 se usa como algoritmo de hash.

Proporciona cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

Cuando el cliente pide al servidor una comunicación segura, el servidor abre un puerto cifrado, gestionado por un software llamado Protocolo SSL Record, situado encima de TCP. Será el software de alto nivel, Protocolo SSL Handshake, quien utilice el Protocolo SSL Record y el puerto abierto para comunicarse de forma segura con el cliente.

En este proyecto sólo a las aplicaciones TB-S y TB-R se les exigirá identificarse, puesto que las aplicaciones TB-C llevarán una tupla nombre de usuario/contraseña en los datos que será suficiente para asegurar su identidad. (Como se explica en este mismo documentos, las aplicaciones TB-C llevarán un certificado autofirmado y TB-S no comprobará la validez del mismo).

Para la implementación de las aplicaciones utilizando la capa de conexión segura, se han utilizado las librerías y herramientas del proyecto OpenSSL. La versión de OpenSSL que se ha utilizado (versión 0.9.8) soporta los protocolos de seguridad de red Secure Sockets Layer v2/v3 (SSLv2/SSLv3) y Transport Layer Security v1 (TLSv1).

Certificados

Los certificados resuelven el problema de la suplantación de identidad. Al utilizar certificados firmados para TB-S, se asegura que ningún TB-C comenzará una conexión segura con un servidor fraudulento. De nada serviría cifrar la comunicación si dicha comunicación se establece entre la aplicación TB-C y un servidor malintencionado.

La aplicación TB-S tendrá un certificado firmado por una autoridad certificadora. Para la

realización del proyecto se utiliza la posibilidad de autofirmado de OpenSSL para simular el papel de Autoridad Certificadora (CA en inglés) y generar los certificados asegurando que los clientes reconocen la CA. En un entorno de producción la CA debería ser una empresa dedicada a tal fin y suficientemente reconocida, como por ejemplo VeriSign.

Este tipo de empresas tienen un papel muy importante en la seguridad de redes y sistemas. Su papel es el de ofrecerse como una tercera parte que da garantías de que las partes implicadas son quienes dicen ser.

Necesidad de certificados en la criptografía de clave pública

Alicia quiere enviar información secreta a Benito, para ello deberá hacerse con una copia de la clave pública de Benito. Antes de hacerlo, sin embargo, necesita asegurarse de que dicha clave pública realmente pertenece a Benito, si no tendría una comunicación cifrada pero con alguien que podría no ser Benito.

Los Certificados se encargan de este problema. Un certificado vincula una clave pública con un individuo u organización particular. Los Certificados son emitidos por las empresas mencionadas anteriormente, que no son más que terceros de confianza. Antes de dar un certificado, se supone que estas empresas utilizan una serie de procedimientos de autenticación para asegurarse de que Benito, en este caso, es quien afirma ser. De tal forma que el certificado contendrá la clave pública de Benito y la garantía de que efectivamente le pertenece a él.

Información contenida en un certificado SSL:

- Nombre común de su organización (ejemplo www.e-sign.cl)
- Información adicional de identificación (ejemplo IP y dirección física)
- Su clave pública
- Fecha de caducidad de la clave pública.
- Identificación de la Autoridad emitió el certificado.
- Un número de serie único.

La forma en que la Autoridad Certificadora garantiza la autenticidad de Benito es utilizando también criptografía asimétrica. El certificado de Benito es firmado con la clave privada de la CA. Si las aplicaciones finales (en este caso las aplicaciones TB-C) tienen la clave pública de dicha CA dentro de su lista de personas de confianza, podrán estar seguros de la legitimidad del certificado de Benito. En este proyecto, que está desplegado en un entorno de desarrollo y no de producció , se autofirma el certificado del servidor y se añade la clave pública correspondiente a la simulación de la CA a la lista de autoridades que los clientes consideran de confianza.

Para la creación de los certificados se han usado herramientas proporcionadas en el proyecto OpenSSL. El mismo proyecto que contiene las librerías para la implementación de la comunicación

en la capa SSL.

Utilización de la librería de OpenSSL

OpenSSL es un paquete de programas y librerías con funciones criptográficas y desarrollados para ayudar en el desarrollo y despliegue de aplicaciones que requieran un mínimo de seguridad. Está basado en el paquete SSLeay de Eric Young.

La utilización de OpenSSL se extiende a todas las aplicaciones TunnelBroker.

OpenSSL implementa el protocolo SSL/TLS sobre TCP. Funciona de la siguiente manera:

TCP efectua su negociación en tres pasos:

El cliente envía el mensaje SYN.

El servidor acepta (ACK) y envía a su vez un mensaje SYN.

El cliente acepta (ACK) y comienza la comunicación.

Imagen 14: Negociación TCP.

OpenSSL añade un extra a esta negociación para el intercambio de claves y algoritmos de cifrado:

- El cliente envía un listado de algoritmos de cifrado que acepta y un *número aleatorio**.
- El servidor devuelve el algoritmo elegido, su certificado, su clave pública y un número aleatorio. (En este punto el servidor podría solicitar el certificado del cliente si quisiese autenticarlo, para el caso de TB-C no será necesario dicha autenticación).
- El cliente verifica el certificado: Aquí aparece la necesidad de la firma por parte de una CA que el cliente considere válida. Será necesario añadir a la lista de CAs de confianza en la aplicación TB-C la CA con el cual se firma el certificado de TB-S. (Para el caso de TB-R ocurre igual).
- El cliente, una vez verificado el certificado enviará una clave privada cifrada con la clave pública que le proporcionó el servidor. (De esta forma sólo podrá descifrar el mismo servidor).
- El servidor acepta la clave privada enviada por el cliente.
- Finalmente se establece la comunicación. Los datos irán cifrados con claves generadas a partir de la clave privada, compartida por ambos extremos, utilizándolas para realizar cifrado simétrico: que utiliza una misma clave para cifrar y descifrar, es más seguro y mucho más rápido.

  *El *número aleatorio* se utiliza para la generación de claves criptográficas y firmas digitales. Es importante un generador de numeros aleatorios lo suficientemente fuerte para poder garantizar la seguridad y claves y certificados no reproducibles. OpenSSL en linux utiliza */dev/urandom, /dev/random,* o */opt/openssl/prngd/prngd* como generadores de números aleatorios.

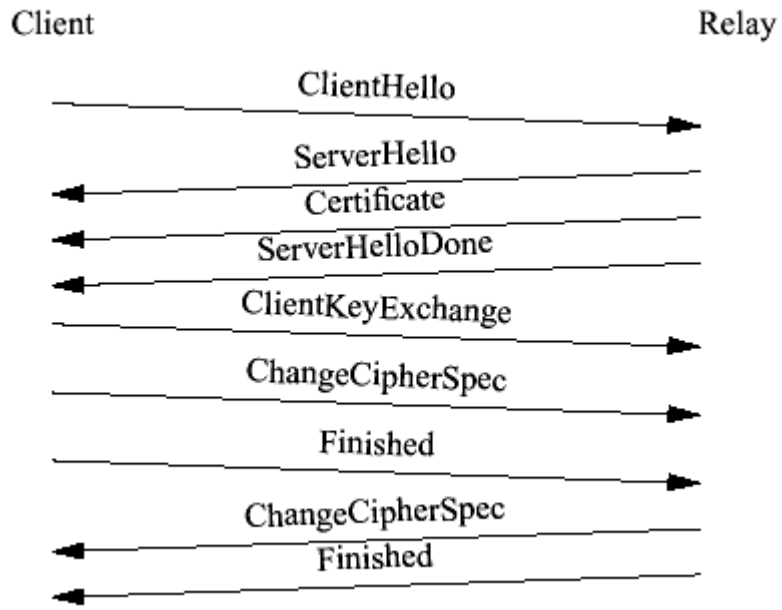


Imagen 15: Intercambio de claves y algoritmos de cifrado en OpenSSL.

Reutilización de sesiones

Un problema que puede surgir en el despliegue en producción de un sistema como TB es la escalabilidad.

Un punto débil en la escalabilidad del sistema será el cifrado de la información en la aplicación TB-S. El cifrado de los datos supone un consumo de CPU mucho mayor que el que se tiene con una conexión sin cifrar. Además, a pesar de que el cifrado de los datos se realiza de forma simétrica con OpenSSL, para obtener la clave privada secreta que se utiliza para generar las claves del cifrado simétrico, TB-S debe descifrar dicha clave cuando se la envía el cliente. Este proceso se realiza con cifrado asimétrico, mucho más costoso y lento. Por este motivo existe el concepto de sesión.

Las sesiones permiten que un mismo cliente reaproveche la clave privada secreta entre más de una conexión con el servidor. Si se mantiene una caché de sesiones en el servidor y el cliente guarda también la clave privada ya utilizada con anterioridad, antes de comenzar la parte de la negociación en la comunicación SSL, el cliente enviará un identificador para que el servidor recupere de la caché la clave a utilizar. De esta forma la negociación sólo requerirá del intercambio de algoritmos de cifrado y el número aleatorio con los cuales se generarán las claves simétricas a partir de la clave privada que se reutiliza. (El hecho de que no se use la misma clave privada para el cifrado simétrico protege frente al envío de datos cifrados capturados por un tercero como se explicó previamente en este documento).

La caché de sesiones no ha sido implementada en la aplicación TB-S. Se han intentado y analizado diversas alternativas para la implementación de la misma, y se ha dejado para futuras ampliaciones

dado el complejo desarrollo que supone.

Utilización de las estructuras de caché de sesiones de OpenSSL

Al utilizar distintos procesos para cada conexión que se establece no es posible utilizar la caché de OpenSSL, puesto que esta sólo funciona si las peticiones de los clientes se llevan en un mismo proceso.

Esta situación lleva a plantear dos posibles alternativas:

- Gestión de cada petición de cliente en hilos en lugar de procesos: El intento de cambiar los procesos por hilos no fue satisfactorio y no se consiguió hacer funcionar la comunicación sobre éstos.
- Utilización de IPC (interprocess communication) para desde el programa padre gestionar la caché de las sesiones: Esta es la idea más prometedora para una ampliación futura. Esta versión se mantendrá sin sesiones dado que un desarrollo como este requiere de recursos y tiempo mayores que los establecidos en este proyecto.

Formato de mensajes entre aplicaciones

Una vez que dos componentes del sistema se intercambian datos a través de las diversas capas que dan seguridad y garantías a la comunicación, los datos recibidos deberán ser comprobados e interpretados por el destinatario.

Se ha diseñado un sencillo protocolo para la interpretación de los datos.

Mensajes enviados desde TB-C a TB-S

Usuario|*Contraseña*|*DireccionIPv4*

Usuario: deberá coincidir con el tipo de datos de la información disponible en la base de datos MySQL. La columna del nombre de usuario se ha configurado con el tipo VARCHAR(150), por lo que se exige al nombre del usuario tener un máximo de 150 caracteres.

El nombre de usuario no permitirá en ningún caso el carácter '|' utilizado para separar los campos. No se ha implementado un carácter de escape para el mismo.

Contraseña: Al igual que el nombre de usuario, la contraseña deberá coincidir con el tipo de dato en que se guarda esta en base de datos. El tipo de datos en MySQL es VARCHAR(100), luego no se permitirán contraseñas con un valor mayor a 100 caracteres.

La contraseña no permitirá en ningún caso el carácter '|' utilizado para separar los campos. No se ha implementado un carácter de escape para el mismo.

Dirección IPv4: La dirección IPv4 que envía el cliente al servido irá en forma de cadena de

caracteres. Deberá tener el formato legible de cualquier dirección IPv4, por ejemplo: 127.0.0.1. La base de datos la almacena con el tipo de datos INTEGER, de tal forma que cada número se almacena como un byte. El paso de la cadena de caracteres al tipo numérico utilizado será trabajo de TB-S en su código interno.

Mensaje de confirmación de TB-S a TB-C

ACK|DirecciónIPv4

ACK

NACK

Existen tres posibles tipos de mensaje de confirmación que TB-S puede devolver a TB-C.

ACKNOWLEDGEMENT + Dirección IPv4: confirmación de que el mensaje ha llegado correctamente y se ha procesado correctamente.

La Dirección IPv4 implica que durante el procesado se ha visto que o bien la conexión con un TB-R no existe o bien ha caducado. Se envía la dirección IPv4 del encaminador seleccionado para rehacer el túnel SIT.

ACKNOWLEDGEMENT: confirmación de que el mensaje ha llegado correctamente y se ha procesado correctamente.

El hecho de que no exista una dirección IPv4 en el mensaje implica que la conexión se mantiene activa y por lo tanto no es necesario modificar el túnel SIT.

NACK: el mensaje enviado no ha podido ser procesado correctamente. Es posible que se deba a que el usuario y contraseña son incorrectos o a que la dirección IPv4 del cliente no se haya podido obtener y se haya enviado una dirección mal formada.

Este último caso podría darse, por ejemplo, si se configura TB-C para obtener la dirección IP pública de un servicio de terceros inexistente, o si se configura para obtener la IP privada de una interfaz errónea.

Mensajes enviados desde TB-S a TB-R:

tunnXX|Opción|DirecciónIPv4

tunnXX: Para la creación de túneles SIT se ha elegido utilizar por defecto nombres similares. Cualquier solicitud de creación o destrucción de túnel creada en TB-S utilizará como nombre de interfaz la palabra 'tunn' seguida del identificador del usuario. Como el identificador es único y sólo se creará un túnel por usuario, sabemos que el nombre tunnXX dado al mismo será también único.

Opción: La opción indica a TB-R que acción debe realizar sobre el interfaz especificado por 'tunnXX'.

- Up: Se debe crear o rehacer un túnel SIT.
- Down: Se debe eliminar el túnel SIT.

Dirección IPv4: La dirección IPv4 que envía el servidor al encaminador irá en forma de cadena de caracteres. Deberá tener el formato legible de cualquier dirección IPv4, por ejemplo: 127.0.0.1. Esta indica la dirección IPv4 del cliente con el cual se establece o elimina el túnel.

Mensaje de confirmación de TB-R a TB-S

ACK

NACK

Existen dos posibles tipos de mensaje de confirmación que TB-R puede devolver a TB-S.

ACKNOWLEDGEMENT: confirmación de que el mensaje ha llegado correctamente y se ha procesado correctamente.

NACK: el mensaje enviado no ha podido ser procesado correctamente.

Trazas de las aplicaciones

Puesto que las aplicaciones TB se ejecutan en modo background sin participación directa del usuario, se ha considerado importante poder hacer un seguimiento de los procesos en ejecución para analizar fallos, funcionamiento o simplemente para obtener información de uso de las mismas. Para realizar este seguimiento se ha decidido escribir trazas descriptivas del proceso durante la ejecución del mismo.

Existen muchas alternativas para la escritura de trazas en aplicaciones C/C++ escritas para Linux. A continuación se describen las que se han considerado:

- **Syslog**: Es la forma que se ha utilizado para escribir las trazas. Su sencillez y buen funcionamiento han sido los motivos principales para su utilización dentro de las aplicaciones.

  Funcionamiento: Las distribuciones linux suelen incorporar por defecto el paquete 'sysklogd' Este paquete incluye dos demonios para grabar los mensajes del sistema y también aquellos generados por el núcleo.

  - **Klogd**: Demonio que se encarga de recoger los mensajes enviados por el núcleo.
  - **Syslogd**: Demonio que se encarga de recoger los mensajes enviados por los programas del sistema. Este segundo demonio es el que utilizan las aplicaciones TB a través de la cabecera syslog.h que incluye el sistema en '/usr/include'. Es una forma realmente sencilla y eficaz de trazas los procesos. Durante las pruebas del desarrollo se ha utilizado mucho el programa 'tail' incluido normalmente por defecto en las distribuciones Linux y otros sistemas basados en Unix. Este programa muestra las últimas lineas de un fichero de texto,

permitiendo ver la evolución de las trazas de forma dinámica.

- **Log4C**: Librería de logs basada en la librería de Apache para el lenguaje Java 'log4j'. Liberada en SourceForge bajo la licencia LGPL. Log4c es ha sido probada en un componente del sistema y resulta bastante sencilla de utilizar y práctica. Si no se ha terminado implantando en las aplicaciones TB ha sido porque ya se había procedido a utilizar syslog, pero Log4c resulta también muy recomendable.
- **Log4cpp**: Librería realmente similar a Log4C pero adecuada para su uso en aplicaciones programadas con C++.
- **Log4cxx**: Se hizo una prueba con esta librería también basada en Apache log4j. Puesto que no se consiguió configurarla rápidamente y dada la existencia de otras alternativas, se decidió no seguir intentándolo con ella.

Aplicación TunnelBrokerClient (TB-C)

Proceso

Se encarga de solicitar al servidor TB la creación de una conexión, para que este le proporcione un encaminador y avise a dicho encaminador de que debe crear el túnel con la IPv4 proporcionada por el cliente.

Las conexiones consisten en dos túneles SIT, uno creado desde la aplicación TB-C hacia la aplicación TB-R y otro en sentido opuesto.

El programa cliente TunelBroker deberá encargase de mantener la conexión enviando keep-alives al servidor con la IPv4 actual. Si ésta cambiase, la aplicación TB-R será avisada para rehacer el túnel SIT hacia la aplicación TB-C.

Seguridad del fichero de configuración

En la aplicación TB-C el nombre de usuario y la contraseña del mismo están escritos en texto plano en el fichero de configuración de la aplicación. Esto obliga a extremar las precauciones con dicho fichero.

Las alternativas planteadas para proteger el fichero de configuración han sido varias:

- **Cifrado**: Se podría haber realizado algún tipo de cifrado/descifrado del mismo. De tal modo que el usuario se encargase de escribir ya cifrados dichos datos. El problema de esta alternativa es que exige la implementación de una librería de descifrado para su uso en la aplicación TB-C, además de la implementación de algún tipo de aplicación que el usuario pueda utilizar para obtener sus datos como cadenas cifradas. Este desarrollo se ha considerado innecesario para esta versión del sistema, quedando abierta esta posibilidad para una futura ampliación.
- **Certificado digital para cada TB-C**: Otra opción sería la utilización de certificados digitales. Actualmente el certificado digital de TB-C se utiliza para garantizar la seguridad en la comunicación con TB-S, no para autenticar al cliente. TB-S no comprueba la firma del certificado de TB-C, únicamente lo utiliza para cifrar/descifrar los datos puesto que en estos datos es donde el usuario introduce su login y contraseña con la cual se autentifica. Podría haber sido una opción firmar y comprobar correctamente el certificado de cada TB-S. Asignando un certificado a cada usuario que lo autentifique ante TB-S. Esta opción no se ha seguido dado el importante desarrollo que ocasionaría la creación de certificados para cada cliente y distribución de los mismos. Podría haberse planteado la creación de un servidor de certificados firmados por un CA propio reconocido por TB-S y que sólo funcionase para usuarios registrados en el sistema,

de esta forma se podría haber hecho que cada TB-C dispusiese de un certificado válido que lo autenticase. Esta opción resulta demasiado costosa en tiempo/recursos como para llevarla acabo en el desarrollo del proyecto.

- **Permisos**: La opción final que se ha tomado para garantizar que la contraseña del usuario no sea robada por un tercero debido a su existencia en el fichero de configuración, ha sido modificar los permisos del mismo para que únicamente el mismo usuario que implanta y configura la aplicación TB-C tenga accesos de lectura y escritura al fichero.

Obtención de la dirección IPv4 de una máquina cliente

Puesto que el programa TB-C puede estar instalado en sistemas organizados de distintas maneras, se han acotado las posibles opciones y se han implementado distintos métodos para obtener la dirección IP con la cual se establece el túnel con el encaminador.

Cliente y encaminador en una misma red interna:

La posibilidad de encontrarse con un sistema montado dentro de una misma red privada exige ofrecer la obtención de la dirección IP privada de un interfaz, y no aquella con la cual salimos a internet.

Para la obtención de la IP de un interfaz el usuario deberá especificar en el fichero de configuración el nombre de dicho interfaz. El programa, si encuentra dicho interfaz, utilizará la primera dirección IP que tenga el mismo. Si la configuración que tiene puesta el usuario en el interfaz de red no se ajusta a este funcionamiento encargado de analizar y obtener la IP de la misma, el usuario siempre tendrá opción de escribir en el fichero de configuración la dirección IPv4 que desee utilizar.

Cliente y encaminador en distintas redes:

La posibilidad de encontrar los encaminadores en redes distintas a los clientes es bastante probable. En este caso, los túneles deberán crearse a partir de la dirección pública con las que cualquier cliente accede a internet, puesto que, obviamente, los encaminadores no tendrán acceso a la red privada donde pueda encontrarse un cliente.

Para la obtención de la IPv4 pública con la que un cliente accede a internet se han explorado distintas posibilidades, optando finalmente por la utilización de un servicio de terceros alojado en un servidor web.

Opciones exploradas:

- ***Simple Network Management Protocol (SNMP)***

  Este es el protocolo para gestión de redes más importante actualmente. Es parte del conjunto de protocolos TCP/IP, encontrándose en la capa de aplicación de la pila OSI.

  Existen tres versiones del protocolo SNMP, siendo SNMPv1 y SNMPv2 muy parecidas y las más utilizadas. SNMPv3 no está siendo masivamente utilizada pese a añadir mejoras importantes en el terreno de la seguridad como autenticación, cifrado y la utilización de reglas

para el control de acceso.

El protocolo SNMP está cubierto por varios RFCs. SNMPv1: RFC 1157, 1215. SNMPv2: del 1441 al 1452. SNMPv3: del 2271 al 2275 y del 2570 al 2575.

Conceptos básicos de SNMP

Object Identifier (OID): Corresponde a un nodo en el árbol de objetos que utiliza el protocolo. Un ejemplo de OID sería ".1.3.6.1.4.1.2021.4.5". Dicho objeto tendrá un valor y un tipo, así como un nombre que también podrá utilizarse para referenciarlo, para el ejemplo anterior "memTotalReal".

Management Information Base (MIB): Es una colección de objetos.

Funcionamiento general de SNMP:

Cada equipo conectado a una red cualquiera ejecuta un programa Agente que irá actualizando variables que pueden ser consultadas de forma externa. SNMP facilita la comunicación entre la estación administradora y los agentes que se estén ejecutando. Los equipos conectados a la red y que ejecutan un agente pueden ser de muy diverso tipo: routers, modems, switches, impresoras, otros PCs, etc.

En el caso de un router, las variables que irán actualizándose pueden ser, por ejemplo: interfaces activos, bytes emitidos, bytes recibidos, dirección IPv4 pública, ... Esta última variable ha sido el motivo por el cual se ha probado la utilización de SNMP en las aplicaciones del proyecto.

Para el análisis de las posibilidades que ofrece SNMP se han utilizado las aplicaciones proporcionadas en la colección de aplicaciones Net-SNMP obtenido de los repositorios de Debian.

La instalación de las aplicaciones básicas para el uso del protocolo es tan sencillo como:

#apt-get install snmp

Una vez instalados los paquetes, tenemos ya las herramientas necesarias para el uso de SNMP desde un simple terminal de consola.

Ejemplo de consulta SNMP:

snmpget dominio comunidad OID

Donde dominio es la dirección IP de la máquina con el agente.

Comunidad por defecto será pública o privada.

OID: el nodo a obtener. Su notación podrá ser una notación intermedia entre ASCII y ASN1.

Para el uso y desarrollo de SNMP en aplicaciones propias se han probado las librerías y cabeceras incluidas en Net-SNMP, en concreto en el paquete libsnmp-dev versión 5.4.1.

Explorada la vía del protocolo SNMP para obtener la dirección IPv4 pública, se ha decidido no

utilizarla finalmente. La principal razón es la falta de homogeneidad entre los distintos equipos que interesa consultar. Según fabricantes, e incluso versiones, los routers y modems pueden estructurar de forma muy distinta sus MIBs con la información que se intenta obtener.

- **Servicio alojado en servidor de terceros**:

  Existen a través de internet numerosas páginas que muestran, entre otros servicios, la dirección IPv4 pública desde la cual se conecta el equipo a internet.

  Se ha optado por utilizar una de las sencillas, puesto que únicamente muestra una cadena de texto:

  checkip.dyndns.org

  Existen numerosos sitios web para la obtención de la IPv4 pública. Algunos de ellos son:

  - www.whatsmyip.org
  - www.cualesmiip.com
  - www.internautas.org/w-localizamiip.html

  Herramientas utilizadas para la obtención de la dirección IPv4 pública a través de un sitio web remoto:

  - **Wget**: es una aplicación que permite la obtención de ficheros existentes en la web utilizando los protocolos http(s) y ftp.

    Entre sus múltiple opciones permite simplemente imprimir en la salida estándar el contenido del fichero obtenido en la web. De tal forma que al acceder a checkip.dyndns.org, obtendremos el contenido del fichero html ofrecido por el servidor web, dentro del cual está la información de la IPv4 pública desde la que estamos accediendo al mismo.

  - **The GNU sed stream editor**: Sed permite editar un texto que se le introduce por la entrada estándar, y una vez modificado, escribir el resultado por la salida estándar.

    A través de un pipe se ofrece la información de la página web con la IPv4 a obtener al programa sed. Dado que conocemos la estructura y formato de dicha página, podemos preparar una expresión regular para obtener únicamente los caracteres correspondientes a la dirección IPv4 que interesa.

    Desde la aplicación lo único que se hace es recoger la información de la IPv4 de la salida estándar. Esta sencillez y eficacia para la obtención la dirección IPv4 pública es lo que ha determinado la utilización de este método.

    Este procedimiento para obtener la dirección de internet puede dar problemas si el usuario se conecta a internet a través de un proxy, puesto que los servicios que proporcionan la IPv4 pública darán la IPv4 del proxy en lugar de la verdadera. Para este tipo de casos existe la posibilidad de que el usuario de las aplicaciones especifique manualmente la dirección IPv4

pública que utiliza.

Aplicación TunnelBrokerServer (TB-S)

Proceso

TB-S se mantiene a la escucha esperando las peticiones de los clientes TB para gestionar las mismas.

Para la comunicación con los clientes TB se utilizan, como se ha explicado en detalle previamente, sockets sobre el protocolo orientado a conexión TCP y añadiendo además en la capa de aplicación del modelo OSI el protocolo SSL para garantizar la seguridad en la comunicación.

El servidor implementado procesa cada petición de un cliente ejecutando un proceso nuevo e independiente. Esta forma simplifica en gran medida la gestión de cada cliente, especialmente la implementación de la seguridad con el uso de la librería OpenSSL.

Durante la comunicación establecida con un cliente, el servidor deberá realizar diversas funciones y comprobaciones:

El servidor deberá comprobar primero que el mensaje recibido es correcto y que contiene un nombre de usuario, una contraseña y una dirección IPv4. Tras ello deberá comunicarse con el gestor de bases de datos para asegurarse que el usuario y la contraseña son válidos.

Si el cliente proporcionase al router un par usuario/contraseña incorrectos, se devolverá un error al mismo. Y el proceso terminará.

Si el cliente se autentica correctamente el proceso de comprobaciones continua.

Se pasa a comprobar si el usuario tiene alguna conexión activa, pudiendo darse tres posibilidades:

- El cliente no tiene ninguna conexión activa: en este caso se le asigna un encaminador y se le envía la dirección IPv4 del mismo al cliente y la dirección IPv4 del cliente al encaminador.
- El cliente tiene una conexión pero su dirección IPv4 ha cambiado: se le envía un mensaje al encaminador para rehacer el túnel SIT con la nueva dirección IPv4.
- El cliente tiene una conexión y su dirección IPv4 se mantiene igual: se le envía un mensaje informando de que todo se mantiene correctamente al cliente y al encaminador no se le envía nada.

Gestor de Bases de Datos MySQL

La toma de decisión sobre el gestor de base de datos a utilizar con la aplicación TB-S se ha visto influenciada por varios motivos:

- Coherencia con la opción a elegir para el interfaz web, puesto que la base de datos es

compartida entre esta aplicación y la aplicación web.

- Las opciones comerciales como Oracle de Oracle Corporation, SQL Server de Microsoft o DB2 de IBM fueron descartadas desde un principio por tratarse de productos cerrados y no gratuitos. Si bien podría no haber sido determinante, en este caso, teniendo alternativas libres extendidas y de gran calidad se ha decidido elegir entre las opciones opensource y gratuitas disponibles.
- Entre los gestores de bases de datos libres que se ha planteado utilizar se encuentran:
  - **MySQL**:

    MySQL es uno de los gestores de bases de datos más popular que existe. Actualmente, y tras la operación de compra de Sun Microsystems, pertenece a Oracle Corporation.

    MySQL ha sido elegido para este proyecto frente al resto de alternativas dada la familiaridad del equipo con ella así como la cantidad de recursos y documentación existente en la red (aunque tampoco las alternativas planteadas escasean en documentación y recursos, está claro que es MySQL el gestor de base de datos más extensamente utilizado).

    Otro factor determinante ha sido el hecho de que el CMS utilizado para la interfaz web: Joomla!, esté preparado para su uso junto a MySQL, y la utilización de otro gestor como PostgreSQL (principal alternativa planteada) exija modificaciones innecesarias con MySQL.

  - **PostgreSQL**:

    PostgreSQL era con bastante seguridad un gestor de bases de datos más completo que MySQL en sus versiones anteriores a la 4.0. A partir de la versión 4.0 y de la utilización por defecto de la tecnología de almacenamiento de datos InnoDB para las tablas, MySQL empieza a incorporar numerosas características que muchos desarrolladores echaban en falta: vistas, foreign keys, procedimientos almacenados, transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), etc.

    Aunque el desarrollo de TB podría perfectamente haberse realizado utilizando PostgreSQL, finalmente se decidió no utilizarla en favor de MySQL.

  - **SQLLite**:

    SQLLite integra el motor de la base de datos directamente en los programas, haciendo el acceso a la base de datos muy rápido y eficaz dado que no es necesario utilizar un proceso a parte para el mismo. Además SQLLite es bastante sencillo. Guarda las bases de datos en un solo fichero, su configuración es mínima y simplemente requiere la instalación del driver puesto que no existe un servidor ejecutándose por detrás como en el caso del resto de gestores de bases de datos comentados.

  Además, existen librerías de SQLLite tanto para C/C++ como para PHP (a partir de la

versión 5 de este lenguaje).

SQLLite también presenta desventajas que han hecho que se decida no utilizarla:

- No existen tipo de datos para SQLLite, lo cual puede entrañar problemas durante la codificación (aunque esta característica tampoco es determinante, de hecho la aplicación web del proyecto está desarrollada con PHP, lenguaje que no es fuertemente tipado).
- Una característica de mayor importancia es que las transacciones en la base de datos exigen el bloqueo del fichero entero, con lo cual las aplicaciones multi-usuario que necesiten acceso concurrente a la base de datos pueden verse negativamente afectadas.

Aplicación TunnelBrokerRouter (TB-R)

Proceso

TB-R se mantiene a la escucha esperando las peticiones de conexión enviadas por TB-S para la creación o destrucción de túneles con los clientes.

La comunicación con TB-S se hace, al igual que en el resto de componentes con sockets y la utilización en la capa de aplicación del protocolo SSL.

El servidor implementado en TB-R procesa cada petición de TB-S en un proceso nuevo. Si bien sólo existe una aplicación TB-S en el sistema, la forma en que ésta está implementada permite que varios procesos en paralelo se comuniquen con un solo encaminador, es por ello que TB-R utiliza llamadas a fork para gestionar cada petición por separado.

Durante la comunicación establecida con un proceso de TB-S, el programa en el encaminador deberá realizar diversas funciones y comprobaciones:

TB-R deberá comprobar que el mensaje recibido es correcto y que contiene el nombre del interfaz a modificar, la opción que indica la acción a realizar, y la dirección IPv4 del cliente que se comunica a través del interfaz mencionado.

Una vez comprobado el mensaje, y si este es correcto según el protocolo especificado para la forma de los mensajes, TB-R realizará la siguiente acción:

- Si la acción a realizar es 'Up', se creará una túnel SIT con el cliente especificado por la dirección IPv4 y habilitando una interfaz de nombre tunnXX según se especifique en el mensaje.
- Si la acción a realizar es 'Down', se eliminará el túnel SIT de la interfaz tunnXX especificada en el mensaje y se deshabilitará dicha interfaz.

Modelos de desarrollo

Para la metodología de trabajo, se han aplicado los conocimientos adquiridos en IS durante el año pasado. Al final, se ha optado por un modelo de trabajo incremental.

Este modelo provee una estrategia para controlar la complejidad y los riesgos, desarrollando una parte del producto software reservando el resto de aspectos para el futuro.

Los principios básicos son:

- Una serie de mini-Cascadas se llevan a cabo, donde todas las fases de la cascada modelo de desarrollo se han completado para una pequeña parte de los sistemas, antes de proceder a la próxima incremental.
- Se definen los requisitos antes de proceder con la evolutivo, se realiza un mini-Cascada de desarrollo de cada uno de los incrementos del sistema.
- El concepto inicial de software, análisis de las necesidades, y el diseño de la arquitectura y colectiva básicas se definen utilizando el enfoque de cascada, seguida por iterativo de prototipos, que culmina en la instalación del prototipo final.

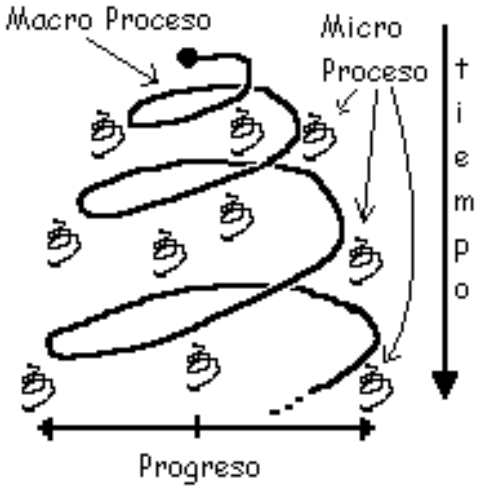


Imagen 16: Modelo de desarrollo incremental.

De este modo, conforme pasaba el tiempo, se iba teniendo un producto más complejo y se iban cumpliendo los requisitos pactados.

Herramientas utilizadas.

Subversion.

Se ha utilizado el subversion por ser una aplicación muy útil, funcional y potente.

Subversion es un software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como svn por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

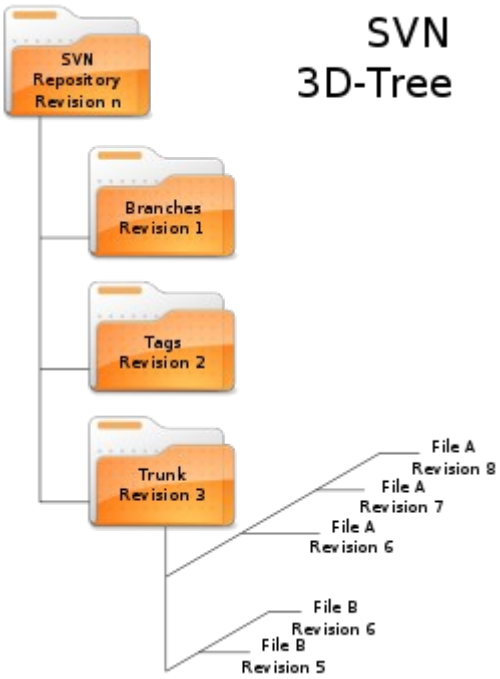


Imagen 17: Ejemplo de carpetas en el SVN.

Ventajas

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente; Tiene costo de complejidad constante (O(1)) y no lineal (O(n)) como en CVS.
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor

archivos completos).

- Puede ser servido mediante Apache, sobre WebDAV/DeltaV. Esto permite que clientes WebDAV utilicen Subversion en forma transparente.
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.
- Cuando se usa integrado a Apache permite utilizar todas las opciones que este servidor provee a la hora de autentificar archivos (SQL, LDAP, PAM, etc.).

Carencias

- El manejo de cambio de nombres de archivos no es completo. Lo maneja como la suma de una operación de copia y una de borrado.
- No resuelve el problema de aplicar repetidamente parches entre ramas, no facilita el llevar la cuenta de qué cambios se han trasladado. Esto se resuelve siendo cuidadoso con los mensajes de commit. Esta carencia será corregida en la próxima versión (1.5).

El cliente que se ha utilizado entre los disponibles es RapidSVN que corre en Linux.

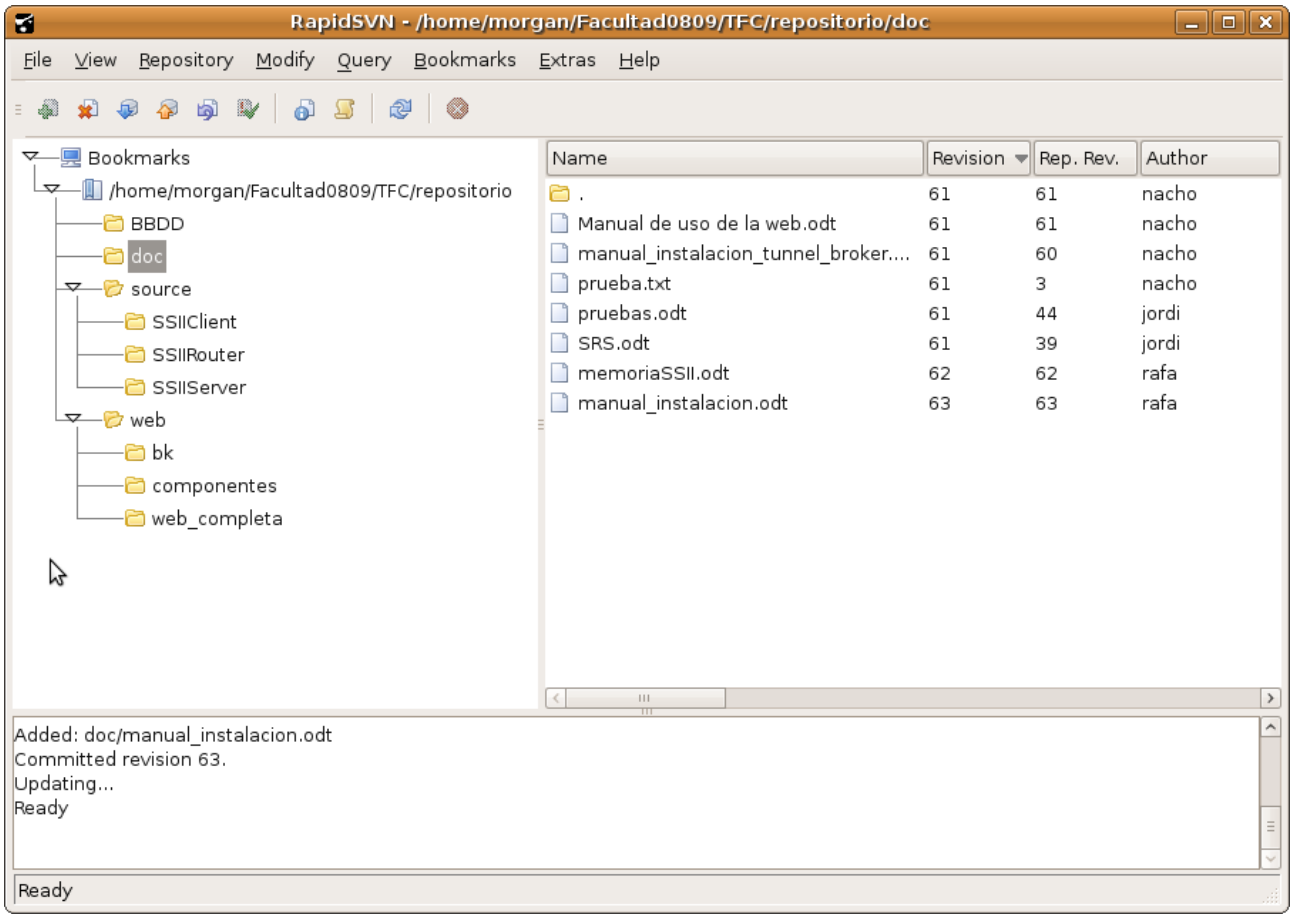


Imagen 18: Pantalla de la interfaz del cliente svn, RapidSVN.

Bibliografía

- The User-mode Linux Kernel Home Page:
  - URL: http://user-mode-linux.sourceforge.net/
- Uso de UML.
  - Editorial: Asignatura Laboratorio de Redes de la FDI de la UCM.
  - Autor: Juan Carlos Fabero Jiménez.
- Programación Orientada a Objetos.
  - Editorial: Asignatura de la FDI de la UCM.
  - Autor: Luis Hernández Yáñez.
- Cryptography and Network Security.
  - Editorial: Prentice Hall
  - Autor: William Stallings.
- Programación Linux
  - Editorial: Anaya Multimedia.
  - Autor: Neil Matthew y Richard Stones.
- Mastering Joomla! 1.5 Extension and Framework Development.
  - Editorial: Packt
  - Autor: James Kennard.
- PHP and MySQL Web Development (3rd Edition)
  - Editorial: Sam's publishing
  - Autor: Luke Welling y Laura Thompson

Autorización

Los integrantes del grupo Rafael Jiménez García, Ignacio Santabárbara Ruiz y Jordi Alcaide Romeu autorizan a la Universidad Complutense a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Rafael Jiménez García

Ignacio Santabárbara Ruiz

Jordi Alcaide Romeu

Resumen de la memoria

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

Entorno del sistema

User Mode Linux

De aquí en adelante UML. Para el desarrollo de las aplicaciones que exige el proyecto, se ha montado un sistema de máquinas virtuales sobre UML. User-Mode-Linux, instalado sobre un Linux, es un método seguro para ejecutar versiones de Linux y sus procesos.

User-Mode-Linux proporciona una máquina virtual que puede tener más recursos hardware y software virtuales que la máquina actual, la máquina física. El almacenamiento en disco para la máquina virtual está contenido completamente en un único fichero en la máquina física. Se puede asignar a la máquina virtual el acceso al hardware que solamente se desee. Con los permisos correctamente configurados, no se puede hacer nada en la máquina virtual que pueda dañar a la máquina física real o a su software.

Internet Protocol

El Protocolo de Internet es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados, provee un servicio de datagramas no fiable. Si se necesita fiabilidad, ésta es proporcionada por los protocolos de la capa de transporte, como TCP.

Los datos en una red basada en IP son enviados en bloques conocidos como paquetes o datagramas (en el protocolo IP estos términos se suelen usar indistintamente). En particular, en IP no se necesita ninguna configuración antes de que un equipo intente enviar paquetes a otro con el que no se había comunicado antes.

Si la información a transmitir ("datagramas") supera la unidad de tamaño máximo (MTU, que viene fijada por la tecnología) en el tramo de red por el que va a circular podrá ser dividida en paquetes más pequeños, y reensamblada luego cuando sea necesario. Estos fragmentos podrán ir cada uno por un camino diferente dependiendo de como estén de congestionadas las rutas en cada momento.

El IP es el elemento común en la Internet de hoy. El actual y más popular protocolo de red es IPv4. IPv6 es el sucesor propuesto de IPv4.

Las versiones de la 0 a la 3 están reservadas o no fueron usadas. La versión 5 fue usada para un protocolo experimental. Otros números han sido asignados, usualmente para protocolos experimentales, pero no han sido muy extendidos.

Dirección IP

Una dirección IP es un número que identifica de manera lógica y jerárquicamente a una interfaz de un dispositivo dentro de una red que utilice el protocolo de Internet.

Es obligatorio que un usuario que se conecta desde su hogar a Internet utilice una dirección IP. Esta dirección puede cambiar al reconectar. A ésta forma de asignación de la dirección IP se denomina dirección IP dinámica (normalmente se abrevia como IP dinámica).

Los sitios de Internet que por su naturaleza necesitan estar permanentemente conectados, generalmente tienen una dirección IP fija (se aplica la misma reducción por IP fija o IP estática), es decir, no cambia con el tiempo. Los servidores de correo, DNS, FTP públicos, servidores web necesariamente deben contar con una dirección IP fija o estática, ya que de esta forma se facilita su ubicación.

Existen protocolos para asignar direcciones IP dinámicas, como por ejemplo DHCP (Dynamic Host Configuration Protocol).

IPv4 (RFC0791)

IPv4 es la versión 4 del Protocolo IP (Internet Protocol).
IPv4 usa direcciones de 32 bits, limitándola a 2^{32} = 4.294.967.296 direcciones únicas, muchas de las cuales están dedicadas a redes locales (LANs). Por el crecimiento enorme que ha tenido del Internet, combinado con el hecho de que hay desperdicio de direcciones en muchos casos, ya hace varios años se vio que escaseaban las direcciones IPv4.
Esta limitación ayudó a estimular el impulso hacia IPv6, que esta actualmente en las primeras fases de implantación, y se espera que termine reemplazando a IPv4.

IPv6 (RFC2460)

El protocolo IPv6 es una nueva versión de IP (Internet Protocol), diseñada para reemplazar a la versión 4 (IPv4). El nuevo estándar mejorará el servicio globalmente; por ejemplo, proporcionará a futuras celdas telefónicas y dispositivos móviles con sus direcciones propias y permanentes. Al día de hoy se calcula que las dos terceras partes de las direcciones que ofrece IPv4 ya están asignadas.

IPv6 admite 340.282.366.920.938.463.463.374.607.431.768.211.456 (2^{128} o 340 sextillones) direcciones —cerca de $3{,}4 \times 10^{20}$ (340 trillones) direcciones por cada pulgada cuadrada (6,7 × 1017 o 670 mil billones direcciones/mm2) de la superficie de La Tierra.

Shell Scripts.

Los shell scripts son programas escritos con comandos UNIX y son equivalentes a los batch de DOS aunque mucho más potentes, pues admiten ejecuciones en segundo plano y tienen un conjunto de expresiones mucho más amplio.

Una de las ventajas que presentan los shell scripts es que pueden ser portadas de una máquina UNIX a otra sin problemas, sin necesidad de retocar nada, salvo que se utilicen llamadas a

programas muy concretos específicos de una versión de UNIX. Otra ventaja es la facilidad de lectura e interpretación.

El principal inconveniente que presentan respecto a los programas compilados es la lentitud de ejecución, que se puede paliar usando las utilidades built-in incluidas en el propio kernel en lugar de llamar a comandos externos que han de ser leídos de disco. Otro inconveniente es que el código resulta visible a cualquier usuario que lo pueda ejecutar.

De cara a nuestro ámbito, se ha generado una jerarquía en árbol en la cual, el usuario debe hacer una llamada inicial a uno de los script y, a partir de ahí, se genera una secuencia de llamadas que hace que se ejecute el conjunto completo. El motivo por el que se usan en el proyecto es porque permiten automatizar una configuración compleja que realizándose a mano, supondría una pérdida de tiempo e ineficacia al tener que ejecutarla en varias ocasiones. De este otro modo, con un sólo script., se desencadena una serie de llamadas a otros scripts y la configuración se lleva a cabo sin errores y en una cantidad de tiempo mucho menor.

Túneles SIT

De entre los túneles disponibles (ipip, sit, gre, isatap) se ha elegido el túnel sit porque es el que más se adecúa al objetivo del proyecto, comunicar dos islas IPv6 a través de IPv4.

Para empezar, SIT (significa Simple Internet Transition) significa transición simple de internet.

SIT dispone de un driver que implementa la encapsulación de IPv6 dentro de paquetes IPv4.

Encaminador

Un encaminador es un dispositivo de interconexión de redes informáticas que permite asegurar el encaminamiento de paquetes entre redes o determinar la ruta que debe tomar el paquete de datos.

Cuando un usuario accede, por ejemplo, a una URL, el cliente web (navegador) consulta al servidor de nombre de dominio, el cual le indica la dirección IP del equipo deseado.

La estación de trabajo envía la solicitud al encaminador más cercano, es decir, a la pasarela predeterminada de la red en la que se encuentra. Este encaminador determinará así el siguiente equipo al que se le enviarán los datos para poder escoger la mejor ruta posible.

Además de su función de encaminar, los encaminadores también se utilizan para manipular los datos que circulan en forma de datagramas, para que puedan pasar de un tipo de red a otra. Como no todas las redes pueden manejar el mismo tamaño de paquetes de datos, los encaminadores deben fragmentar los paquetes de datos para que puedan viajar libremente.

Red

Una red de computadoras es un conjunto de equipos (computadoras y/o dispositivos) conectados

por medio de cables, señales, ondas o cualquier otro método de transporte de datos, que comparten información, recursos y servicios, etc.

Para simplificar la comunicación entre programas (aplicaciones) de distintos equipos, se definió el Modelo OSI por la ISO, el cual especifica 7 distintas capas de abstracción. Con ello, cada capa desarrolla una función específica con un alcance definido.

Host

Se define como una máquina conectada a una red de ordenadores y que tiene una dirección ip asociada. Es una dirección ip única que se le da a un dispositivo conectado a una red informática. Puede ser un ordenador, un servidor de archivos, un dispositivo de almacenamiento por red, una máquina de fax, impresora, etc.

Iptables

Es el componente más popular construido sobre Netfilter. Iptables es una herramientas de cortafuegos que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPv4 o mantener registros de log. El proyecto ofrecía compatibilidad hacia atrás con ipchains hasta hace relativamente poco, aunque hoy día dicho soporte ya ha sido retirado al considerarse una herramienta obsoleta.

Netfilter es un framework disponible en el núcleo Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento. Netfilter es también el nombre que recibe el proyecto que se encarga de ofrecer herramientas libres para cortafuegos basados en Linux.

El proyecto Netfilter no sólo ofrece componentes disponibles como módulos del núcleo sino que también ofrece herramientas de espacio de usuario y librerías.

Forwarding

Forwarding es una transmisión de los paquetes a través de un segmento de la red hacia otro a través de los nodos (máquinas físicas) en una red de ordenadores.

Para que esto funcione se debe tener activado el forwarding para que se transmitan.

Bridge (Puente de Red)

Un puente o bridge es un dispositivo de interconexión de redes de ordenadores que opera en la capa 2 (nivel de enlace de datos) del modelo OSI. Este interconecta dos segmentos de red (o divide una red en segmentos) haciendo el pasaje de datos de una red hacia otra, con base en la dirección física de destino de cada paquete.

Un bridge conecta dos segmentos de red como una sola red usando el mismo protocolo de establecimiento de red.

Sockets

Un socket es un concepto abstracto por el cual dos programas pueden intercambiarse cualquier flujo de datos. Queda definido por una dirección IP, un protocolo y un número de puerto.

Los sockets permiten implementar una arquitectura cliente-servidor.

Un socket es un fichero existente en el cliente y en el servidor, que sirve en última instancia para que el programa servidor y el cliente lean y escriban la información. Esta información será la transmitida por las diferentes capas de red.

Secure Socket Layer (SSL)

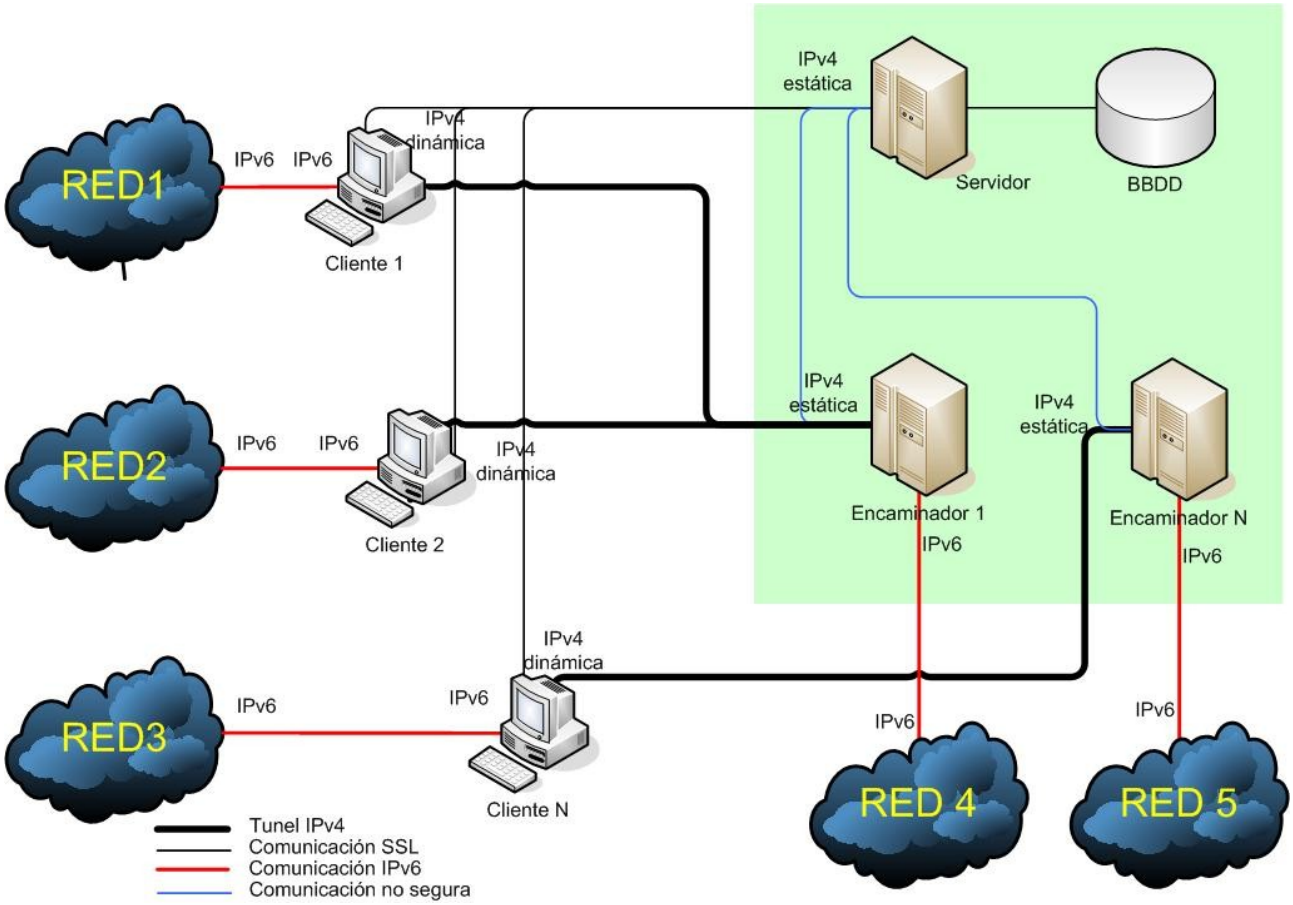
El protocolo SSL es un sistema diseñado y propuesto por Netscape Communications Corporation. Se encuentra en la pila OSI entre los niveles de TCP/IP y de los protocolos HTTP, FTP, SMTP, etc. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA.

Proporciona cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

Estructura.

Sistema emulado

El sistema de máquinas que se ha diseñado que a gran escala y teniendo en cuenta máquinas físicas reales y no UMLs, será.



Interfaz de Gestión

Introducción a los requisitos de la interfaz

Los requisitos del proyecto exigen que un usuario cualquiera se pudiera dar de alta en el servicio de túneles a través de una interfaz web, por otro lado, no se especificaba cómo debía ser la interfaz que gestionase el resto de funcionalidades. Por ello, se abren múltiples posibilidades, ya que se puede solucionar el problema de muchas formas.

Elección del framework

Una vez descartadas todas las opciones que no supusieran realizar un página web, se debía tomar una decisión sobre el modo de generar la página web. La solución elegida fue:

- Utilizar el framework Joomla!1.5.x

Joomla! además de un framework escrito en PHP, es un gestor de contenidos, es decir, es una aplicación que permite a los usuarios contribuir con artículos y a los administradores gestionar todo ese contenido generado por los usuarios.

Aplicaciones TunnelBroker

Introducción

El sistema TunnelBroker permite la conexión automática entre redes independientes que funcionan sobre el protocolo IPv6. Para conseguir dicha conexión una red conectada por IPv6 que compone una isla independiente, deberá tener configurado un terminal que le de acceso a través de un túnel a otro terminal que hará el papel de encaminador hacia algún backbone de internet IPv6.

Una vez que el usuario se ha dado de alta en el sistema a través del interfaz web ofrecida a tal fin, y ha obtenido un rango de direcciones IPv6 para la configuración de su isla, podrá instalar en un terminal de su red la aplicación TunnelBrokerClient para que el sistema le ofrezca acceso automático a algún backbone de internet IPv6 a través de túneles con los encaminadores propios del mismo.

El funcionamiento básico del sistema de conexión se basa en tres componentes:

- **TunnelBrokerClient**: Se instala en la red configurada por el usuario. El terminal donde va instalado TunnelBrokerClient es uno de los extremos del túnel a través del cual viajarán los paquetes IPv6 encapsulados en paquetes IPv4 para aprovechar la infraestructura del internet actual basado en IPv4. TunnelBrokerClient se comunica con TunnelBrokerServer para crear y mantener túneles con los encaminadores que le proporcionan acceso a Internet IPv6.
- **TunnelBrokerServer**: Instalado en una máquina accesible a través de Internet IPv4 y con dirección IPv4 estática. La aplicación TunnelBrokerServer mantiene un control de las conexiones existentes entre clientes y encaminadores. Mediante el envío de keep-alives las aplicaciones TunnelBrokerClient mantienen viva la conexión con el encaminador, además de proveer la información de su dirección IPv4 por si esta hubiese sido modificada.
- **TunnelBrokerRouter**: El encaminador se encarga de hacer, rehacer y deshacer los túneles con los clientes que le haya asignado la aplicación TunnelBrokerServer. El encaminador estará siempre a la escucha por si la aplicación TunnelBrokerServer le comunica algún cambio en los túneles creados actualmente o por si le comunica la necesidad de crear un nuevo túnel con un cliente.

Comunicaciones entre componentes

Introducción

La seguridad de las comunicaciones entre los tres componentes del sistema es un requisito indispensable. Una comunicación insegura que permitiese el robo de información por un tercero utilizando un sniffer o una comunicación en la que se pudiese suplantar la identidad de uno de los actores impiden una implantación mínimamente realista del sistema.

Posibles vulnerabilidades

Man in the middle

Ataque consistente en leer y también modificar la comunicación entre dos extremos sin que estos tengan constancia de dicho ataque.

Denegación de servicio

Ataque consistente en impedir que un recurso en una red sea inaccesible a los usuarios que pretenden utilizarlo.

Reenvío de datos cifrados

Otro ataque que podría tener lugar es la captura de datos cifrados, y el reenvío de los mismos tiempo después aunque esta situación no podrá darse, puesto que TunnelBrokerClient y TunnelBrokerServer, gracias al protocolo SSL, cifran de forma distinta cada conexión que establecen entre ellos, de tal modo que un mensaje cifrados en una conexión, no valdrá tiempo después puesto que será necesario rehacer el handshake.

Comunicación segura

Las comunicaciones entre los distintos componentes del sistema, como ya se ha mencionado anteriormente, utilizan el protocolo SSL. La utilización de dicho protocolo permite asegurar tanto la seguridad como, en el caso de requerirlo, la autenticación de los componentes.

Trazas de las aplicaciones

Puesto que las aplicaciones TunnelBroker se ejecutan en modo background sin participación directa del usuario, se ha considerado importante poder hacer un seguimiento de los procesos en ejecución para analizar fallos, funcionamiento o simplemente para obtener información de uso de las mismas. Para realizar este seguimiento se ha decidido escribir trazas descriptivas del proceso durante la ejecución del mismo.

Aplicación TunnelBrokerClient

Proceso

Se encarga de solicitar al servidor TunnelBroker la creación de una conexión, para que este le proporcione un encaminador y avise al encaminador de que debe crear el túnel con la IPv4 proporcionada por el cliente.

Las conexiones consisten en dos túneles SIT, uno creado desde la aplicación TunnelBrokerClient hacia la aplicación TunnelBrokerRouter y otro en sentido opuesto.

El programa cliente TunelBroker deberá encargase de mantener la conexión enviando keep-alives al servidor con la IPv4 actual. Si esta cambiase, la aplicación TunnelBrokerRouter será avisada para rehacer el túnel SIT hacia la aplicación TunnelBrokerClient.

Aplicación TunnelBrokerServer

Proceso

TunnelBrokerServer se mantiene a la escucha esperando las peticiones de los clientes TunnelBroker para gestionar las mismas.

Para la comunicación con los clientes TunnelBroker se utilizan, como se ha explicado en detalle previamente, sockets sobre el protocolo orientado a conexión TCP y añadiendo además en la capa de aplicación del modelo OSI el protocolo SSL para garantizar la seguridad en la comunicación.

El servidor implementado procesa cada petición de un cliente ejecutando un proceso nuevo e independiente.

Durante la comunicación establecida con un cliente, el servidor deberá realizar diversas funciones y comprobaciones.

El servidor deberá comprobar primero que el mensaje recibido es correcto y que contiene un nombre de usuario, una contraseña y una dirección IPv4. Tras ello deberá comunicarse con el gestor de bases de datos para asegurarse que el usuario y la contraseña son válidos.

Si el cliente proporcionase al router un par usuario/contraseña incorrectos, se devolverá un error al mismo. Y el proceso terminará.

Si el cliente se autentica correctamente el proceso de comprobaciones continua.

Se pasa a comprobar si el usuario tiene alguna conexión activa, pudiendo darse tres posibilidades:

- El cliente no tiene ninguna conexión activa: en este caso se le asigna un encaminador y se le envía la dirección IPv4 del mismo al cliente y la dirección IPv4 del cliente al encaminador.

- El cliente tiene una conexión pero su dirección IPv4 ha cambiado: se le envía un mensaje al encaminador para rehacer el túnel SIT con la nueva dirección IPv4.
- El cliente tiene una conexión y su dirección IPv4 se mantiene igual: se le envía un mensaje informando de que todo se mantiene correctamente al cliente y al encaminador no se le envía nada.

Aplicación TunnelBrokerRouter

Proceso

TunnelBrokerRouter se mantiene a la escucha esperando las peticiones de conexión enviadas por TunnelBrokerServer para la creación o destrucción de túneles con los clientes.

La comunicación con TunnelBrokerServer se hace, al igual que en el resto de componentes con sockets y la utilización en la capa de aplicación del protocolo SSL.

El servidor implementado en TunnelBrokerRouter procesa cada petición de TunnelBrokerServer en un proceso nuevo. Si bien sólo existe una aplicación TunnelBrokerServer en el sistema, la forma en que ésta está implementada permite que varios procesos en paralelo se comuniquen con un solo encaminador, es por ello que TunnelBrokerRouter utiliza llamadas a fork para gestionar cada petición por separado.

Durante la comunicación establecida con un proceso de TunnelBrokerServer, el programa en el encaminador deberá realizar diversas funciones y comprobaciones:

TunnelBrokerRouter deberá comprobar que el mensaje recibido es correcto y que contiene el nombre del interfaz a modificar, la opción que indica la acción a realizar, y la dirección IPv4 del cliente que se comunica a través del interfaz mencionado.

Una vez comprobado el mensaje, y si este es correcto según el protocolo especificado para la forma de los mensajes, TunnelBrokerRouter realizará la siguiente acción:

- Si la acción a realizar es 'Up', se creará una túnel SIT con el cliente especificado por la dirección IPv4 y habilitando una interfaz de nombre tunnXX según se especifique en el mensaje.
- Si la acción a realizar es 'Down', se eliminará el túnel SIT de la interfaz tunnXX especificada en el mensaje y se deshabilitará dicha interfaz.

Modelo de desarrollo

Se ha optado por un modelo de trabajo incremental.

Este modelo provee una estrategia para controlar la complejidad y los riesgos, desarrollando una parte del producto software reservando el resto de aspectos para el futuro.

Los principios básicos son:

- Una serie de mini-Cascadas se llevan a cabo, donde todas las fases de la cascada modelo de desarrollo se han completado para una pequeña parte de los sistemas, antes de proceder a la próxima incremental.
- Se definen los requisitos antes de proceder con la evolutivo, se realiza un mini-Cascada de desarrollo de cada uno de los incrementos del sistema.
- El concepto inicial de software, análisis de las necesidades, y el diseño de la arquitectura y colectiva básicas se definen utilizando el enfoque de cascada, seguida por iterativo de prototipos, que culmina en la instalación del prototipo final.

Herramienta utilizada

Subversion

Se ha utilizado el subversion por ser una aplicación muy útil, funcional y potente.

Subversion es un software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como svn por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de Subversion es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

El cliente que se ha utilizado entre los disponibles es RapidSVN que corre en Linux.

Especificación de requisitos según el estándar IEEE 830-1998

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

1. Introducción

1.1. Propósito

Este documento presenta, en castellano, la especificación de requisitos del proyecto de forma clara.

El documento se dirige tanto al tutor de proyecto para su revisión y visto bueno, como a los miembros del grupo de desarrollo para la realización del diseño e implementación.

1.2. Ámbito del Sistema

El proyecto se llamará TunnelBroker.

La aplicación pretende conectar 2 islas IPv6 a través de una infraestructura IPv4. Para ello deberá utilizar túneles entre un host de una isla y un host de la otra.

En una de las islas se encontrará al cliente, cuya IPv4 es dinámica. Dicha isla IPv6 tendrá un prefijo generado por la propia aplicación y solo tendrá un host de acceso.

En la otra isla se encontrará al host de acceso a la troncal de IPv6. Dicho acceso puede no ser único, de forma que se pueda balancear la carga. Por tanto, podrán existir varios host que permitan el acceso a la misma isla, estos hosts reciben el nombre de encaminadores destino.

La aplicación deberá ser segura, para ello el cliente deberá autenticarse con una conexión cifrada.

La aplicación deberá tener una interfaz gráfica con la que se pueda gestionar a clientes, encaminadores destino y prefijos.

Con este proyecto, los clientes podrán tener configurada su red con IPv6 sin esperar a que los ISP se decidan a dar soporte del mismo. Además será compatible con IPv4 dinámicas.

1.3. Definiciones, Acrónimos y Abreviaturas

BDD: Base de datos.

HTTPS: Hypertext Transfer Protocol Secure

LAN: Local Area Network.

MySQL: Sistema de gestión de base de datos relacional, multihilo y multiusuario.

SSL: Secure Socket Layer. Protocolos criptográficos de la capa de transporte que proporcionan comunicaciones seguras por una red.

SNMP: Simple Network Management Protocol.

Túnel SIT: encapsula datagramas IPv6 dentro de datagramas IPv4 para poder enviar estos sobre una infraestructura de red IPv4.

2. Descripción General

2.1. Perspectiva del Producto

Aplicación depende por un lado de la IPv6 (en proceso de implantación) y por otro lado de IPv4. En cuanto desaparezca IPv4, la aplicación dejará de cobrar sentido.

2.2. Funciones del Producto

Entre las funciones del lado del cliente se encuentran:

- Detección de cambio de IPv4 en el cliente y comunicación con el servidor central.
- Asignación de un prefijo de la longitud requerida al cliente.
- Envío de información cifrada en la autenticación.
- Envío del keep-alive al servidor central.
- Generación del túnel con el encaminador destino.

Entre las funciones del lado del servidor central se encuentran:

- Autenticación y comunicación con clientes.
- Comunicación con el/los encaminador/es destino.
- Generación de prefijos a los clientes mediante 3 algoritmos diferentes.
- Alojamiento de una aplicación web donde los clientes se puedan dar de alta.
- Alojamiento de la BDD.

Entre las funciones del lado del encaminador destino se encuentran:

- Comunicación con el servidor central
- Regeneración de los túneles con los clientes

2.3. Características de los Usuarios

Los usuarios del producto deberán desenvolverse con soltura en el uso de las nuevas tecnologías, en concreto deberán tener unas nociones básicas sobre IPv6.

2.4. Restricciones

Políticas de la empresa

El grupo estará compuesto por tres miembros, según indica la normativa de la asignatura “Sistemas Informáticos” aprobada por la Junta de la Facultad de Informática en su sesión del 5 de noviembre de 2008.

Limitaciones del hardware

El aplicación está diseñada para funcionar con IPv4 dinámicas. Para ello es necesario poder conocer la IPv4 pública en todo momento.

La aplicación podrá acceder a dicha información de 2 formas, y en el caso de no poder obtener la información requerida por ninguna de las vías, la aplicación no podrá funcionar correctamente.

Las 2 formas para conocer la IPv4 dinámica pública son:

- Mediante el protocolo SNMP
- Mediante un servidor externo ajeno a nuestras competencias.

Para la primera forma, el router del cliente debe soportar SNMP y ser capaz de enviar la IPv4 por dicho protocolo.

La segunda forma, depende de un tercero, y puede no estar siempre disponible. Además, pueden surgir problemas si se utiliza un proxy.

Operaciones paralelas

El sistema proporciona una conexión entre 2 islas. Como estén configuradas las islas internamente queda fuera del ámbito de la aplicación

Funciones de auditoría y de control

El proyecto se someterá en todo momento a un seguimiento y evaluación por parte del profesor director.

Lenguaje(s) de programación

Se utilizarán los lenguajes de programación C y C++, no siendo una restricción el entorno de desarrollo a utilizar.

Para el almacenamiento de datos se utilizará un sistema gestor de bases de datos, siendo SQL el lenguaje a utilizar en el mismo. Como sistema gestor se utilizará MySQL.

La aplicación web existente en el sistema se desarrollará utilizando el conjunto de subsistemas software comúnmente conocido como LAMP (Linux-Apache-MySQL-PHP-Phyton-Perl).

Protocolos de comunicación

Para realizar las conexiones que requieren un mínimo grado de seguridad se utilizará el protocolo SSL (Protocolo de Capa de Conexión Segura), y el protocolo HTTPS (Hypertext Transfer Protocol Secure).

Las comunicaciones que no requieran de ningún tipo de seguridad especial se desarrollaran sobre el protocolo TCP/IP usual.

Requisitos de habilidad

Solo se requerirá habilidad a la hora de la configuración del sistema, ya que es necesario unos conocimientos mínimos en redes.

Criticalidad de la aplicación

No dependerán vidas humanas o de cualquier otro ser vivo del funcionamiento de la aplicación. Aun así, el funcionamiento de la aplicación es crítico, ya que de ella depende la comunicación de toda una isla IPv6.

Consideraciones acerca de la seguridad

El registro de usuarios y la modificación de la dirección IPv4 en la BDD son operaciones críticas que deberán ir cifradas.

Si un programa o individuo malintencionado consigue acceder a los datos guardados en la BDD, el sistema dejará de ser seguro. Será responsabilidad del administrador de la misma la realización de respaldos (backups), la seguridad a través de conexiones externas al servidor y la seguridad física del equipo en que se encuentra la misma.

2.5. Suposiciones y Dependencias

El sistema se ha desarrollado y probado principalmente sobre el sistema operativo GNU/Linux.

- Versión del núcleo: 2.6.24 .
- Distribución para el desarrollo: Ubuntu 8.x
- Distribución en las máquinas virtuales User Mode Linux: Debian Lenny.

Dado que el desarrollo y pruebas se han realizado como se mencionaba previamente, se considerará necesaria para el correcto funcionamiento del sistema la utilización de alguno de los sistemas operativos descritos. Siendo recomendable su uso en un sistema GNU/Linux basado en la distribución Debian.

2.6. Requisitos Futuros

En un futuro se podría ampliar para mantener túneles SIT entre varios clientes con IPv4 dinámicas.

Se podrá añadir nuevos algoritmos para repartir la carga entre los distintos encaminadores destino en función de nuevas variables más complejas, como la carga real.

3. Requisitos Específicos

Cliente

1. El sistema debería detectar los cambios en la IPv4 pública de la interfaz en caso de que dicha IPv4 sea dinámica. El cliente deberá configurar la aplicación indicando si la IPv4 es dinámica o no. En el caso de que sea dinámica, el cliente deberá poder obtener la IPv4 pública de alguna de las siguientes formas:

   a. Mediante el protocolo SNMP

   b. Mediante un servidor externo.

   Si por algún motivo, el cliente no puede obtener la información requerida por ninguno de los 2 medios, la aplicación no podrá funcionar con IPv4 dinámicas, es decir, sólo podrá funcionar con IPv4 estáticas.

2. El sistema deberá tener una interfaz web que se conecte de modo seguro al servidor (SSL) y permita al usuario registrarse en éste y obtener un rango de direcciones IPv6 para configurar su LAN de direcciones IPv6. No será necesario que la LAN de direcciones IPv6 se configure automáticamente, bastará con mostrar al usuario el rango asignado para que éste realice la configuración manualmente empleando las ordenes adecuadas a su S.O..

3. El sistema debería crear automáticamente túneles SIT con la dirección IPv4 del encaminador destino que le mande el servidor. Cuando existan cambios en la IPv4 pública del cliente, deberá rehacer el túnel con la dirección IPv4 destino, y avisar a través de una conexión segura (SSL) al servidor del cambio producido para que este pueda avisar al destino de que es necesario rehacer el túnel.

4. Los túneles SIT no requieren participación activa del destinatario del túnel, puesto que es su interfaz IPv4 quien recibe los paquetes IPv6 encapsulados. Por ello, cuando se deshace o rompe un túnel, el cliente sólo debe preocuparse de rehacerlo en el sentido de salida.

Será el propio destino quien se encargue de rehacerlo en el sentido inverso.

Servidor

1. Deberá implementar una interfaz web que permita el registro de usuarios. Deberá funcionar con una conexión segura (https). Deberá insertar en la BDD el nuevo usuario, y generarle un prefijo si éste lo desea. Además, deberá permitir al usuario consultar sus datos y modificarlos, mediante el registro de su nombre de usuario y su palabra clave.
2. Deberá escuchar los avisos del cliente respecto a su dirección IPv4 pública por conexión segura y modificar la base de datos cuando se produzcan dichos avisos. También deberá avisar al encaminador destino del cambio, para que éste pueda rehacer el túnel SIT con el cliente. El encaminador destino deberá confirmar que el túnel ha sido rehecho, si no, se seguirán mandando avisos.
3. Deberá enviar la dirección del encaminador destino al cliente en el caso de que las direcciones del Servidor y del encaminador destino sean diferentes. Dicha dirección podrá cambiar de una conexión a otra, pero nunca en medio de una conexión.

Encaminador destino

1. Deberá estar preparado para recibir los avisos del servidor para crear o regenerar el túnel con el cliente. Si no se instala el programa en el encaminador destino, la comunicación no podrá realizarse bidireccionalmente a través de los túneles. Deberá avisar al servidor de que el túnel ha sido hecho o rehecho.
2. Si se quiere comunicar con otros encaminadores destino, el administrador tendrá que configurarlo ya que la aplicación no se encarga de dicha configuración.
3. La dirección IPv4 del encaminador destino no podrá cambiar a lo largo de toda la conexión con el cliente.

3.1. Interfaces Externas

3.1.1 Cliente

El programa cliente no provee funcionalidad para poder registrarse en el servicio ni tampoco para modificar la configuración en el servidor. De forma muy resumida el programa cliente se ocupa de detectar la IP pública, rehacer los túneles y mandar los keep-alive, por ello, el programa cliente no tendrá interfaz gráfica, para configurarlo tan solo hay que rellenar los parámetros en el fichero de configuración.

3.1.2 Servidor

El servidor tendrá instalado 4 aplicaciones distintas cómo mínimo:

- El sistema gestor de base datos Mysql.
- El servidor web Apache2 con soporte para PHP5
- La aplicación web de gestión de la BDD.
- La aplicación servidor que atiende al programa del cliente y al programa del encaminador destino.

La configuración de los distintos componentes del sistema instalados en el servidor se realizará de diversas formas dependiendo de su naturaleza.

- Para aquellos componentes desarrollados por terceros (Sistema gestor de bases de datos, Servidor Web, etc.) la configuración se realizará utilizando las herramientas existentes para ello, no siendo necesario el desarrollo de ninguna herramienta extra.
- Para aquellos componentes desarrollados exclusivamente en el marco del proyecto propuesto en estos requisitos, la forma en la que se podrá configurar las aplicaciones dependerá del tipo de la aplicación.

Para la parte de gestión, la configuración se hara mediante la interfaz web o modificando directamente los ficheros de configuración de la misma.

Para el resto, al ser programas demonio, la configuración se hará exclusivamente modificando los ficheros de configuración de cada uno de ellos.

3.1.3 Encaminador destino

No serán necesarias interfaces externas para el encaminador destino, su configuración, de ser necesaria, se hará a través de un fichero de configuración.

3.1.4 Resto de aplicaciones

Para el buen funcionamiento de la aplicación, es necesario utilizar componentes externas al proyecto, como por ejemplo, el servidor web apache, el sistema gestor de base de datos Mysql, etc.

Todas las componentes utilizadas, se distribuyen con su propia interfaz, delegando la responsabilidad de la misma a los desarrolladores de la componente.

3.2. Funciones

3.2.1 Cliente

3.2.1.1 Obtención de la IPv4 dinámica pública del interfaz.

Obtención de la IPv4 pública del interfaz que utilizará el destino para crear el túnel SIT con el cliente.

Existen dos posibilidades contempladas:

I. Cliente conectado directamente a Internet: la obtención de la IPv4 pública deberá ser directa a través de las herramientas proporcionadas por el Sistema Operativo.

II. Cliente conectado a Internet a través de un router: en este caso, tal y como se contempla en el apartado de restricciones de este documento, la dirección se podrá obtener mediante 2 formas

    a. Mediante el protocolo SNMP

    b. Mediante un servidor externo ajeno a nuestras competencias.

    Existe la posibilidad de que no se pueda obtener dicha información

La diferenciación entre las dos posibilidades deberá especificarla el usuario por configuración de la aplicación.

3.2.1.2 Demonio para envíos de keep-alive

Una de las funciones que debe cumplir el cliente es el aviso periódico al servidor de la IPv4 pública que tiene en ese momento.

El demonio deberá crear una conexión segura (SSL) con el servidor y enviará un aviso de comprobación y actualización junto a la autenticación (usuario y contraseña) y la IPv4 que tiene en el momento actual.

3.2.1.3 Generación o Regeneración del túnel

Si la dirección del encaminador destino cambia o el cliente no la sabe, el servidor mandará la dirección al cliente para que genere o regenere el túnel.

3.2.2 Servidor

3.2.2.1 Aplicación web de registro

La aplicación web de registro debe permitir dar de alta a clientes desde un navegador web que cumpla los estándares de la W3C. Una vez el cliente haya sido dado de alta, el cliente podrá suscribirse al servicio de túneles rellenado un nuevo formulario con la longitud del prefijo y el tiempo de vida del mismo. Una vez concluido el proceso de alta, la interfaz web mostrará a el cliente el prefijo IPv6 asignado o, en caso de error, la descripción del mismo.

La aplicación web accederá a la base de datos para comprobar que el usuario no tiene otro prefijo asignado. Si el usuario no está dado de alta en el servicio de túneles, le registrará y le asignará un prefijo IPv6. Si por el contrario, el usuario ya se encuentra dado de alta en el servicio, se avisará de ello.

Adicionalmente, el cliente podrá borrar el prefijo IPv6 que se le haya asignado y modificar el tiempo de vida del mismo.

En la parte de la administración, el super administrador debe poder ser capaz de:

- Listar a los clientes.
- Bloquear a un cliente.
- Elegir el método de asignación de prefijos entre los algoritmos mejor ajuste, peor ajuste y primero el mejor.

- Dar de alta a los encaminadores destino.

3.2.2.2 Recepción de mensajes de los clientes

El servidor deberá tener instalado un demonio que se mantenga constantemente a la escucha para poder recibir los avisos keep-alive y las peticiones de registro de los clientes a través de una comunicación segura (SSL).

Dichos avisos deberán incluir la siguiente información:

- Nombre de usuario
- Contraseña
- IPv4 pública del cliente
- Tipo de aviso (keep-alive,consulta IPv4 encaminador destino,...)

3.2.2.2.1 Recepción de consultas

Si el mensaje recibido especifica una consulta, se comprueba que el usuario está dado de alta y se envían los datos asociados a dicho usuario.

3.2.2.2.2 Recepción de avisos keep-alive

Si el mensaje recibido especifica un tipo de aviso keep-alive, el demonio deberá comprobar en la BDD el nombre de usuario y la contraseña, después, una vez autentificada la petición de esta forma, se comprobará que la dirección IPv4 asociada a dicho cliente se mantiene igual. En caso de que existiese una modificación en la dirección IPv4 se deberá avisar al destino de los túneles (ver apartado 3.2.2.3 Envíos de avisos al destino).

Además, la recepción de estos avisos reinicializan el temporizador al valor por defecto.

El tiempo del temporizador deberá ser una variable fácilmente modificable con un valor por defecto de 300 segundos.

3.2.2.3 Envíos de avisos al encaminador destino

Cuando el servidor comprueba al recibir un mensaje keep-alive que un cliente ha cambiado su dirección IPv4, debe establecer un canal de

comunicación con el encaminador destino de los túneles SIT de dicho cliente y enviar la nueva dirección IPv4 del cliente. El canal de comunicación deberá ser seguro para evitar suplantación de identidad.

3.2.3 Encaminador destino

3.2.3.1 Recepción de avisos para reconstrucción de túneles SIT

El encaminador destino de los túneles de un cliente deberá tener instalado un demonio para la recepción de avisos del servidor con los cambios en la IPv4 pública del cliente de tal modo que al recibir un aviso con la IPv4 del cliente, el destino pueda rehacer el túnel SIT con la IPv4 recibida.

Para el caso en que el destino aún no hubiese creado ningún túnel SIT con el cliente, el comportamiento será el mismo exceptuando que no será necesario borrar el túnel SIT actualmente montado puesto que no existirá.

3.3. Requisitos de Rendimiento

La carga esperada para la aplicación en su conjunto no es muy elevada, ya que hay sólo dos operaciones que requieran mucho procesamiento:

- Generación de un prefijo IPv6
- Generación de la conexión SSL entre cliente y servidor.

El coste de la generación de un prefijo depende cuadráticamente del número de prefijos asignados, aunque dependiendo del algoritmo utilizado, se puede mejorar hasta que dependa linealmente del número de prefijos asignados. El administrador del sistema, puede que se vea obligado a utilizar un algoritmo menos costoso para mejorar el rendimiento del sistema.

3.4. Restricciones de diseño

No se presenta ninguna restricción en el diseño. Se puede utilizar diseño orientado a objetos o diseño estructurado a nuestra elección.

3.5. Atributos del Sistema

Portabilidad: La aplicación no será portable a otros sistemas operativos, al menos no se diseñará con tal objetivo.

Fiabilidad: La aplicación debería tener máxima fiabilidad ya que se va a ejecutar en un entorno productivo, aun así, en caso de fallo, no correrán peligro vidas humanas o animales, con lo que la fiabilidad no es crítica.

Mantenabilidad: La aplicación se dejará estructurada para poder acometer en el futuro ampliaciones en la funcionalidad, la aplicación no necesitará mantenimiento, aunque si requerirá que el servidor esté operativo y un mínimo mantenimiento de la BDD en caso de un número elevado de clientes.

Seguridad: al ser una aplicación en la que se mandarán mensajes que podrían leer terceras personas, es necesario idear un sistema que evite la suplantación de identidad, y ataques por denegación de servicio.

Para ello, debemos asegurarnos que los mensajes que se envíen entre cliente y servidor, así como los mensajes que se envíen entre el servidor y el encaminador destino, no puedan ser suplantados por terceras personas, de ahí la necesidad de utilizar passwords y un mecanismo de cifrado.

Para poder cifrar mediante SSL, es necesario obtener un certificado de una autoridad o firmarse un certificado uno mismo.

Muchos navegadores darán un aviso de seguridad si encuentran un certificado firmado por uno mismo. Para el desarrollo de la aplicación se utilizará este tipo de certificado a sabiendas de que no sería válido para una implantación real.

La comunicación entre el cliente y el encaminador destino no requiere seguridad, si se desean enviar datos correrá por cuenta del usuario que vayan cifrados.

3.6. Otros Requisitos

No hay ningún requisito que encaje en ésta sección.

4. Apéndices

No hay apéndices.

Manual de instalación del TunnelBroker

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

I. Introducción.

En la siguiente sección se detallan algunos términos y conceptos que ayudarán al entendimiento de la tecnología que se desarrollará, así como de las explicaciones y diagramas que en el manual se ilustran.

En la siguiente ilustración se muestra la infraestructura "real" que se está emulando. Su comportamiento en máquinas físicas como virtuales es idéntico. En algunos aspectos se va a suponer que se parte de una red IPv6 ya configurada.

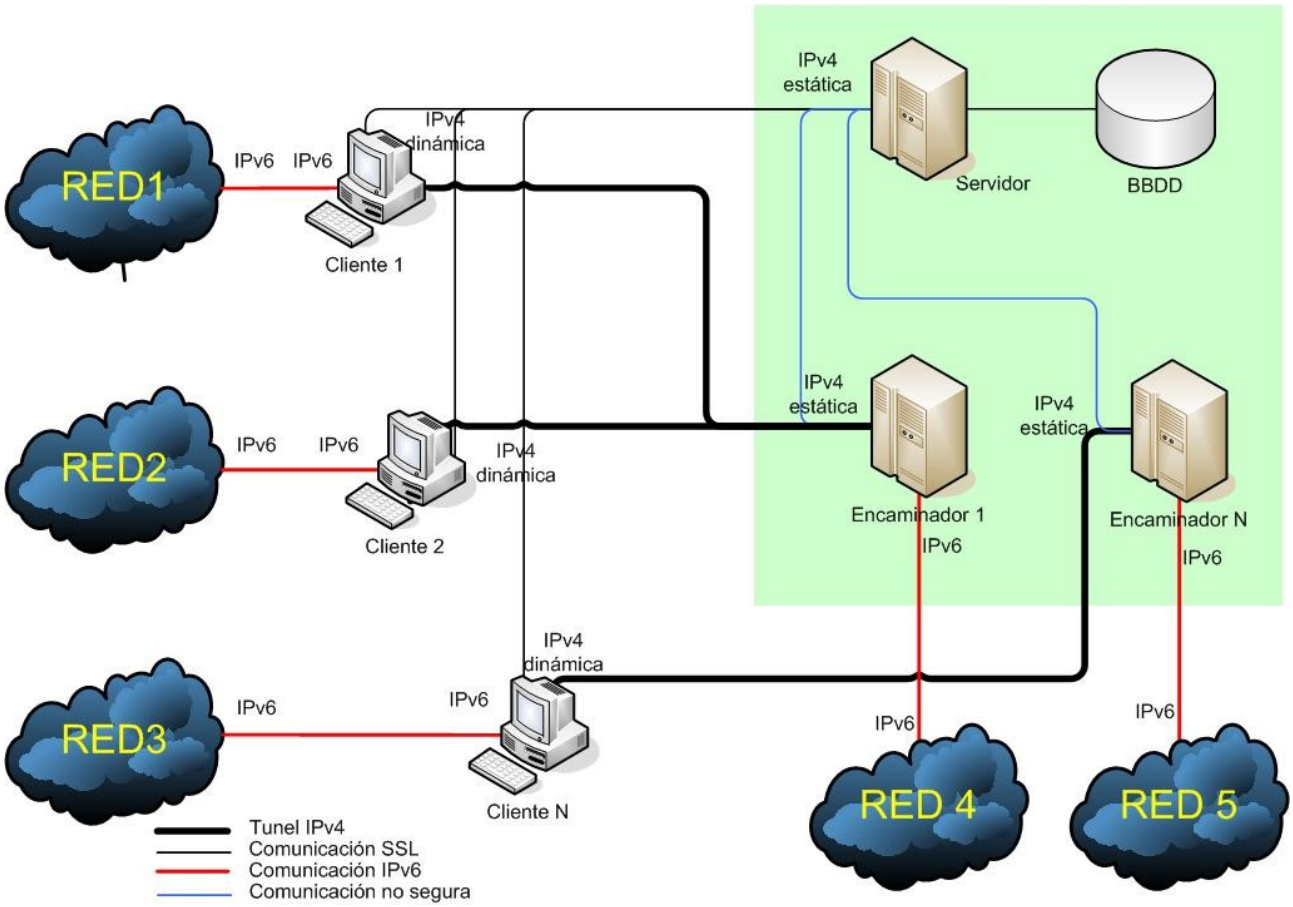


Figura 1.1.Gráfico explicativo de la infraestructura a emular.

A. Términos y abreviaturas.

Túnel: efecto de la utilización de ciertos protocolos de red que encapsulan a otro protocolo.

Así, el protocolo A es encapsulado dentro del protocolo B, de forma que el primero considera al segundo como si estuviera en el nivel de enlace de datos.

Tunelizar: técnica que se suele utilizar para trasportar un protocolo determinado a través de una red que, en condiciones normales, no lo aceptaría.

II. Prerrequisitos.

Antes de ejecutar los scripts, asegurarse que se tienen los permisos adecuados de ejecución.

A. Jerarquía de ficheros.

Para el montaje del proyecto, se ha pensado en una estructura de ficheros similar a la siguiente.

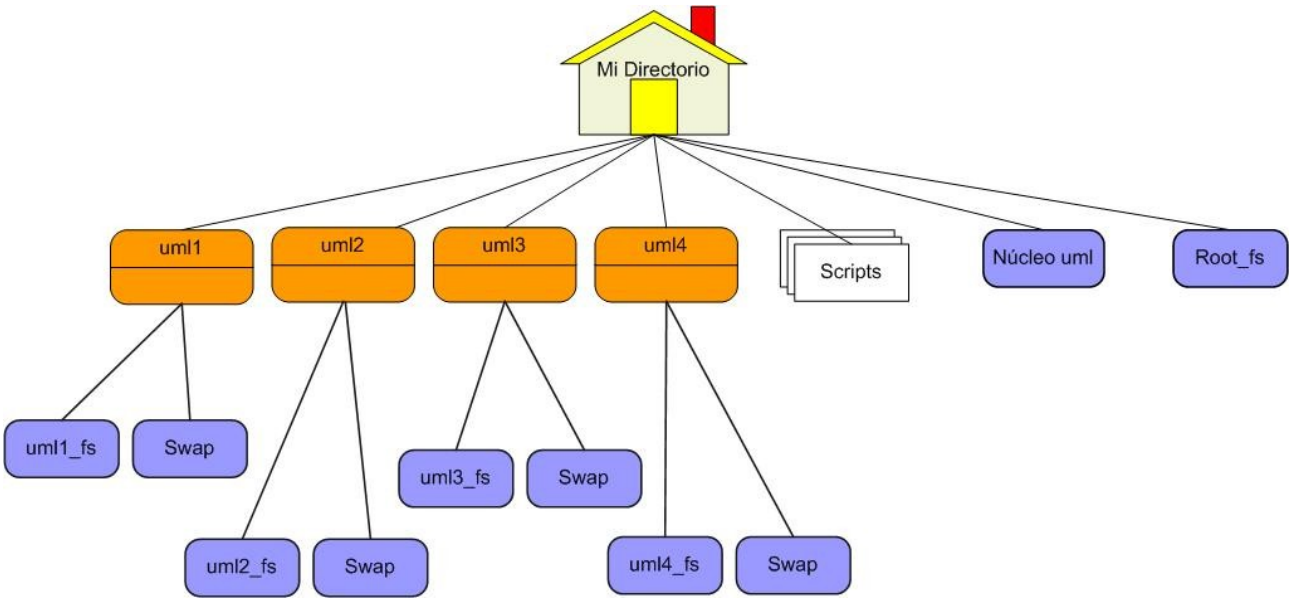


Figura 2.1. Diagrama con la estructura de ficheros utilizada en el presente manual.

III. Modo de uso.

A. Arrancar el proyecto.

Para arrancar el entorno del sistema, se deberá ejecutar el script *PROYECTO* del siguiente modo:

./proyecto <interfaz_internet> <usuario>

donde se le indica el usuario que ejecutará el sistema y qué interfaz de red utilizará para salir a internet.

B. Parar la ejecución del proyecto.

Para parar el entorno del sistema, se deberá ejecutar el script *PARA* del siguiente modo:

./para

Este script manda una señal de parada a todas las máquinas virtuales de tal modo que inician el proceso de shutdown. Es importante que las máquinas se hayan iniciado correctamente, ya que en caso contrario, este comando no funcionaría correctamente y podría provocar el mal funcionamiento del sistema.

C. Borrar los ficheros COW.

Para borrar los ficheros Cow contenidos en las carpetas uml1, uml2, uml3 y uml4 se deberá ejecutar el script *ELIMINACOWS* del siguiente modo:

./eliminaCows

Este script elimina los ficheros COW del sistema, por lo que todos los cambios que se hayan hecho en particular sobre las máquinas virtuales se perderán y solamente permanecerán aquellas características configuradas en el fichero root_fs.

D. Configuración de las máquinas virtuales.

Para configurar la red de las máquinas virtuales se deberá hacer lo siguiente (UNA ÚNICA VEZ):

a) Montar en la máquina virtual el sistema de ficheros del host. Esto se hace con la instrucción “mount -t hostfs none <directorio>”.

b) Navegar hasta el directorio donde se encuentren los scripts 'configuracionVirtualHost' y 'configuracionVirtualRouter'. Cada uno de estos scripts sirve para configurar una clase de host dentro de la jerarquía, teniendo dos tipos: el que comunica la intranet IPv6 con el exterior (llamemosle Router) y el equipo interno de la red IPv6 (llamemosle host). A continuación se detalla el modo de uso:

i) Si nos encontramos configurando un host, deberemos:

- copiar el script 'configuracionVirtualHostN' donde N es el nombre del host al directorio '/etc/init.d/'.

- Otorgarle permisos con 'chmod 755 configuracionVirtualHostN'
- Añadir la referencia relativa al script desde cada uno de los modos de ejecución con la ejecutando 'update-rc.d configuracionVirtualHostN defaults'. Si nos equivocamos en este paso se puede volver atrás quitando la referencia con 'update-rc.d -f configuracionVirtualHostN remove'
- Con esto el script se ejecutará cada vez que se arranque la máquina virtual y se configurará automáticamente.

ii)De forma análoga para el script 'configuracionVirtualRouterN' de los Router y, además, en este caso se realizará el mismo proceso con el fichero 'configuracionTunelX' donde la X es 1 o 3, los router de nuestra red.

- copiar el script 'configuracionTunelN' donde N es el nombre del router para el que se configura el extremo del túnel.
- Otorgarle permisos con 'chmod 755 configuracionTunelN'
- Añadir la referencia relativa al script desde cada uno de los modos de ejecución con la ejecutando 'update-rc.d configuracionTunelN defaults'. Si nos equivocamos en este paso se puede volver atrás quitando la referencia con 'update-rc.d -f configuracionTunelN remove'
- Con esto el script se ejecutará cada vez que se arranque la máquina virtual y se configurará automáticamente el túnel.

c) Desmontar el sistema de ficheros del host con la instrucción “umount <directorio>”.

d) La máquina queda así configurada para conectarse al host y a internet.

E. Añadir un usuario a la máquina virtual.

Esta sección indica cómo añadir nuevos usuarios a las máquinas virtuales.

Para crear un usuario nuevo en la máquina virtual se deberá hacer lo siguiente:

a) Primero se añade una cuenta nueva:

useradd -d /home/<nombre_del_usuario> -g <grupo_del_usuario> -m nombre_del_usuario

(-d: directorio home del usuario; -g: grupo al que pertenecerá el usuario; -m: crear el directorio home si éste no existe)

b) A continuación se especifica una contraseña para la cuenta creada:

#passwd nombre_del_usuario

c) Seguidamente se añade el usuario a un grupo existente (omitir si se especifica en useradd):

#gpasswd -a nombre_del_usuario grupo_del_usuario

F. Baja de usuarios o grupos.

- Usuarios.

  #userdel nombre_de_usuario

  (si se desea eliminar también todos los archivos y subdirectorios contenidos dentro del directorio de trabajo del usuario a eliminar, se dee agregar la opción -r).

- Grupos.

  #groupdel nombre_de_grupo

IV. Acceso a una máquina virtual mediante SSH.

Para acceder a una máquina virtual desde el host se deberán seguir los siguientes pasos.

a) Abrir una consola y usar la siguiente instrucción: “ssh usuario@maquina”

Donde máquina será el nombre (si esta configurado en el DNS) o la IP de la máquina.

V. Acceso al host desde una máquina virtual mediante FTP.

Para acceder al host desde una máquina virtual via FTP se deberán seguir los siguientes pasos.

a) Instalar el servidor de ftp: “apt-get install ftp”.

b) Ejecutar la instrucción “ftp”.

c) Ejecutar la instrucción “open” e introducir la dirección del servidor e iniciar la sesión.

d) Utilizar las directivas put, get mput, mget... para trabajar con los ficheros.

e) Utilizar la directiva “bye” para salir y cerrar la sesión.

VI. Acceso al host montando su sistema de ficheros.

Esta ha sido la mejor opción y la que más se ha utilizado durante el desarrollo. Consiste en montar el sistema de ficheros del host (la máquina física) en un directorio de la máquina virtual utilizando el sistema de ficheros como si de una sola máquina se tratase.

Para ello se debe ejecutar el siguiente comando:

“mount -t hostfs none <directorio>”.

VII. Actualización del paquete user-mode-linux.

a) Mirar las dependencias del paquete user-mode-linux con “apt-cache show user-mode-linux”.

b) Normalmente aparece 'uml-utilities' y no frecuentemente 'libc6'.

c) Comprobar las versiones de las dependencias con “dpkg -l nombre_paquete”.

d) Instalar nueva versión del paquete user-mode-linux con “dpkg -i nombre_paquete”

e) Cambiar la instrucción de llamada en el script 'lanza'. Donde ahora pone 'linux2.6.24-rc7' se quedará como 'linux'.

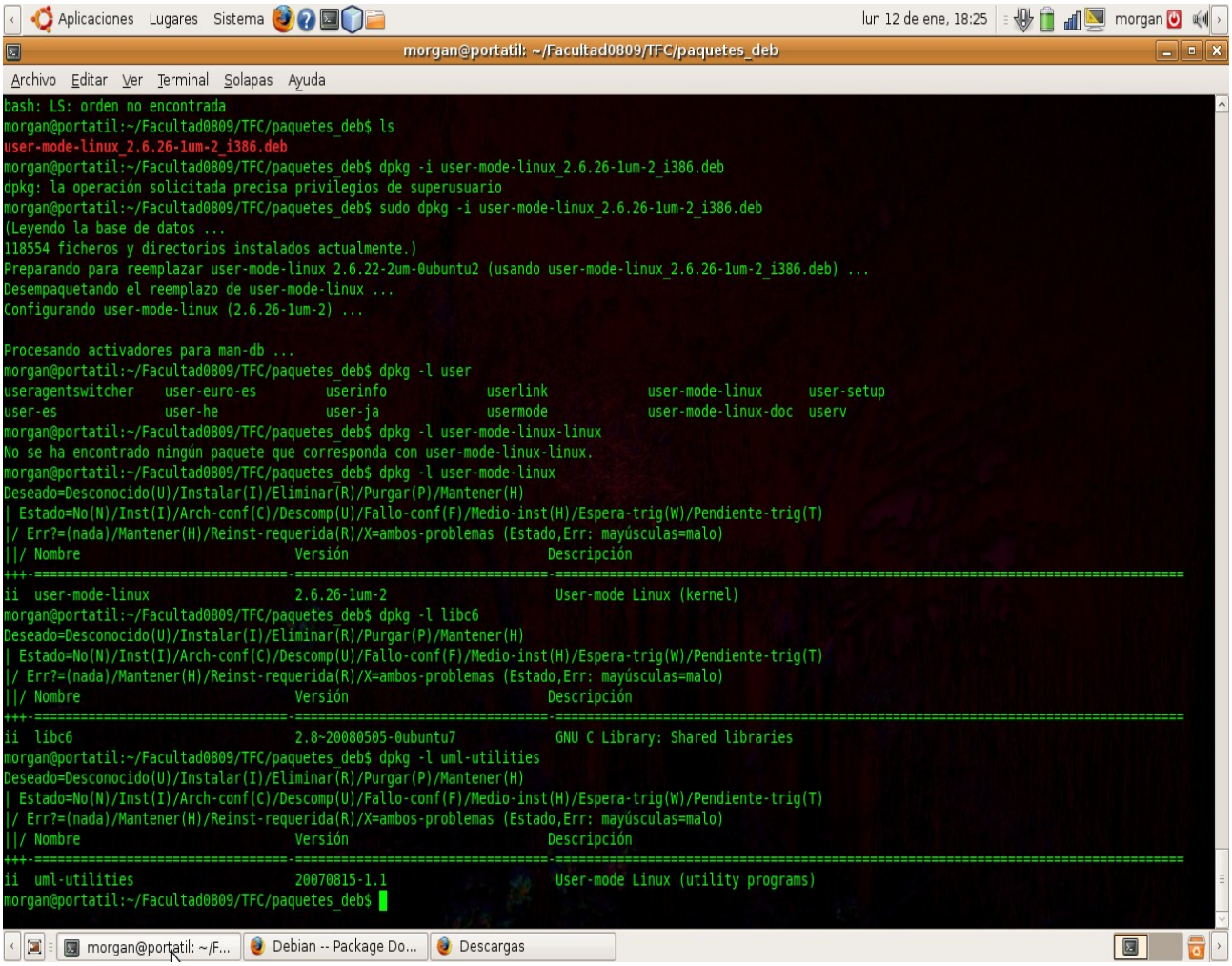


Figura 7.1. Aspecto del terminal al finalizar el proceso.

Al hacer un dpkg -l user-mode-linux se puede comprobar que la versión ha sido instalada satisfactoriamente.

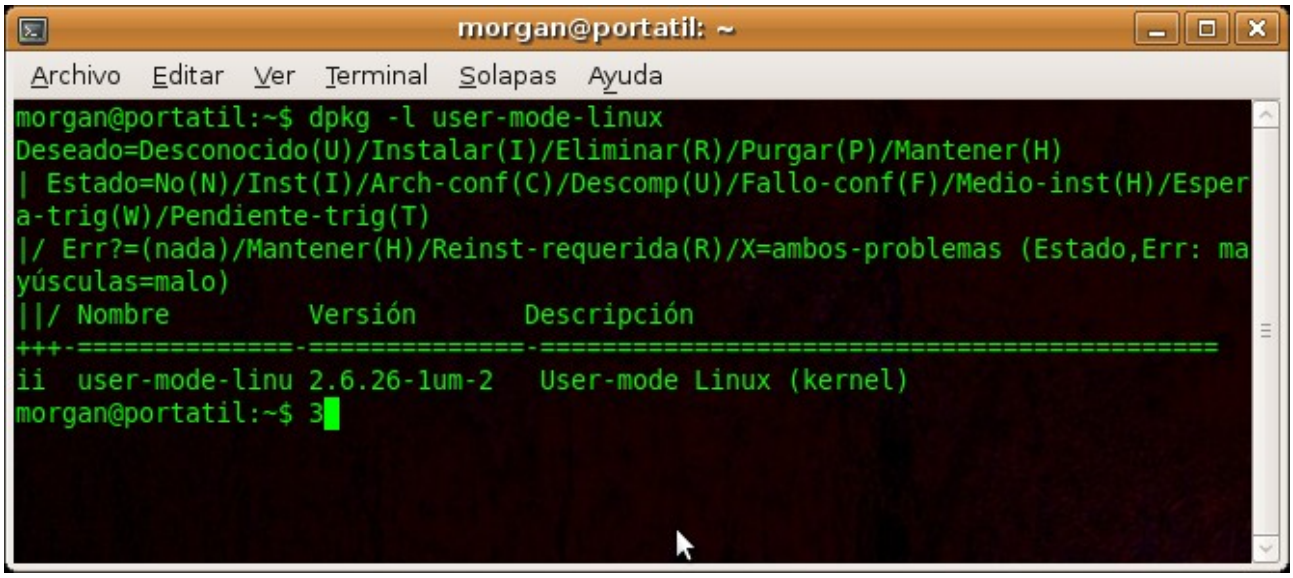


Figura 7.2. Aspecto del terminal al comprobar la instalación.

El siguiente paso es copiar los modules nuevos del nuevo núcleo al root_fs.

Para ello se debe montar el sistema de ficheros del root_fs en el host con el siguiente comando:

'sudo mount -o loop root_fs /mnt'

Seguidamente se copian los modules al sistema de ficheros montado con el siguiente comando:

'cp -r /usr/lib/uml/modules/2.6.26 /mnt/lib/modules'

A la hora de desmontar el sistema de ficheros se debe ejecutar la siguiente comando:

'umount /mnt'

A veces, a la hora de ejecutar esta acción, nos informa de que no se puede desmontar ya que se encuentra ocupado. Para ello ejecutamos 'lsof /mnt' y vemos el proceso que lo tiene ocupado. Una vez identificado lo matamos con 'kill numero_proceso'.

A continuación se debe ejecutar un '*ELIMINACOWS*' para que la próxima vez que se arranquen las máquinas virtuales se creen con las nuevas versiones de los modules.

Finalmente se ejecuta el script '*PROYECTO*' para arrancar las máquinas virtuales.

VIII. Instalación y configuración de la infraestructura en las máquinas virtuales

Primero refrescamos la lista de paquetes que vamos a instalar en la máquina donde se harán todas las pruebas.

Para ello ejecutamos:

'apt-get update'

Luego instalamos:

'apt-get install user-mode-linux uml-utilities debootstrap bridg-utils'

Los paquetes umlrun y umlrun-uml no se encuentran en el apt y se deben descargar de internet.

En este caso son paquetes .deb que se instalan con la sentencia:

"sudo dpkg -i fichero.deb [fichero.deb,...]'

Partimos de que tenemos los ficheros root_fs y los correspondientes umlN_fs según la jerarquía comentada en Jerarquía de Ficheros.

IMPORTANTE: la versión del paquete user-mode-linux que se instala del apt, no es la última. Es la versión de kernel 2.6.22 y no soporta ficheros cow. Es necesario instalar un paquete superior o igual al 2.6.23. En nuestro caso usamos el 2.6.26.

Una vez instalado debemos crear N carpetas (una por cada máquina virtual que queramos) y hacer una copia del fichero swap en ellas.

Para arrancar el proyecto ejecutamos './proyecto <interfaz_red> <usuario>

IX. Instalación y configuración de la base de datos.

A. Instalación del SGBD Mysql.

Instalación de Mysql Server

La aplicación Mysql-Server es un sistema gestor de base de datos que nos permitirá guardar la información de forma relacional.

Se instala mediante el comando:

'sudo apt-get install mysql-server'

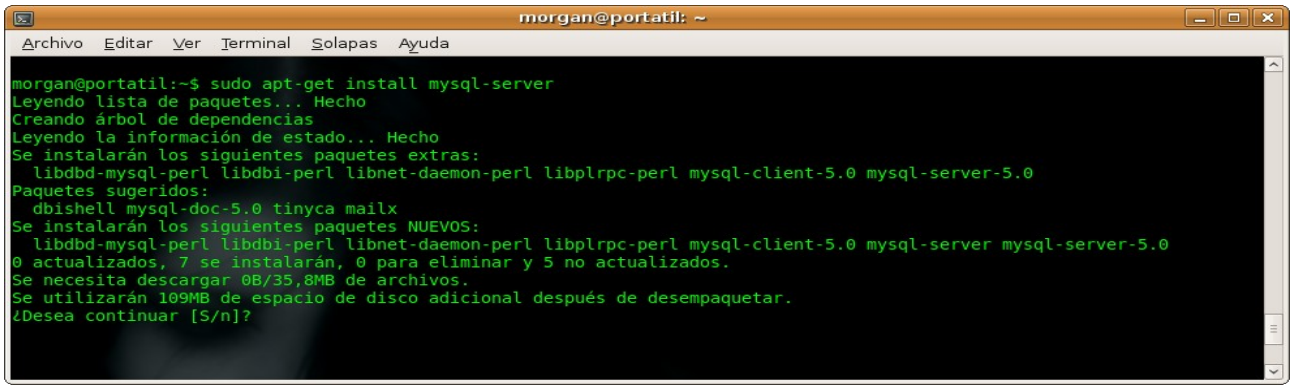


Figura 9.1

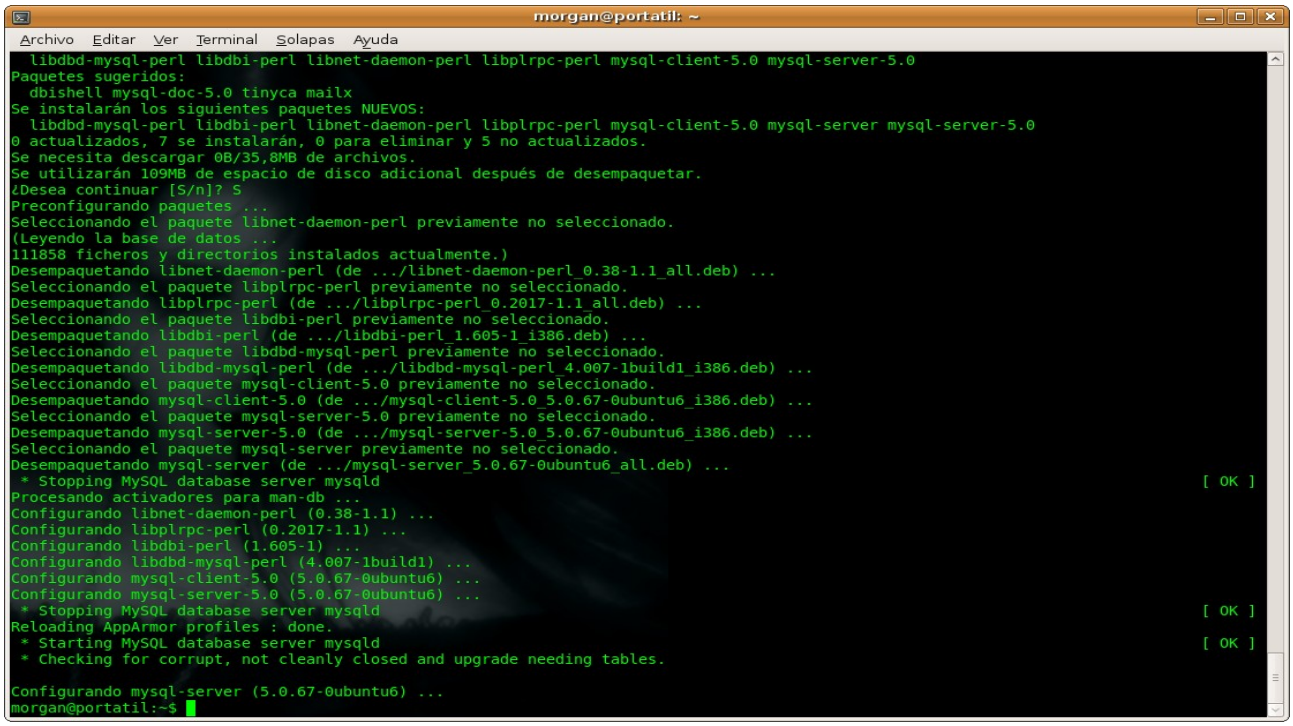


Figura 9.2

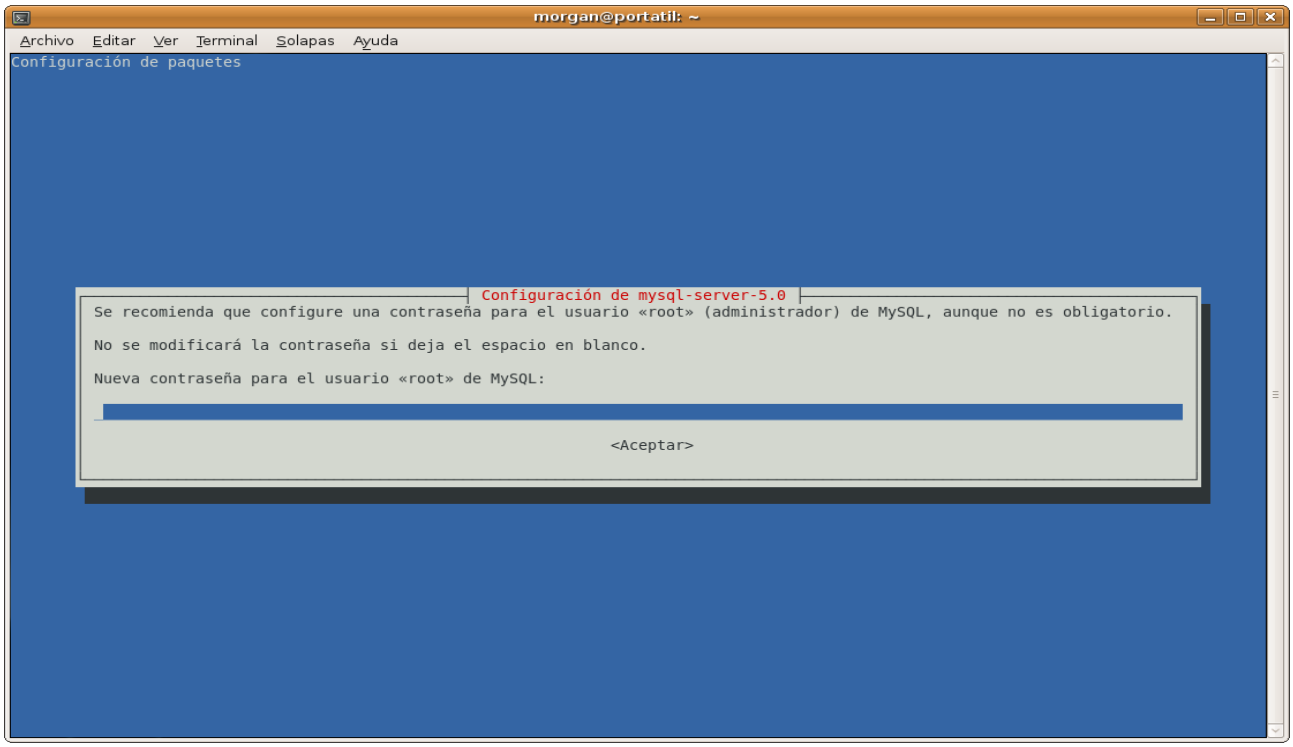


Figura 9.3

Instalación de MysqlAdministrator

MysqlAdministrator es una aplicación que nos permitirá configurar el servidor Mysql de manera gráfica e intuitiva.

Se instala mediante el comando:

'sudo apt-get install mysql-admin'.

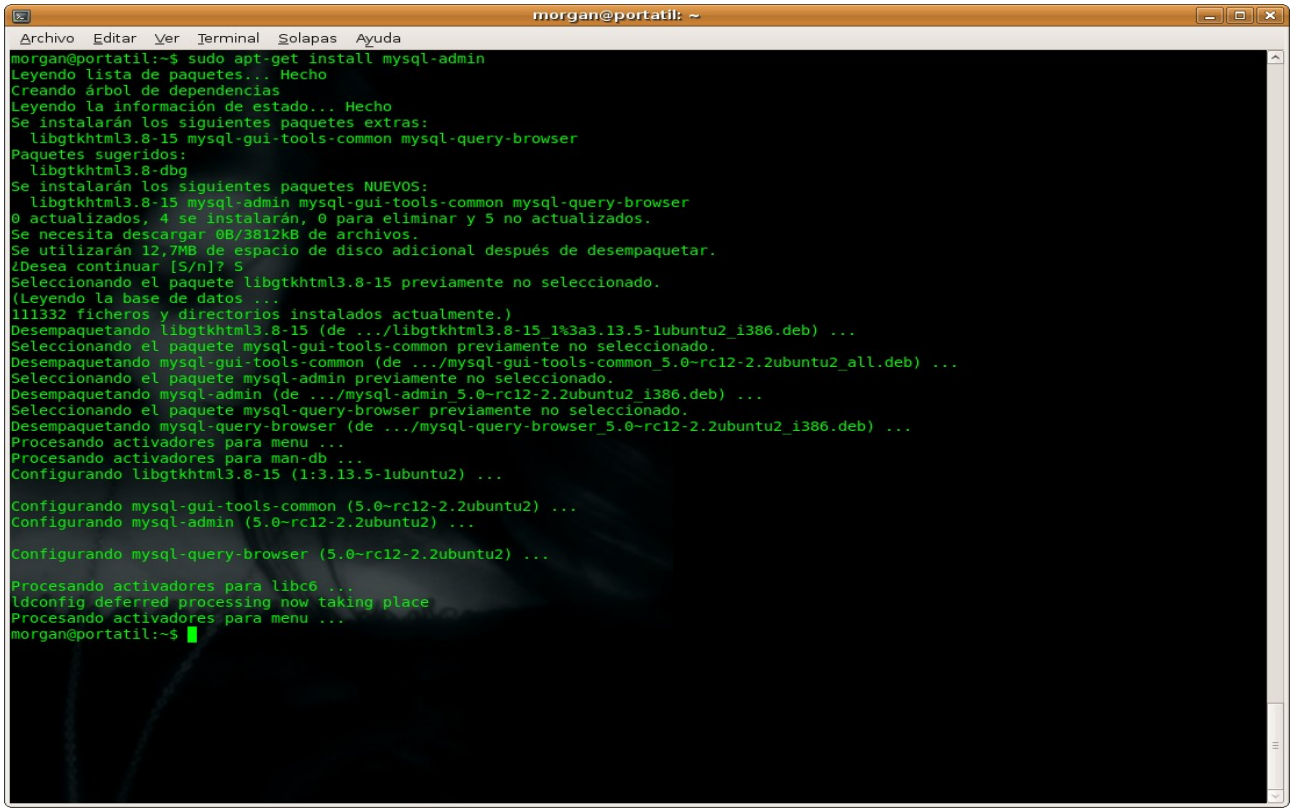


Figura 9.4

B. Configuración del servidor Mysql.

Una vez instalado el servidor es necesario configurar los privilegios de los usuarios así como la estructura relacional de nuestra aplicación. Para ello nos vamos a ayudar de la herramienta gráfica MysqlAdministrator.

Las acciones necesarias para configurar la BDD son:

- Crear el esquema donde se va a guardar la estructura relacional.
- Crear un usuario con los permisos adecuados para manejar dicho esquema.
- Restaurar la información contenida en el script. En él se encuentra toda la información de la estructura de la BDD de la aplicación.

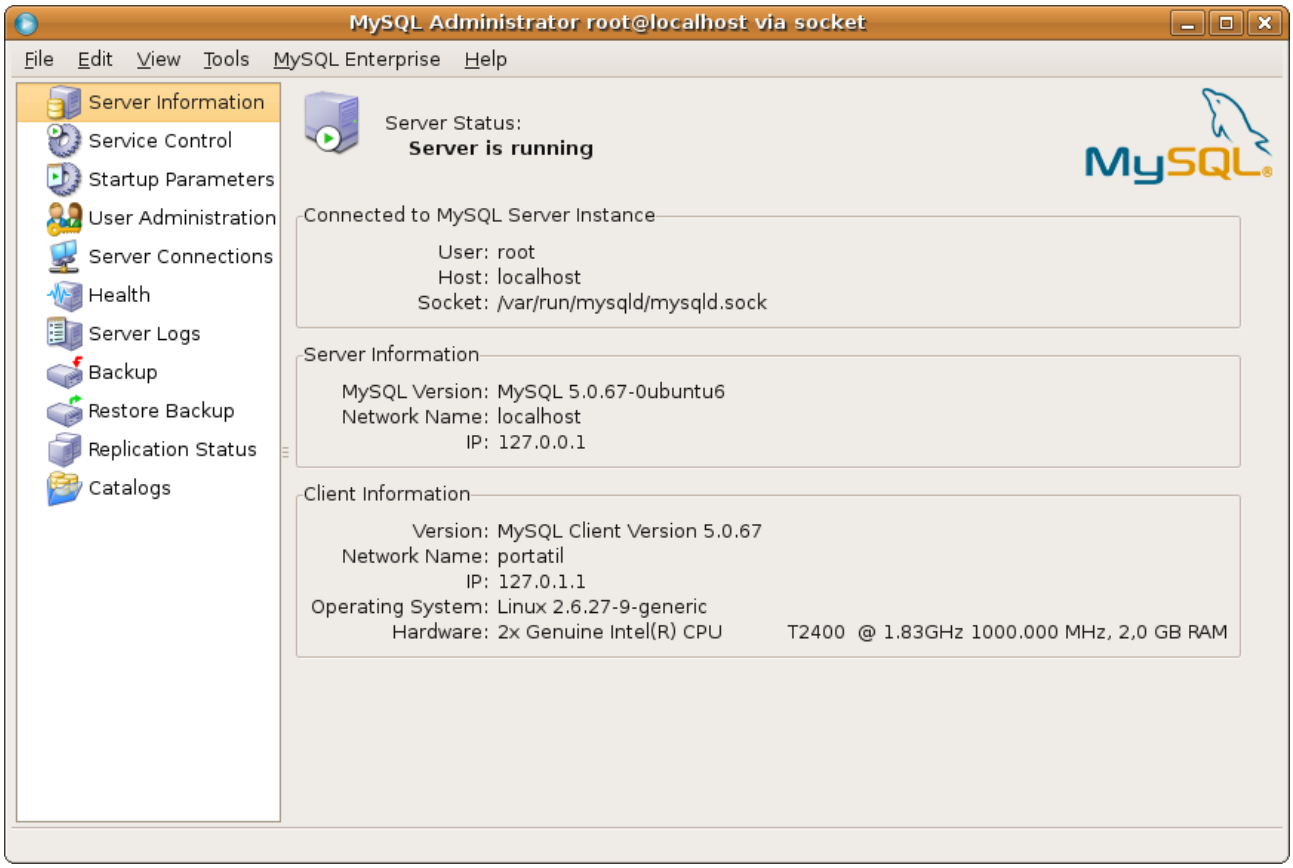


Figura 9.5. Apariencia de la aplicación MysqlAdministrator.

A continuación se detallan todos los pasos necesarios para configurar correctamente el servidor Mysql.

1. Crear un esquema llamado ***tunnel_broker***.

Si por algún motivo no se puede o no se desea que el esquema se llame de esa manera, existe la posibilidad de crear el esquema con el nombre que nos interese simplemente cambiando la variable $db del fichero de configuración por el valor deseado.

Los pasos para crear un esquema son:

a) Abrir el programa de administración de la BDD ***MysqlAdministrator*** e inicializar la sesión con permisos de root.

b) Seleccionar la opción ***Catalogs*** del menú de la izquierda.

c) Presionar el botón derecho en el recuadro ***Squemata*** y seleccionar ***Create Squema***.

d) Llamarlo tunnel_broker.

2. Restaurar la BDD a partir del script *tunnel_broker_BBDD_v1.x.sql.*

En dicho script está guardada toda la estructura e información de la base de datos.

Los pasos para restaurar un esquema son:

a) Abrir el programa de administración de la BDD ***MysqlAdministrator*** e inicializar la sesión con permisos de root.

b) Seleccionar la opción ***Restore Backup*** del menú de la derecha.

c) Presionar el botón ***Change Path*** de la esquina inferior izquierda y seleccionar el directorio en el que se encuentra el script.

d) Seleccionar el fichero en el recuadro de la izquierda.

e) Aceptar la codificación de caracteres UTF-8.

f) Presionar el botón ***Restore Backup***.

3. Crear un usuario ***tunnel_broker*** con derechos para SELECT, INSERT, UPDATE, DELETE y LOCK TABLES para el esquema creado anteriormente.

Existe la posibilidad de utilizar un usuario distinto al propuesto en el manual, para ello simplemente deberá cambiar la variable $user del fichero de configuración.
Los pasos para crear un usuario de la BDD son:

a) Abrir el programa de administración de la BDD ***MysqlAdministrator*** e inicializar la sesión con permisos de root.

b) Seleccionar la opción ***User Administration*** del menú de la derecha.

c) Presionar el botón derecho del ratón en el recuadro ***User Accounts*** y seleccionar ***New User***.

d) En la pestaña ***User Information***, introducir el nombre del usuario (***tunnel_broker***), la contraseña y aplicar los cambios. Esta contraseña también deberá introducirse en la variable $password del fichero de configuración.

e) Como a este esquema se va a acceder desde la máquina local por la interfaz web y por el programa del servidor debemos dar permisos al usuario para admitir conexiones entrantes desde la propia máquina.

f) Para ello, presionar el botón derecho sobre el nombre del usuario recién creado y seleccionar la opción ***Add Host*** y seguidamente, en la ventana que aparezca,

seleccionar la opción ***Localhost*** como se aprecia en la Figura 1.6.

g) Seleccionar la conexión @localhost del usuario y, en la pestaña ***Schema Privileges***, asignar al esquema los permisos SELECT, INSERT, UPDATE, DELETE y LOCK_TABLES como se aprecia en la figura 1.7.

h) Aplicar los cambios.

i) Opcionalmente, si se desean conexiones únicamente desde la propia máquina, se puede borrar la conexión @% para que nadie pueda conectarse desde el exterior.

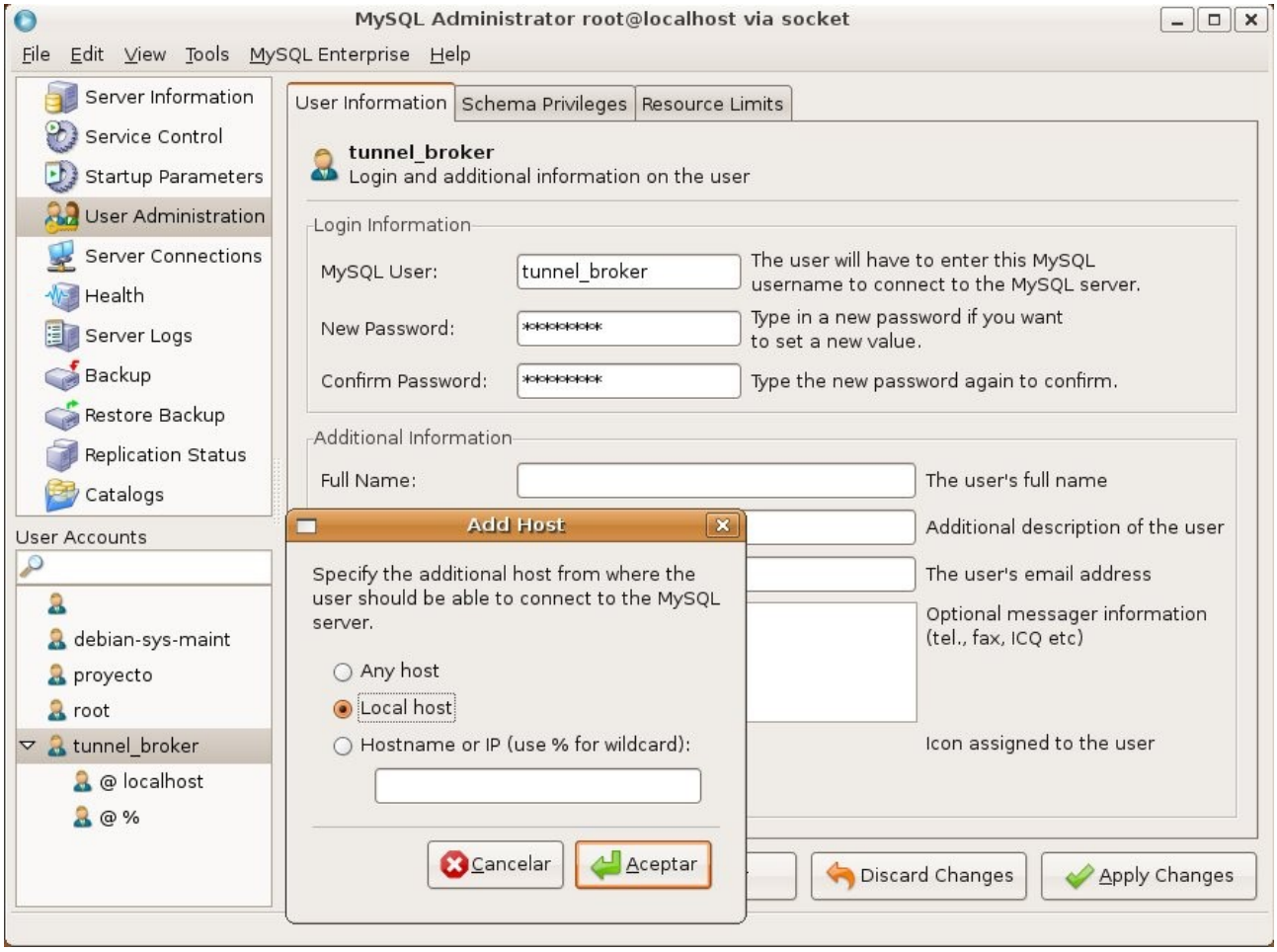


Figura 9.6. Se ha añadido un nuevo host a través del cual el usuario tunnel_broker se podrá conectar al servidor.

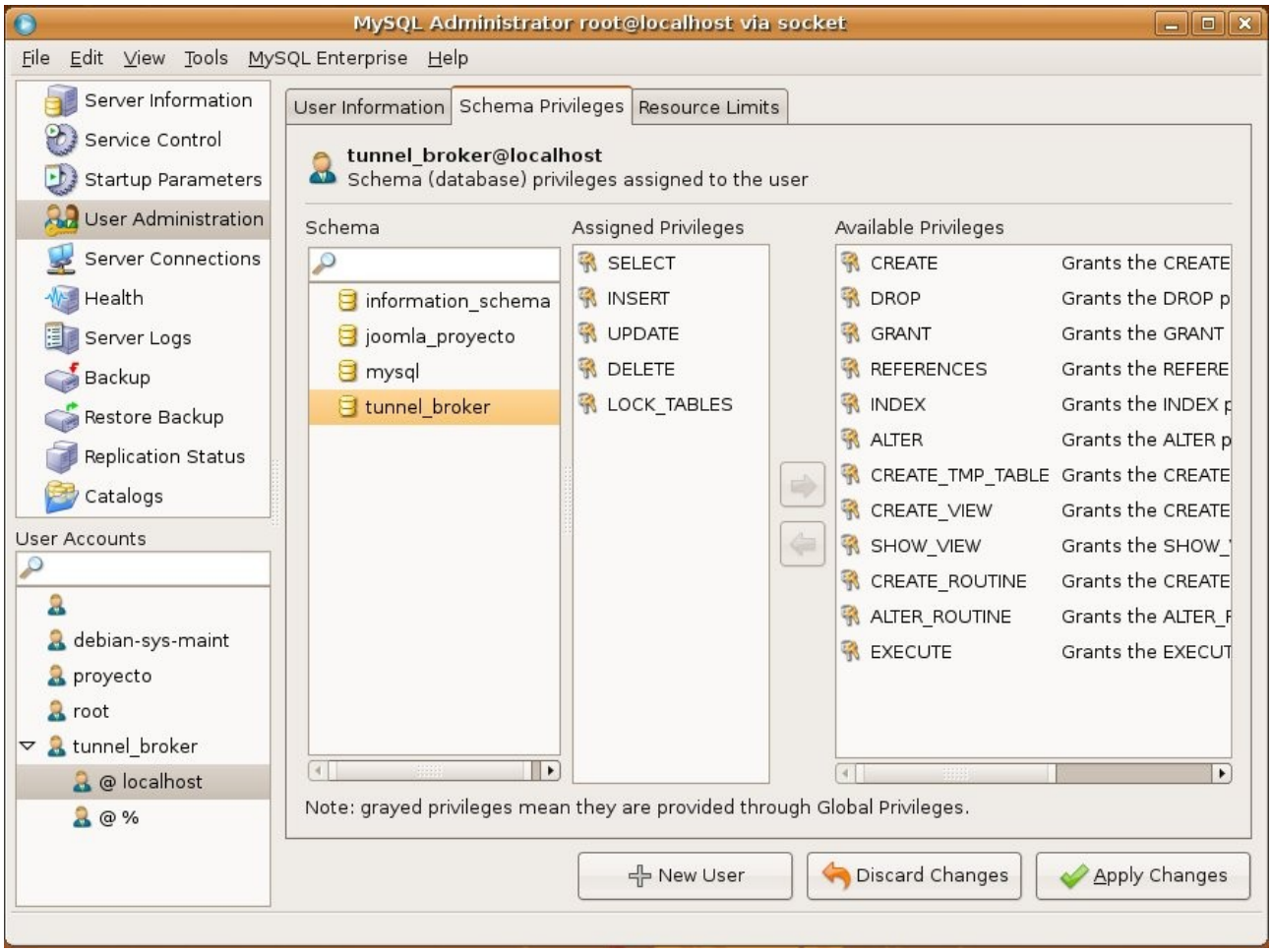


Figura 9.7. Privilegios del usuario tunnel_broker para el esquema tunnel_broker y para las conexiones provenientes del localhost.

X. Instalación y configuración de los archivos de la interfaz web.

Se asume que el servidor apache2 está instalado y configurado con soporte para php5. Ha día de hoy, la instalación por defecto de php5, incluye las librerías adecuadas para poder conectarse a una BDD Mysql.

1. Descomprimir el archivo .zip en la carpeta raíz del servidor web.

El servidor web apache2 dispone de un directorio raíz en el que se deben colgar todas las páginas web que se deseen. El directorio por defecto es /var/www/, sin embargo dicho directorio se puede sustituir en los archivos de configuración de dicha aplicación. Se

recomienda instalar la aplicación web en el directorio /var/www/tunnelbroker para así permitir servidores virtuales.

Los pasos detallados son:

a) Descomprimir el archivo tunnel_broker_v1.x.zip en el directorio raíz del servidor web o del servidor virtual.

b) Editar las siguientes variables del archivo de configuración llamado configuration.php si no se han utilizado los valores propuestos por el manual.

$log_path: Ruta hasta el directorio /log de la aplicación. Por defecto /var/www/log

$tmp_path: Ruta hasta el directorio /tmp de la aplicación. Por defecto /var/www/tmp

$db: El nombre del esquema donde se ha restaurado la BDD.

$user: El nombre de usuario con permisos para acceder al esquema.

$password: Password del usuario $user para poder conectarse a Mysql.

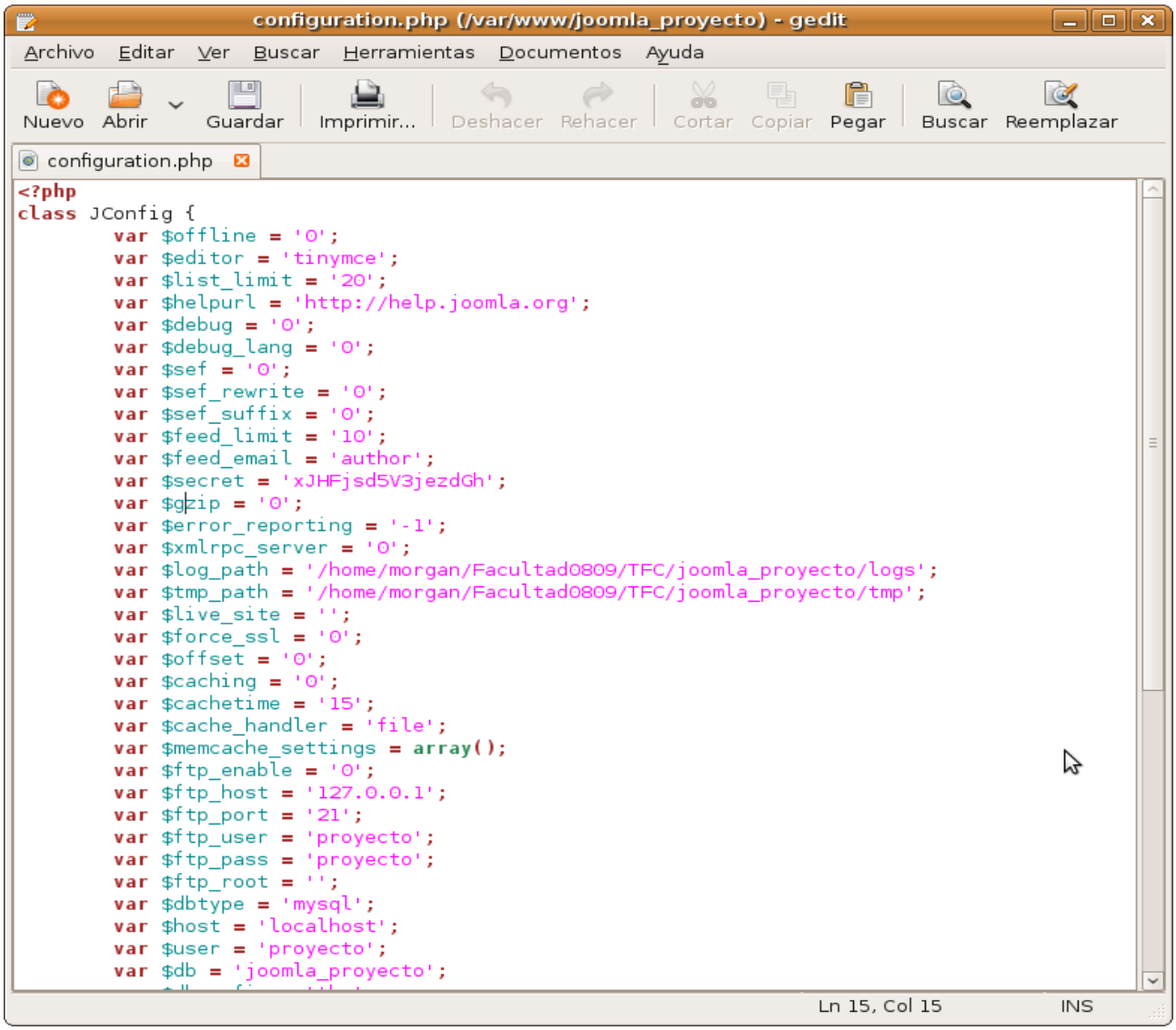


Figura 10.1. Detalle del archivo de configuración con los nuevos valores.

2. Cambiar el dueño del directorio descomprimido para que el dueño sea el servidor apache2.

   Suponiendo que se ha instalado en el directorio /var/www/ el comando sería:

   chown -R www-data /var/www/

3. Cambiar los permisos para que www-data pueda leer y escribir y el resto sólo leer en los ficheros y en los directorios que www-data también pueda

ejecutar.

Suponiendo que se ha instalado en el directorio /var/www/tunnel_broker el comando sería:

- Para los directorios: find /var/www/tunnel_broker -type d -exec chmod 744 \{\} \;
- Para los ficheros: find /var/www/tunnel_broker -type f -exec chmod 644 \{\} \;

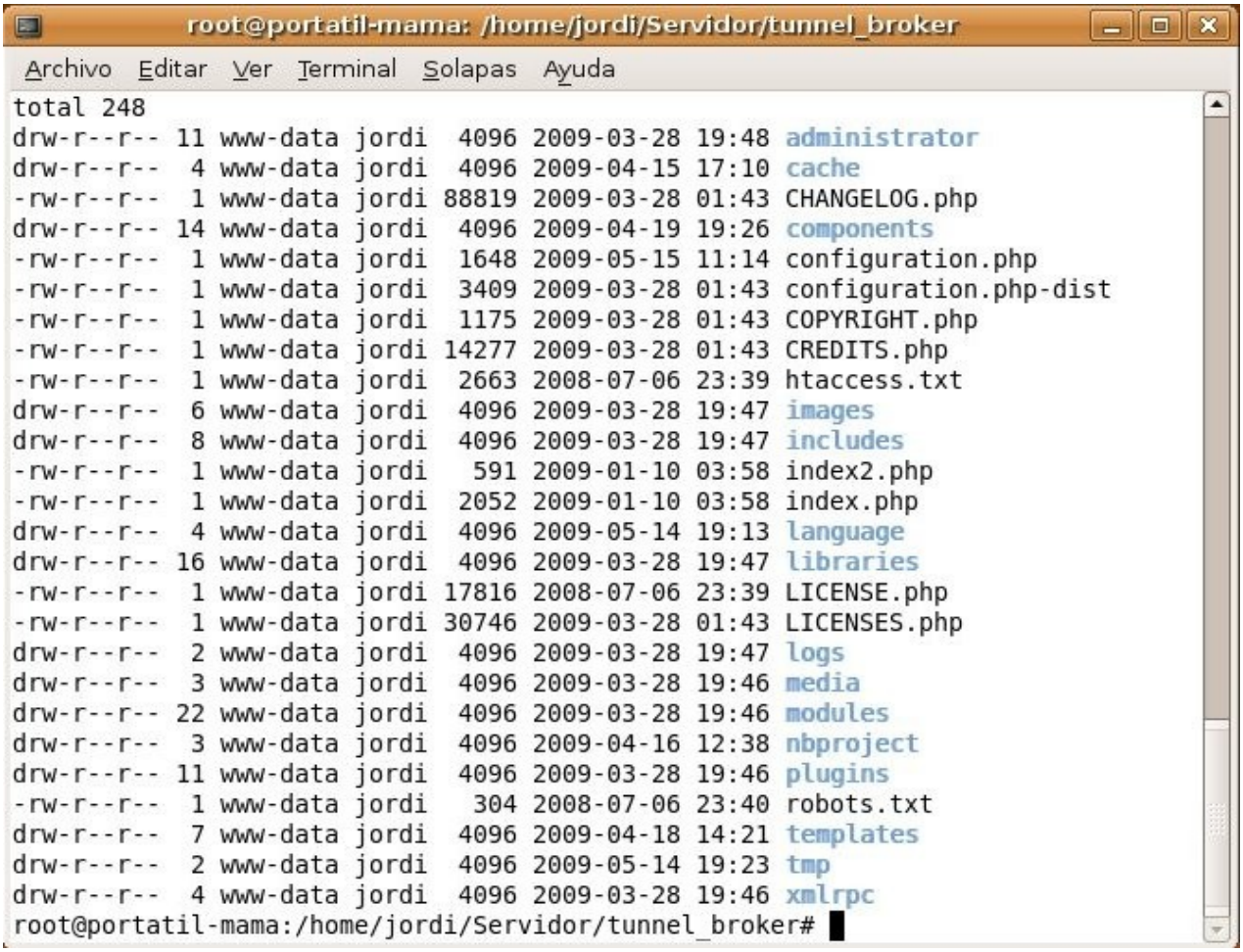


Figura 10.2. Detalle de los permisos del directorio donde se ha instalado la página web.

XI. Comprobar la instalación web.

Abra su navegador e introduzca http://localhost/tunnel_broker/. Si se ha instalado correctamente debería ver la página web del tunnel broker.

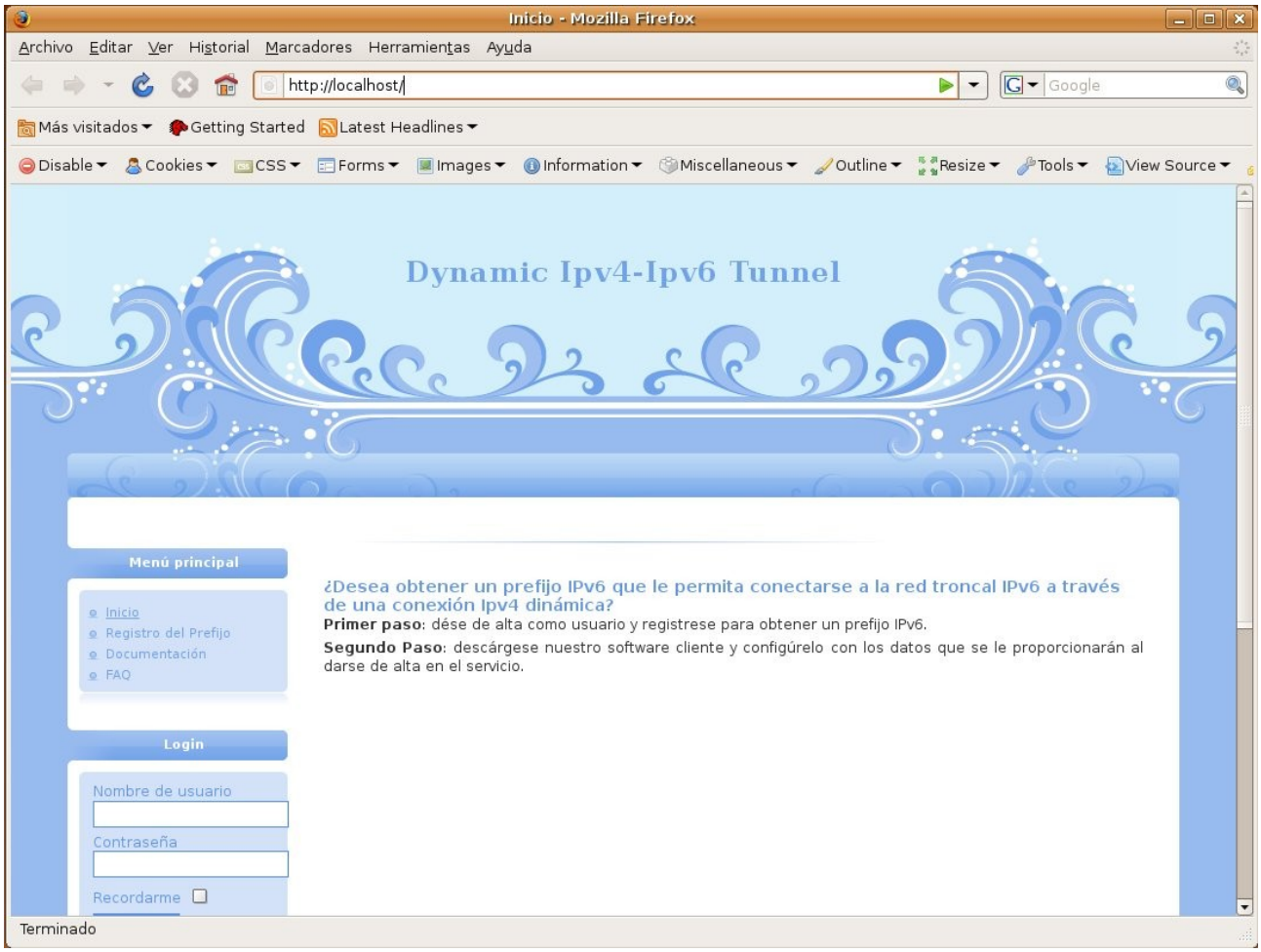


Figura 11.1. Página de bienvenida de la aplicación web.

Manual de uso del TunnelBroker

Jordi Alcaide Romeu
Ignacio Santabárbara Ruiz
Rafael Jiménez García

Profesor: Juan Carlos Fabero Jiménez
Curso académico 2008-2009
PROYECTO DE SISTEMAS INFORMÁTICOS
Facultad de informática
Universidad Complutense de Madrid

Índice

I. Introducción.

La página web se implementado utilizando el framework Joomla! 1.5.x el cual nos proporciona un sistema gestor de contenidos totalmente operativo y funcional.

Joomla! se distribuye con una licencia GPL, está escrito en PHP, está en continuo desarrollo por su comunidad y es totalmente gratuito.

Como Joomla! no proporciona toda la funcionalidad que la aplicación requiere, se ha creado una componente (una extensión de Joomla! 1.5.x) que amplía la funcionalidad que proporciona el núcleo de Joomla!. De esta forma reutilizamos código ya probado en las partes comunes y sólo implementamos la parte correspondiente a la gestión de túneles, encaminadores y prefijos.

Joomla proporciona mucha más funcionalidad que la que requiere la aplicación. Entre las funcionalidades que no se van a utilizar cabe mencionar la gestión de archivos multimedia, la gestión de idiomas y traducciones así como la gestión de banners.

Aunque, como hemos dicho, se ha creado una componente para Joomla!, la aplicación se distribuye con ella integrada y personalizada, con lo que podemos empezar a hablar de la aplicación por sí sola sin tener que hacer referencias constantes al framework del que deriva.

La aplicación web tiene 2 partes bien diferenciadas; la parte a la que accederán los clientes a partir de ahora llamado sitio (front end) y la parte a la que accederán las personas encargadas de gestionar la página web a partir de ahora llamado administración (back end). Están tan diferenciadas que los clientes nunca sabrán de la existencia de la administración, ya que ni siquiera hay enlaces entre las dos partes.

Por ello, se va a explicar por separado cada una de las dos partes

II. El sitio (Front End).

A. Introducción

Desde el sitio se van a poder realizar las siguientes acciones:

- Navegar como invitado.
- Registrarse como nuevo usuario.
- Inicializar la sesión de un usuario ya dado de alta.
- Obtener un prefijo IPv6 si se es un usuario registrado.

- Modificar o borrar el prefijo IPv6 si se es un usuario registrado que obtuvo un prefijo.

Aunque el aspecto final de la página es totalmente configurable, tanto en número de elementos, como posición, como el estilo, por defecto la página de inicio debería ser la siguiente:

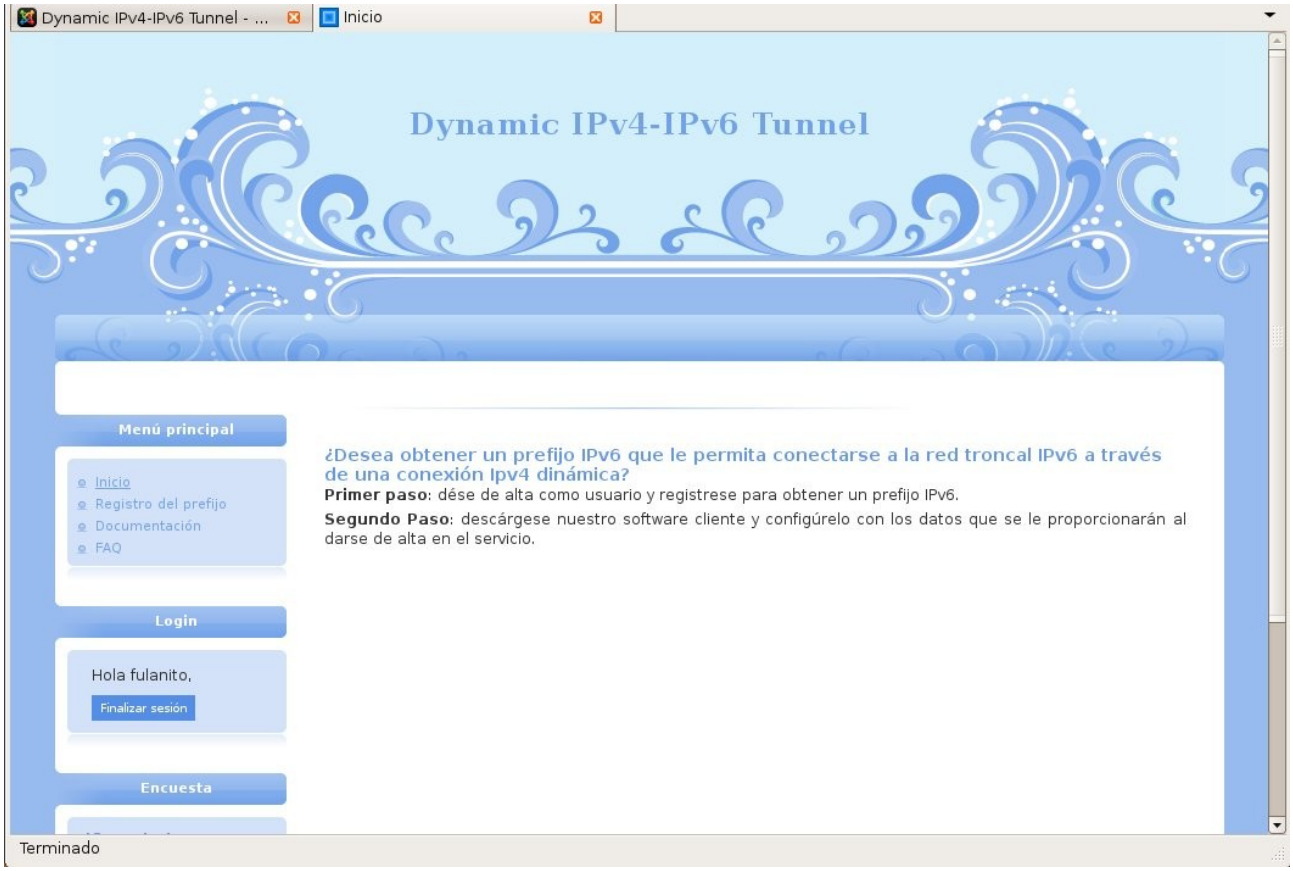


Figura 1.1. Aspecto de la página de inicio.

B. Registro de un nuevo cliente e inicio de la sesión

Si se quiere tener acceso al servicio de túneles, es necesario estar registrado. Si no se ha dado de alta previamente, deberá hacerlo pinchando en el enlace *Regístrese aquí.*

Le aparecerá una pantalla como la de la figura 1.2. Rellene los datos y pulse en el botón *Regístrese*.

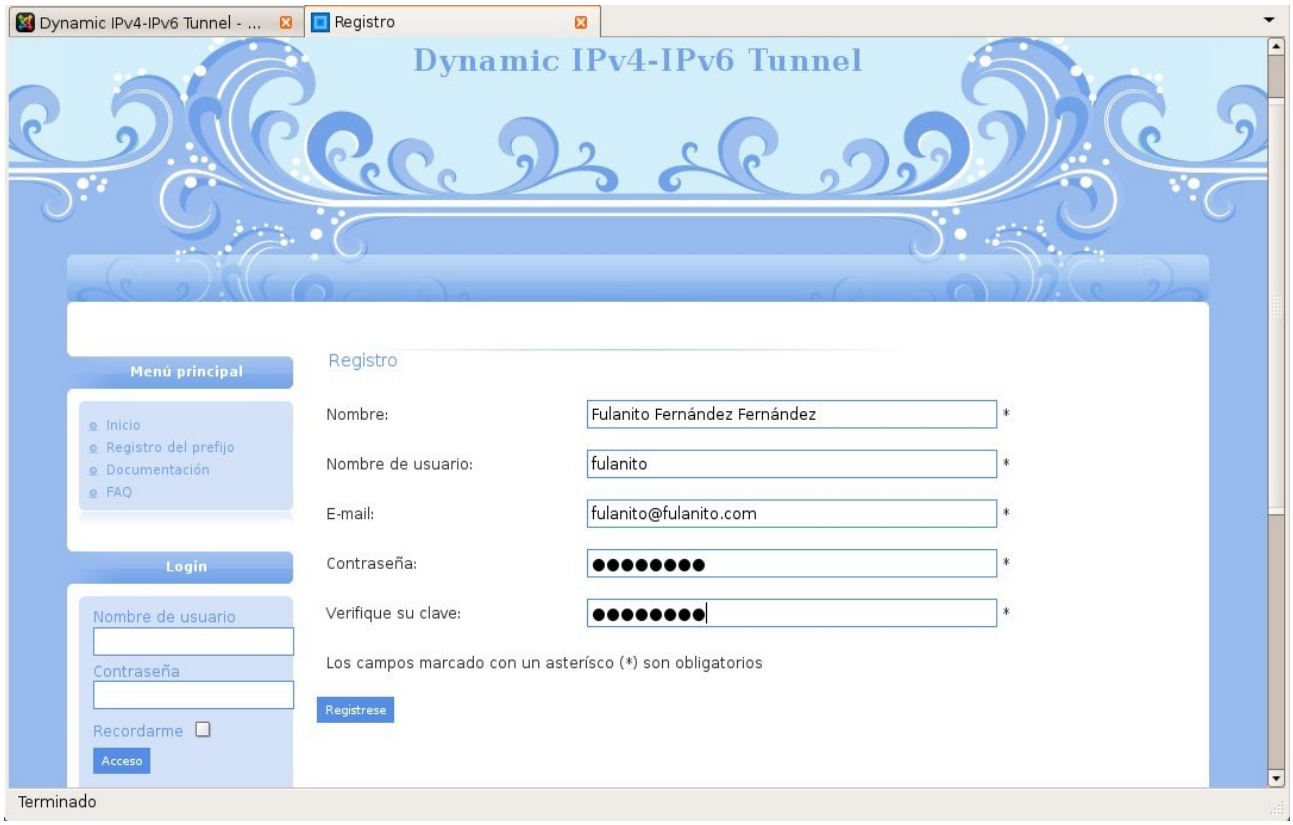


Figura 1.2. Aspecto del formulario para registrase como nuevo cliente.

Según haya decido el administrador del sitio, pueden ocurrir las siguientes dos situaciones:

- El registro ha finalizado. El cliente ya puede inicializar la sesión.
- El registro todavía no ha finalizado. Se le mandado al potencial cliente, un correo electrónico a la cuenta de correo que ha introducido, con un enlace que deberá seguir. Si el potencial cliente sigue el enlace, se activará la cuenta. Por tanto el proceso no se completará hasta que el cliente siga el enlace.

Una vez finalizado el proceso de registro el usuario ya podrá iniciar la sesión.

Como se aprecia en la figura de la izquierda, simplemente hay que introducir el nombre de usuario y contraseña en el recuadro Login y pulsar sobre el botón *Iniciar sesión*.

Una vez iniciada la sesión, el recuadro Login se transforma en el que aparece en la figura de la izquierda.

c. Obtener un prefijo IPv6

Los requisitos mínimos para obtener una prefijo IPv6 son:

- Ser un usuario registrado
- Tener la cuenta activada
- Haber iniciado la sesión.

Para poder solicitar un prefijo, se debe elegir la opción registro del menú de opciones.

El cliente puede pedir un prefijo de la longitud que estime oportuno (siempre que haya direcciones disponibles) y con un tiempo de vida comprendido entre un día e infinito.

El aspecto del formulario para solicitar un prefijo es el siguiente:

Figura 1.3. Formulario para solicitar un prefijo.

Como se aprecia en la Figura 1.3 simplemente hay que rellenar los datos pedidos y pulsar sobre el botón Dar de alta.
Si se desea un tiempo de vida indefinido ponga en la casilla un 0.
Una vez el proceso de alta haya finalizado satisfactoriamente, se le mostrará la pantalla una pantalla como la siguiente:

Figura 1.4. Formulario de edición del prefijo solicitado.

Como se puede apreciar, en dicho formulario se muestran todos los datos necesarios para configurar su red, entre ellos el prefijo que se ha generado.

D. Modificar el prefijo IPv6

El cliente no puede modificar el prefijo generado, es decir, tanto el prefijo propiamente dicho como la longitud del prefijo no se pueden modificar una vez dados de alta. Sin embargo, si se desea cambiar de prefijo, el cliente puede borrar el prefijo actual y registrar una nuevo con la nueva longitud deseada.

El único parámetro que el cliente puede modificar es el tiempo de vida.

E. Borrar el prefijo IPv6

El cliente puede dar de baja el prefijo que le haya sido concedido en cualquier momento, simplemente debe presionar sobre el botón borrar en el formulario de edición del prefijo.

III. Manejo de la administración (back end).

A. Introducción a la administración

En la parte de la administración se podrán realizar las siguientes tareas (entre otras) :

- Configurar el servidor (Ftp, Smtp,...)
- Gestionar las usuarios (dar permisos de administración, deshabilitar un usuario, cerrar la sesión de un usuario,...)
- Configuración de la página web (tiempo de duración de la sesión, número de items por página, idioma, hora, caché, SSL,...)
- Cambiar la plantilla (aspecto de la página web)
- Administrar el contenido de la página de inicio
- Gestionar el contenido de la página web (crear artículos, secciones y categorías)
- Gestionar las entidades del tunnel broker.

Por ello, a la parte de la administración sólo van a tener acceso un reducido grupo de usuarios, en concreto aquellos a los que se les haya concedido permisos de administración.

B. Acceso a la administración

La ruta para acceder a la administración es la misma que a la página del sitio seguida de '/administrator'.

Por ejemplo, si al sitio se accede con la dirección http://www.ejemplo.com a la administración se accederá con la dirección http://www.ejemplo.com/administrator.

La figura 2.1 muestra la página de inicio a la parte de la administración. Por defecto, sólo un usuario tiene permisos para acceder a dicha parte, los datos a introducir son:

- Usuario: administrator
- Contraseña: administrator

Figura 2.1. Página de acceso a la administración.

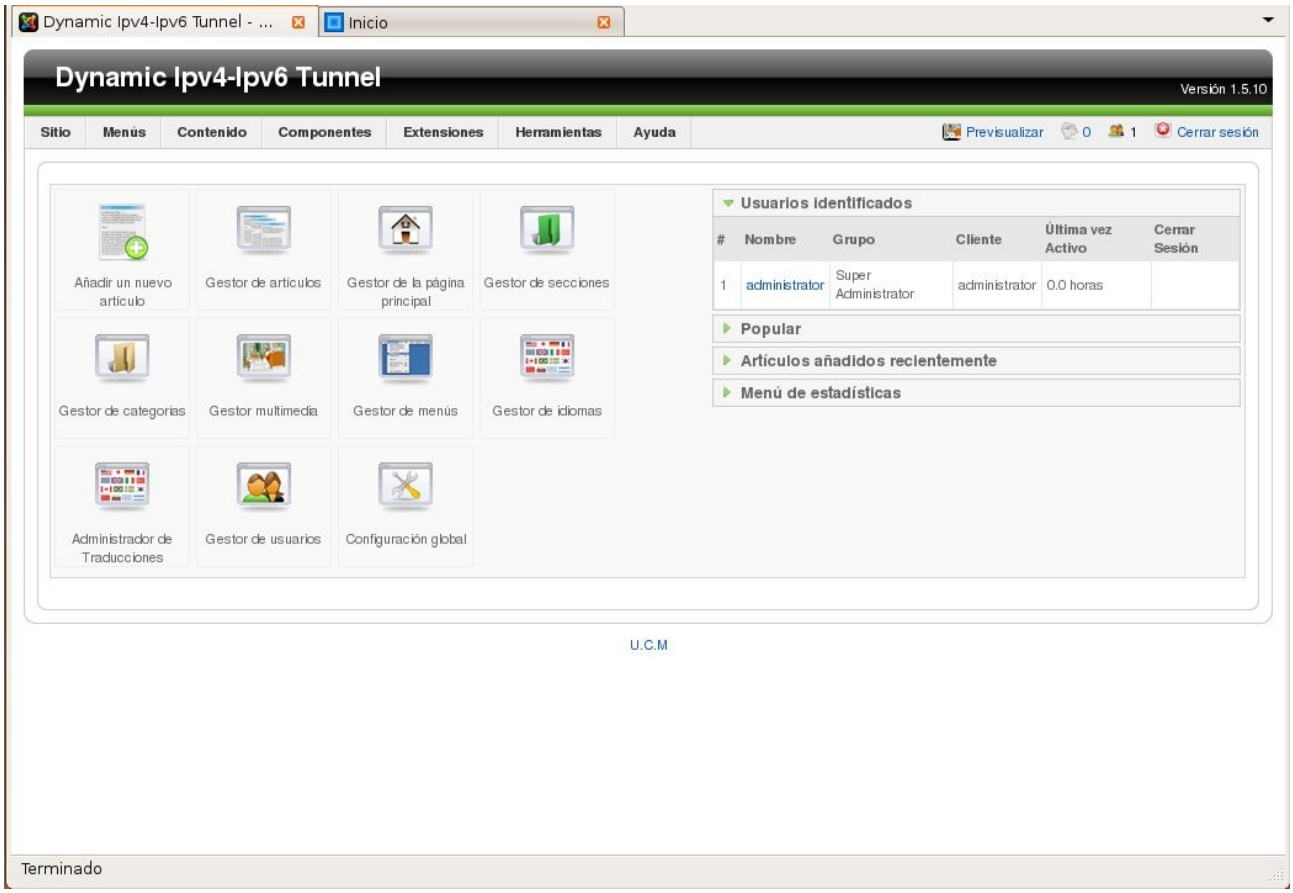


Figura 2.2. Página de bienvenida de la administración.

C. Configuración básica del Servidor

Unas de las primeras tareas a realizar una vez instalada la aplicación es configurarla correctamente. En esta sección se va abordar la configuración básica de la aplicación.

Resumiendo, los servicios y tareas a configurar son:

- El servicio de correo
- El servicio FTP
- La seguridad SSL
- Cambio del usuario por defecto

Si se desean configurar aspectos que no cubre este manual, se sugiere leer un manual avanzado de Joomla!1.5.x.

La mayoría de los aspectos de configuración se pueden gestionar a través del enlace Sitio->Configuración Global.

Configuración del Correo electrónico.

Joomla! tiene dos maneras de enviar los correos electrónicos, mediante Sendmail y mediante el protocolo SMTP. Para que funcionen correctamente, es necesario configurarlos.

Las funciones de correo se configuran en: Sitio->Configuración Global->Servidor.

En la pestaña servidor, hay una sección llamada C*onfiguración del correo* donde se deben rellenar los parámetros de configuración.

Consulte con su proveedor de servicio si tiene alguna duda acerca de los valores de los parámetros.

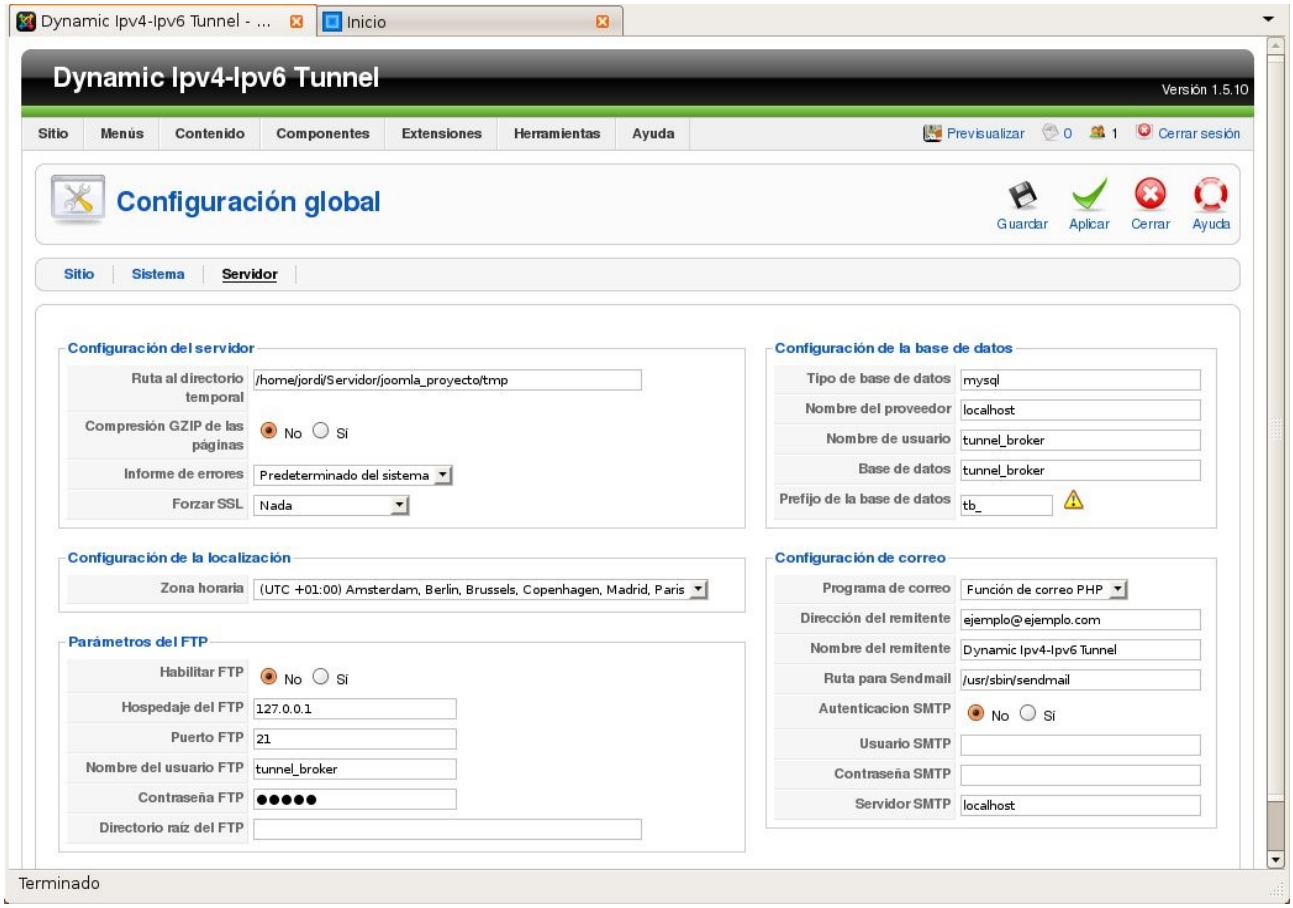


Figura 2.3. Pestaña de configuración del servidor.

Configuración del FTP.

Este servicio es opcional aunque recomendable si se tiene alojada la aplicación en un servidor sin acceso físico, ya que permite operaciones de archivo (como instalar extensiones o actualizar la configuración principal) sin tener que dar permisos de escritura a los directorios que se quieran modificar.

Las funciones de FTP se configuran en: Sitio->Configuración Global->Servidor.

En la pestaña servidor, hay una sección llamada *parámetros del FTP* donde se deben rellenar los parámetros de configuración.

Consulte con su proveedor de servicio si tiene alguna duda acerca de los valores de los parámetros.

Seguridad SSL.

Si se desea cifrar mediante el protocolo SSL, es necesario obtener un certificado de una autoridad reconocida e instalarla correctamente en el servidor.

Una vez resueltos estos requisitos simplemente se debe indicar a Joomla! que utilice una conexión segura.

El SSL se configura en: Sitio->Configuración Global->Servidor.

En la pestaña servidor, hay una sección llamada *Configuración del servidor* donde se encuentra una variable llamada *Forzar SSL*. Es ahí donde se indica de forma genérica si se desea utilizar SSL para la administración o para el sitio entero.

También existe la posibilidad de indicar cuáles son los menús que deben ir cifrados con ssl y cuáles no. Consulte un manual avanzado de Joomla!1.5.x si desea profundizar en dicho aspecto.

Cambio de usuario por defecto

Es recomendable crear un usuario nuevo con permisos de super administrador y borrar el que viene por defecto (administrator).

Para ello, cree un nuevo usuario haciendo click en Sitio->Gestor de usuarios en el menú superior con permisos de super administrador y borre el que venía por defecto.

Como no se pueden borrar usuarios con privilegios de super administrador, primero hay que degradar su categoría (a cualquiera inferior) antes de poderlo borrar.

D. Gestión de usuarios

Hay 7 tipos de usuarios, cada uno con privilegios diferentes. Los tipos de usuarios se dividen en los que tienen acceso al sito y los que tienen acceso a ambos.

Usuarios sólo del sito:

- Invitado
- Registrado
- Autor
- Editor
- Supervisor

Usuario del sito y de la administración:

- Gestor
- Administrador
- Super Administrador

Hay que tener en cuenta que Joomla! está pensado para que todo el mundo pueda participar editando y creando artículos que más tarde se podrán publicar.

A nuestros efectos, no tiene sentido una clasificación tan extensa, nos valdría simplemente con 2 tipos de usuarios: registrado y super administrador.

De hecho, la aplicación sólo dejará a los Super Administradores manejar todo lo concerniente a los túneles y prefijos en la administración.

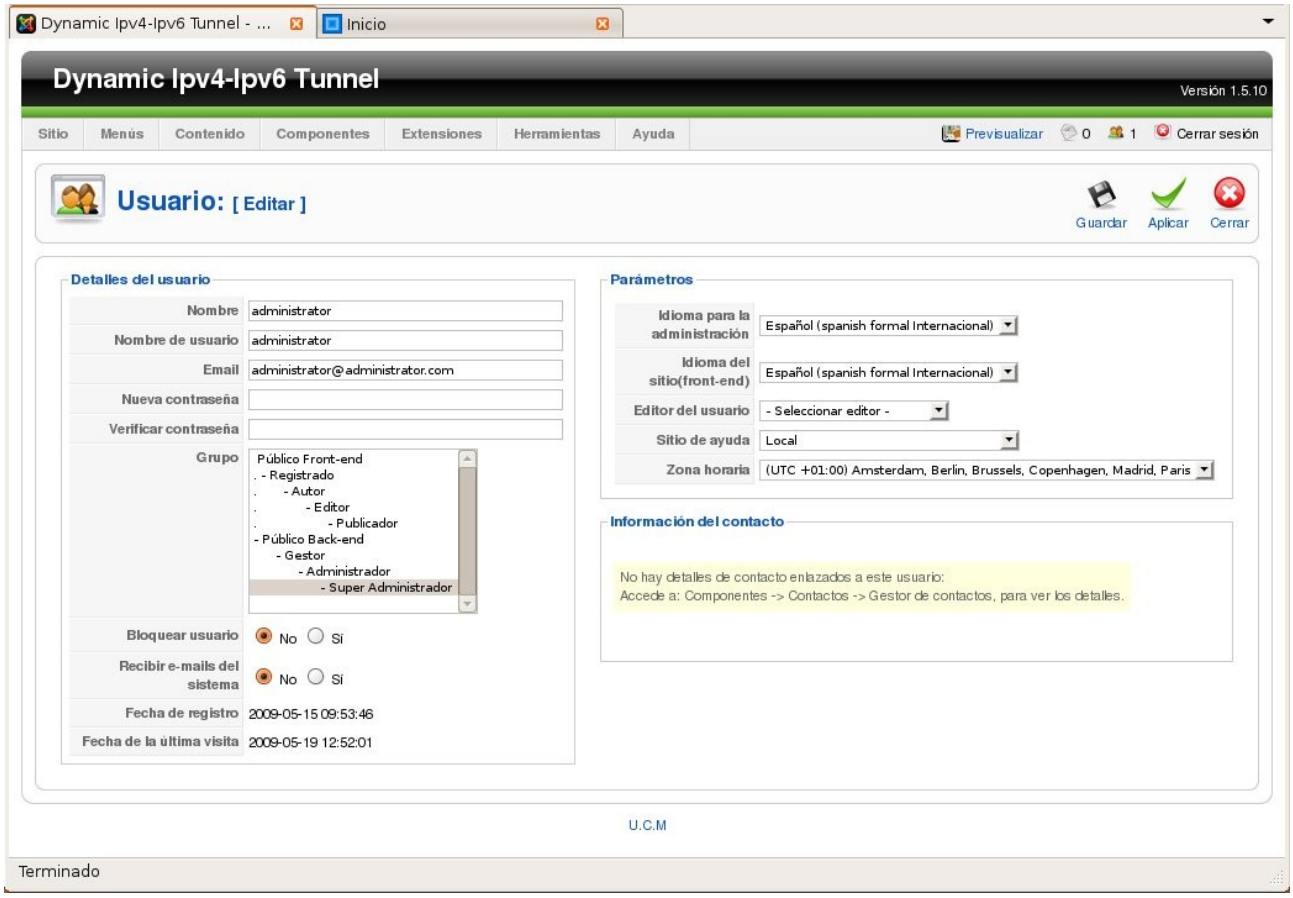


Figura 2.4. Detalle del usuario Super Administrador por defecto.

E. Estructura de los contenidos de Joomla!

Joomla! es un sistema gestor de contenidos, fue pensado e ideado para realizar esta tarea de forma muy eficiente aunque por ello es necesario una pequeña explicación del funcionamiento.

La unidad mínima de un gestor de contenidos es el artículo. Este puede contener, además de texto, elementos multimedia como imágenes. Todos los artículos se pueden categorizar, esto es, asignarles una categoría, y todas las categorías se pueden agrupar en secciones. Por tanto en Joomla! se pueden crear secciones (tantas como se quiera), que contendrán categorías (tantas como se quiera) que contendrán artículos (tantos como se quiera).

En nuestra aplicación se han dado de alta tres secciones:

- FAQ

- Documentación
- Otros

Además se han dado de alta algunas categorías. Por ejemplo, en Documentación, se han dado de alta las siguientes categorías:

- Documentación de la interfaz web.
- Documentación del programa cliente.

Esta es la clasificación que viene por defecto en la aplicación, pero es totalmente personalizable a nuestras necesidades.

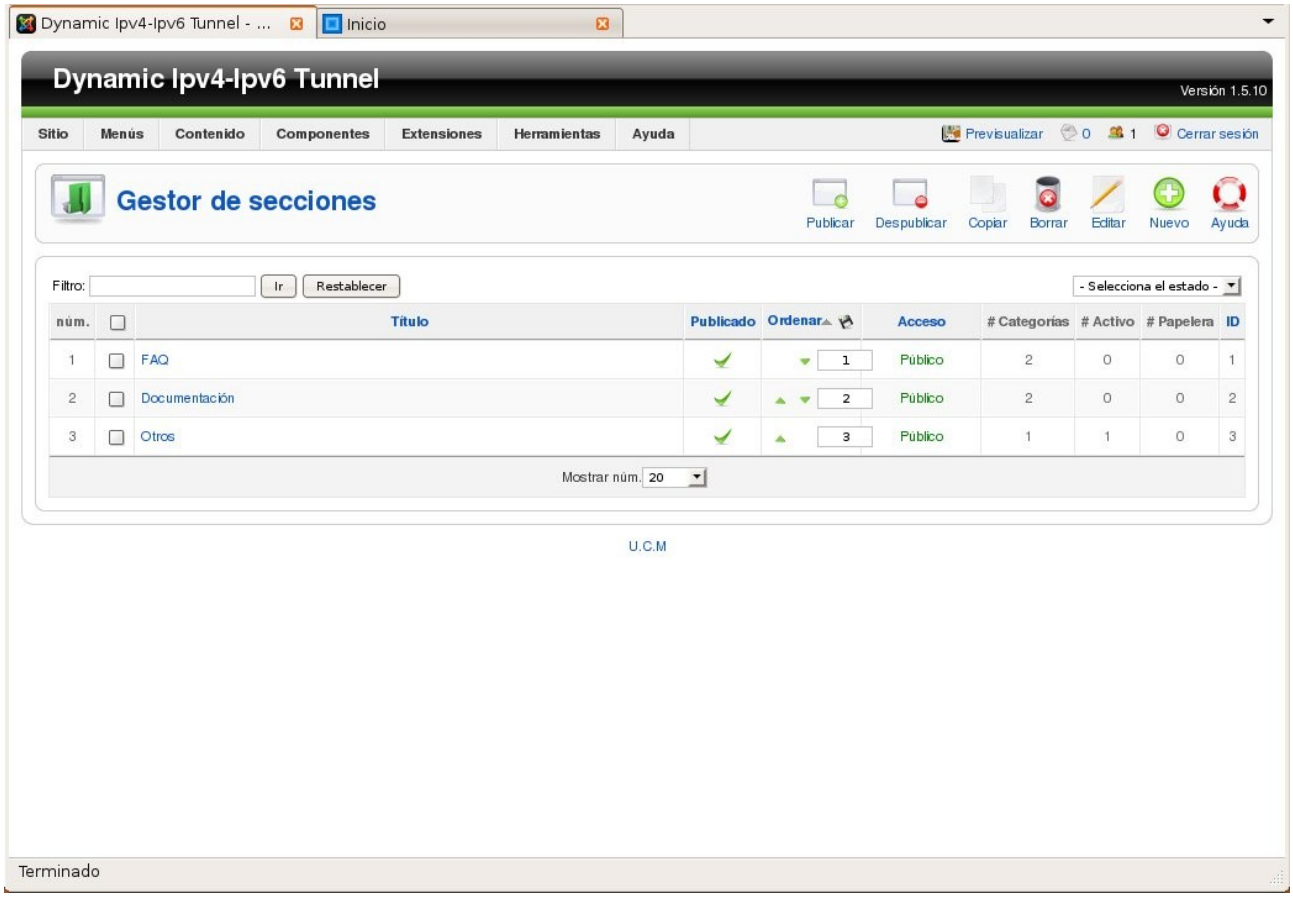


Figura 2.5. Detalle del listado de secciones dadas de alta por defecto.

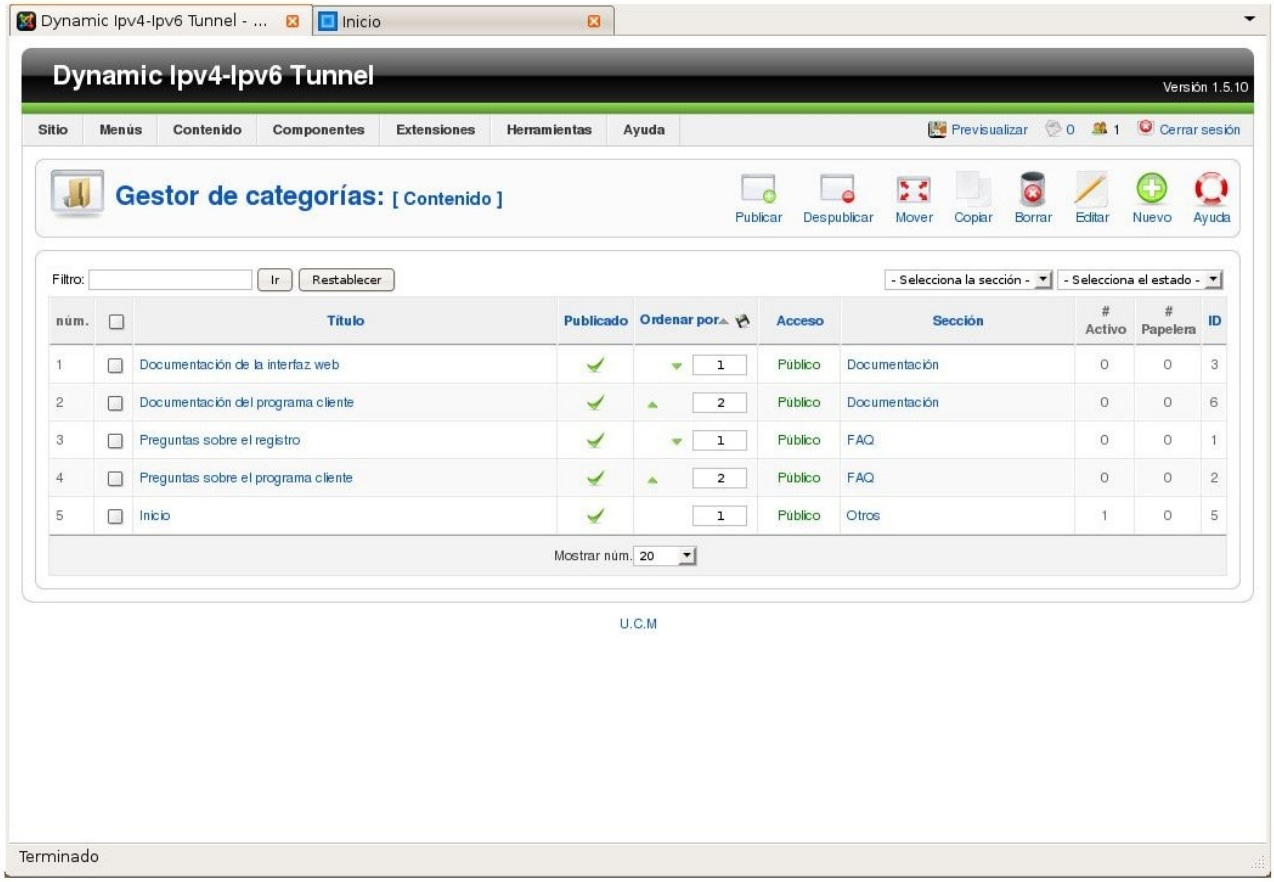


Figura 2.6. Listado de todas las categorías dadas de alta por defecto.

F. Componente Tunnel Broker

Hasta ahora, sólo se han abordado asuntos propios de Joomla!, pero lo realmente importante de la aplicación es la interfaz que proporciona para gestionar todas las entidades que conforman el tunnel broker.

Se accede a la gestión de la componente haciendo click en Componentes->Tunnel Broker. Aparecerá la pantalla de la figura 2.7.

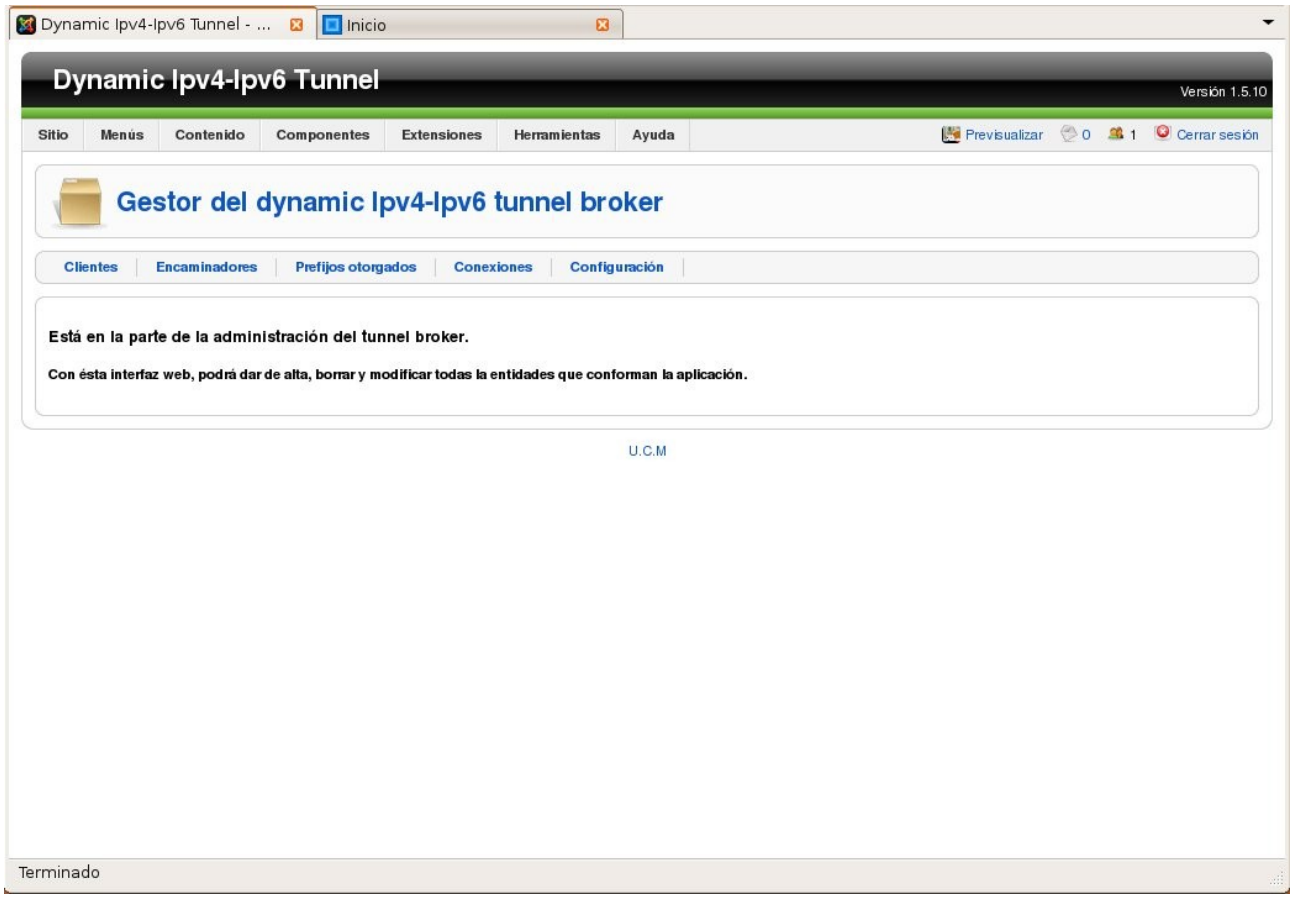


Figura 2.7. Pantalla de inicio de la componente que gestiona el tunnel broker.

Hay cuatro entidades que deben poder gestionarse correctamente:

- Clientes
- Encaminadores
- Prefijos
- Conexiones

Por cada una de ellas hay un submenú.

Clientes

Como es lógico, se ha aprovechado el sistema de usuarios de Joomla! para gestionar los clientes, por tanto, un usuario registrado de Joomla! será un cliente potencial del tunnel broker.

Es posible que un cliente esté registrado y no tenga un prefijo, en dicho caso no podrá utilizar el servicio de túneles hasta que no obtenga uno.

Si se hace click en el submenú clientes, aparecerá un listado con todos los clientes dados de alta en la aplicación, incluyendo los bloqueados. Dicho listado se puede ordenar por columnas, haciendo click en la columna por la que queramos ordenar.

Si se hace click en alguno de los clientes del listado, aparecerán los detalles de dicho cliente.

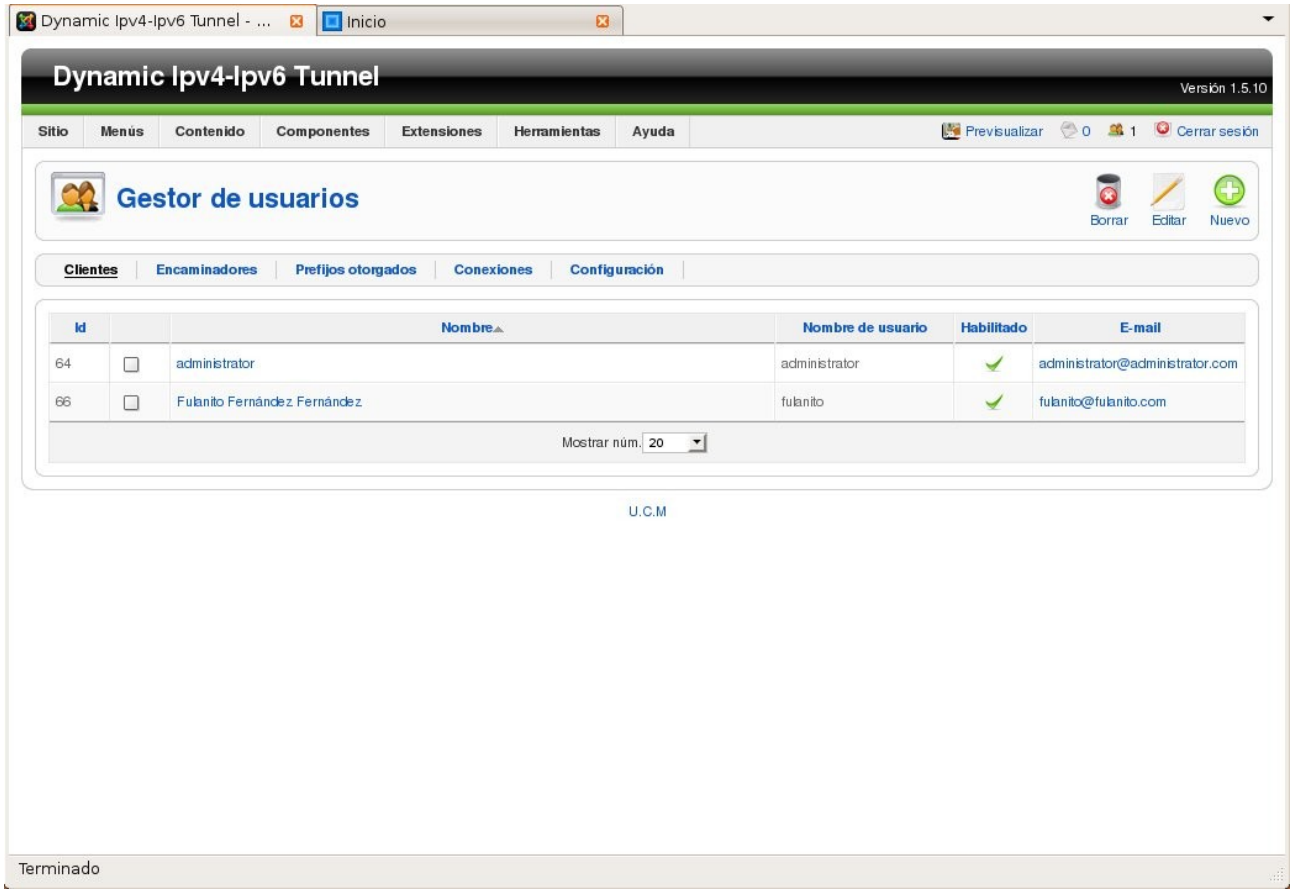


Figura 2.8. Listado de usuarios dados de alta.

Encaminadores

La aplicación ha sido diseñada para que pueda haber uno o más encaminadores, de ésta forma se consigue una infraestructura escalable.

Por defecto, no viene ningún encaminador dado de alta, pero para que funcione correctamente, es necesario que al menos haya un encaminador dado de alta.

Si se hace click en el submenú *encaminadores*, aparecerá un listado con todos los encaminadores dados de alta en la aplicación. Dicho listado se puede ordenar por columnas, haciendo click en la columna por la que queramos ordenar.

Si se hace click en alguno de los encaminadores del listado, aparecerán los detalles de dicho encaminador.

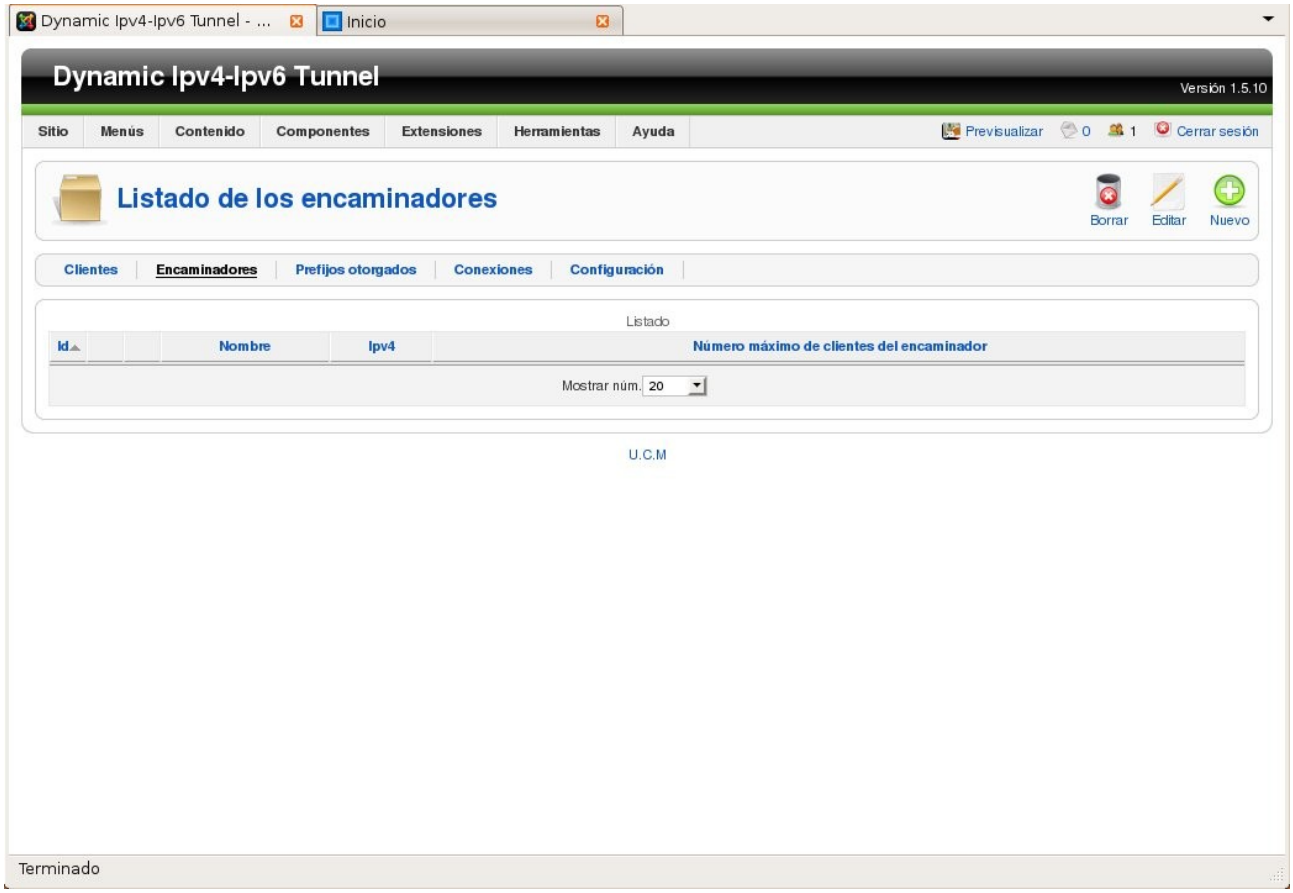


Figura 2.9. Listado de los encaminadores por defecto.

Para dar de alta un nuevo encaminador, pulsar sobre el botón Nuevo una vez haya accedido al submenú e*ncaminadores*. Aparecerá el formulario de la figura 2.10 el cual se deberá rellenar correctamente.

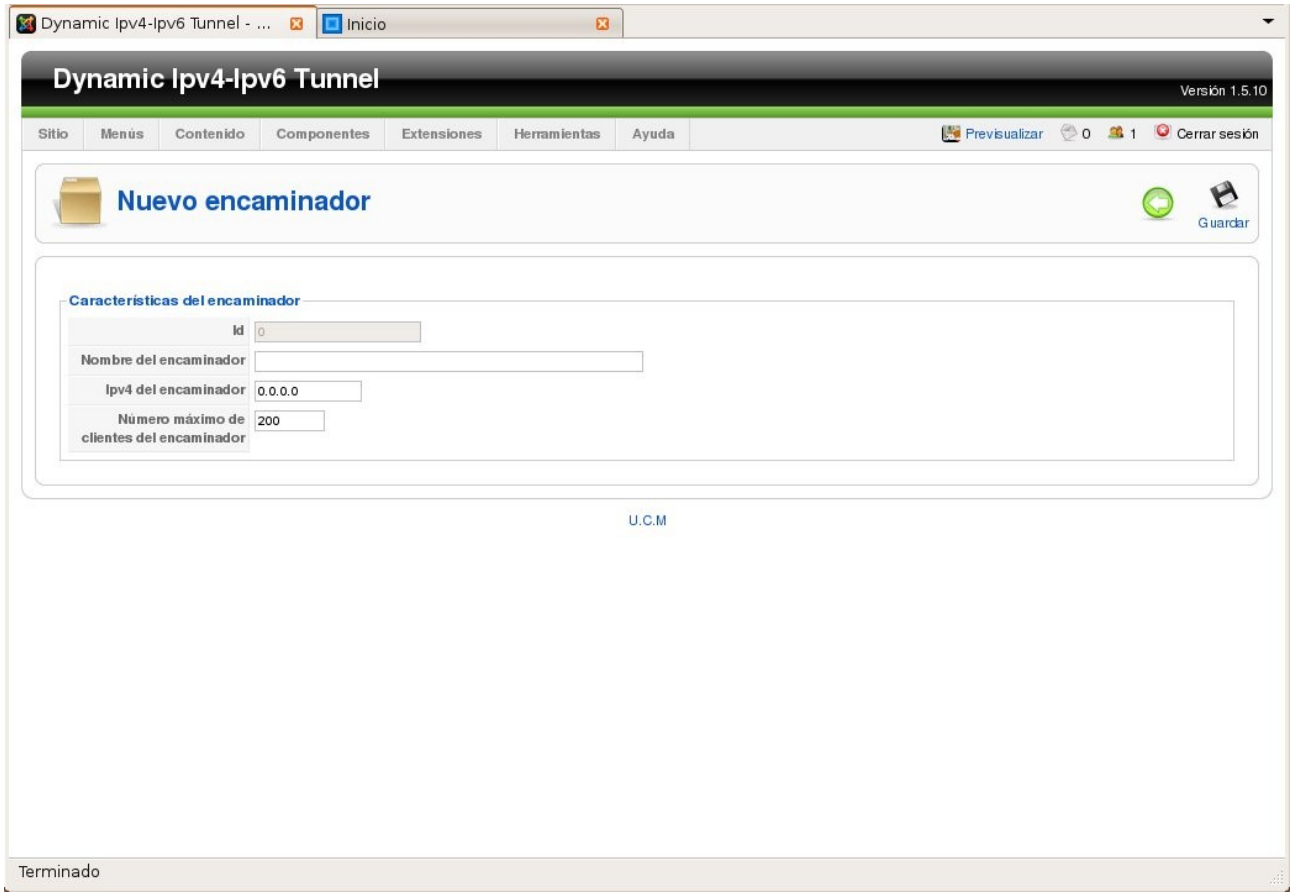


Figura 2.10. Formulario de alta de un encaminador.

Prefijos

La aplicación tiene asignado un prefijo, de forma que a sus clientes les asignará prefijos contenidos estrictamente en su propio prefijo. Por tanto, cuanto menor sea la longitud del prefijo, mayor número de direcciones dispondrá y mayor número de prefijos podrá asignar a los clientes.

Sólo se podrán asignar prefijos a los clientes dados de alta.

Si se hace click en el submenú *prefijos otorgados*, aparecerá un listado con todos los prefijos asignados. Dicho listado se puede ordenar por columnas, haciendo click en la columna por la que queramos ordenar.

Si se hace click en alguno de los prefijos del listado, aparecerán los detalles de dicho prefijo.

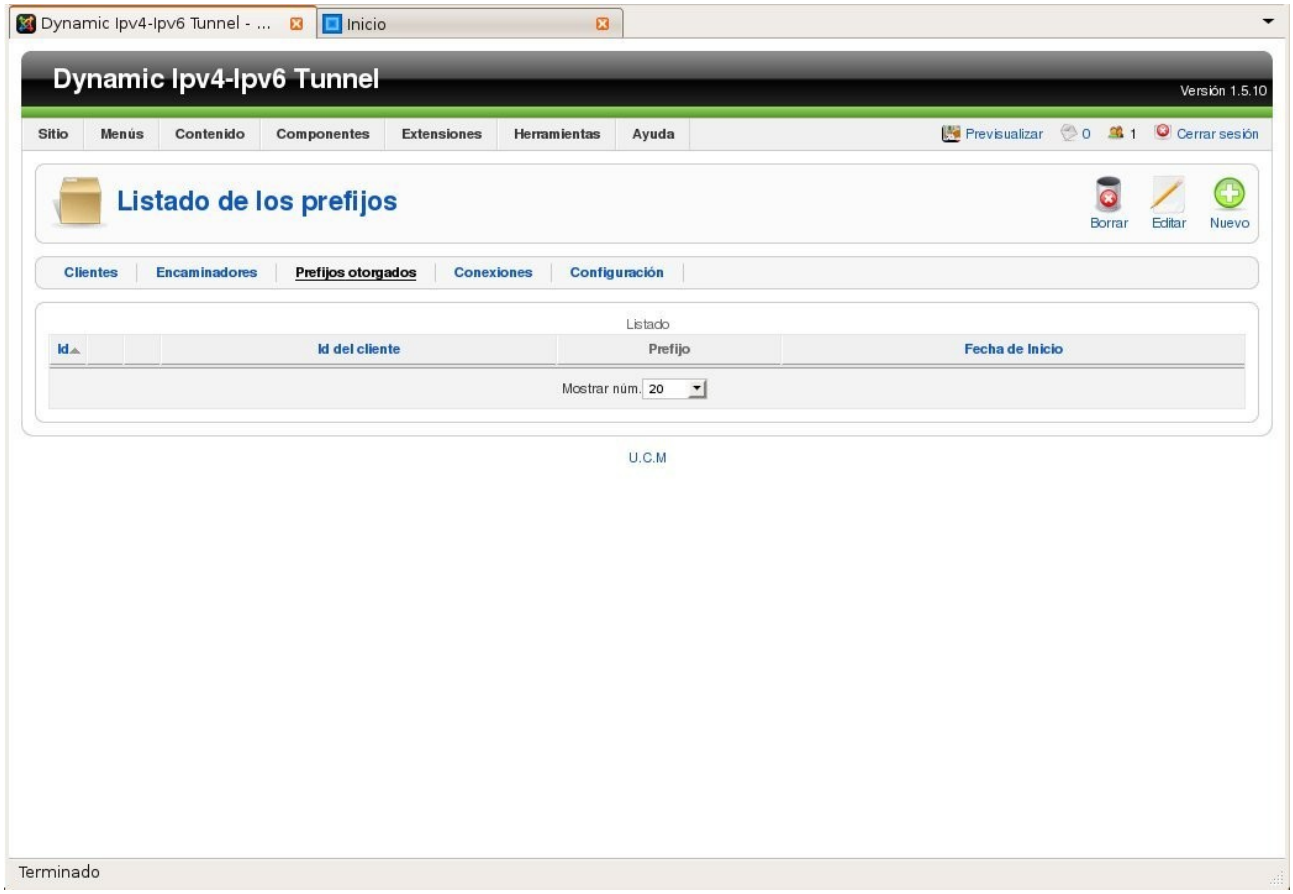


Figura 2.11. Listado de los prefijos asignados (inicialmente vacío).

Para dar de alta un nuevo prefijo, pulsar sobre el botón Nuevo una vez haya accedido al submenú *Prefijos otorgados*. Aparecerá el formulario de la figura 2.12 el cual se deberá rellenar correctamente.

Hay que señalar que en la gestión normal de la web, no se darán de alta manualmente los prefijos, cada cliente obtendrá el suyo en el sito. Aún así, la aplicación permite un control total de la entidad.

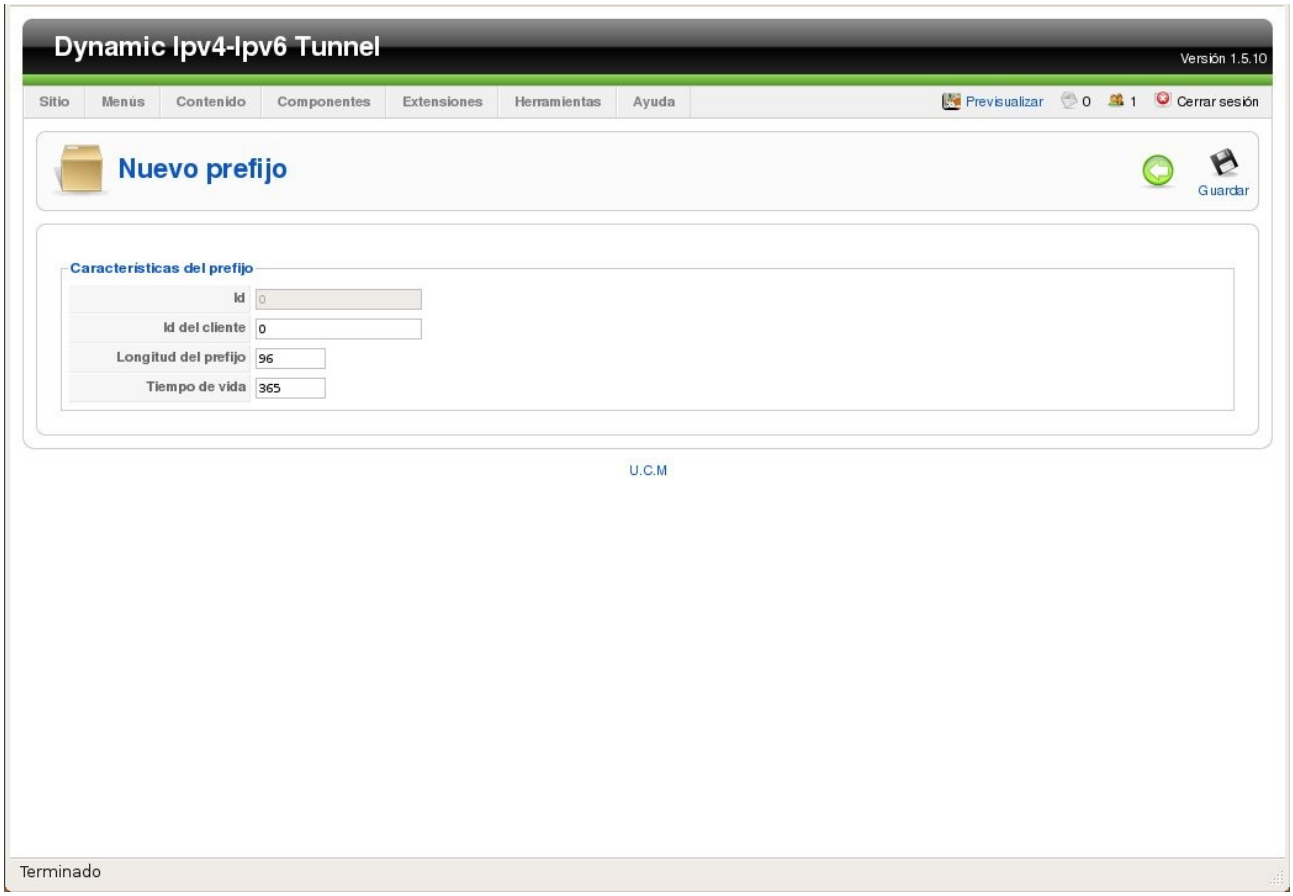


Figura 2.12. Formulario para dar de alta un prefijo.

Conexiones

La entidad conexión representa un enlace entre la interfaz IPv4 de un cliente y un encaminador de la aplicación. Dicho enlace debe mantenerse activo mientras se sigan recibiendo los keep alive.

Por defecto, no hay ninguna conexión dada de alta, es más, desde la interfaz web nunca se darán de alta ya que ese cometido exclusivo de la aplicación del servidor. No obstante, se debe dar la posibilidad gestionar las conexiones de forma manual.

Si se hace click en el submenú *conexiones*, aparecerá un listado con todos las conexiones dadas de alta en la aplicación. Dicho listado se puede ordenar por columnas, haciendo click en la columna por la que queramos ordenar.

Si se hace click en alguno de las conexiones del listado, aparecerán los detalles de dicha conexión.

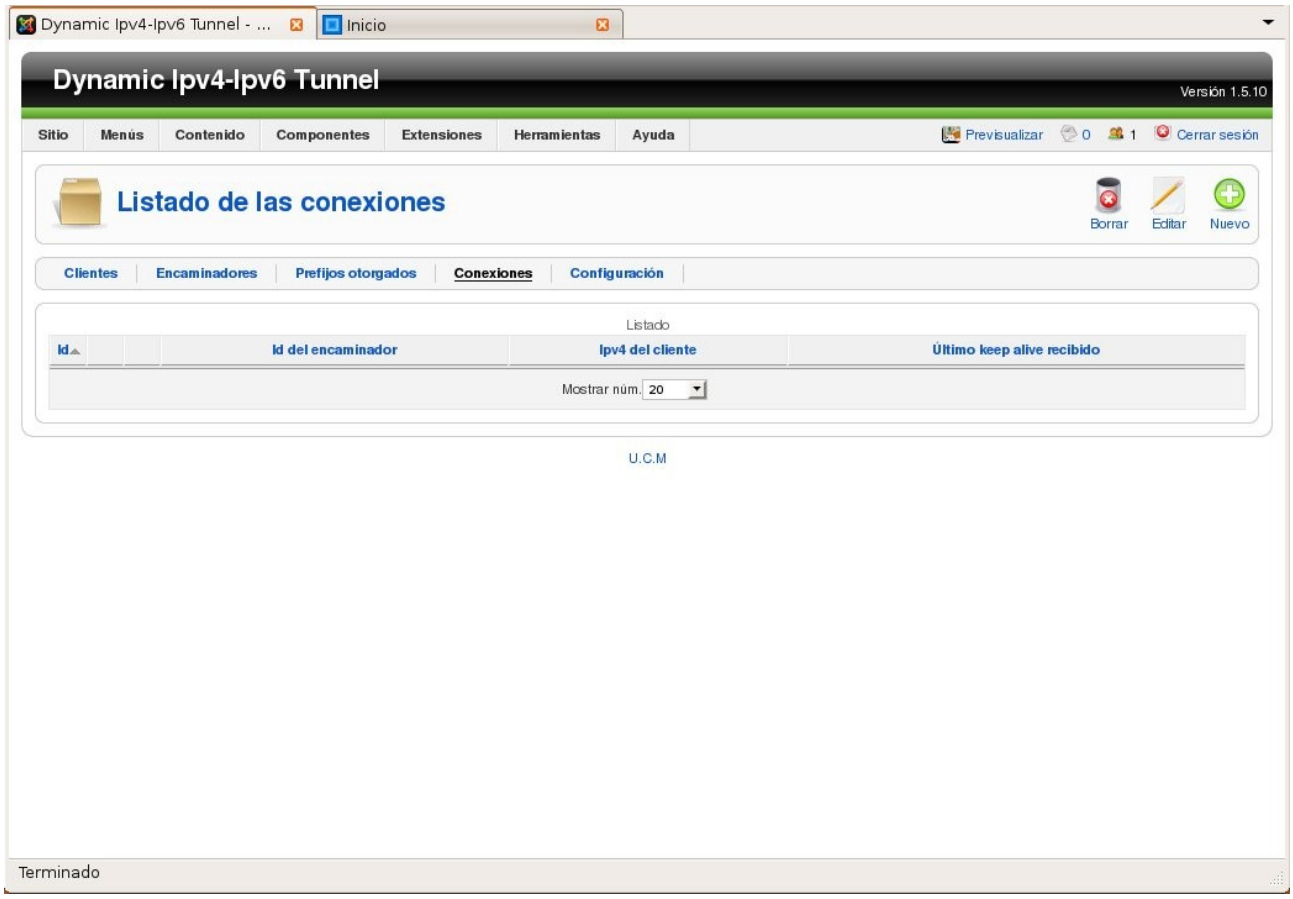


Figura 2.13. Listado vacío de conexiones.

Para dar de alta una nueva conexión, pulsar sobre el botón Nuevo una vez se haya accedido al submenú *conexiones*. Aparecerá el formulario de la figura 2.14 el cual se deberá rellenar correctamente.

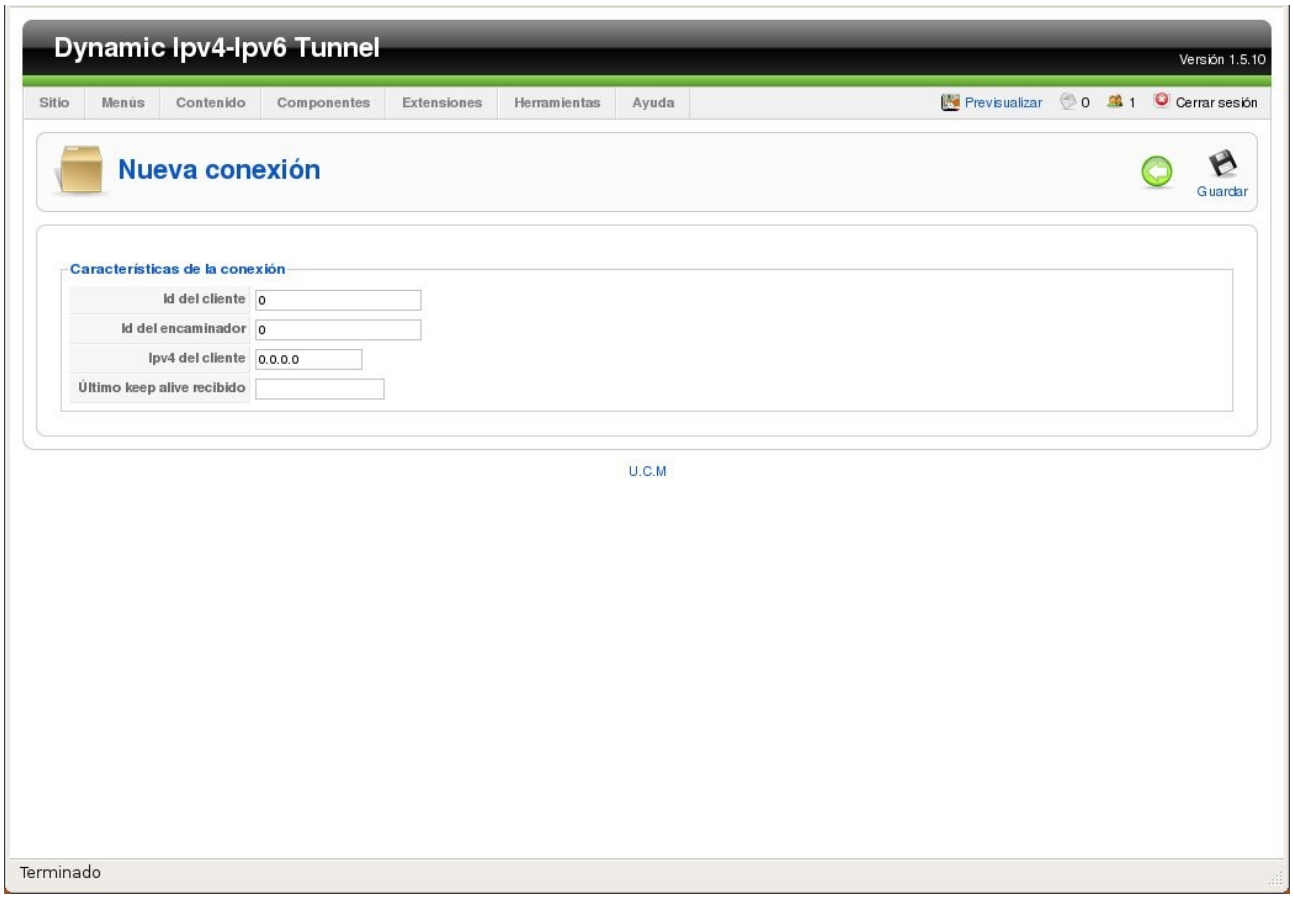


Figura 2.14. Formulario para dar de alta una nueva conexión.

Configuración

En esta pestaña se configuran los parámetros necesarios para el correcto funcionamiento de la aplicación.

El parámetro más importante es el prefijo que se le ha asignado a la aplicación y del cual se conseguirán subprefijos para los clientes.

A la hora se asignar los prefijos, se pueden seguir 3 políticas diferentes:

- **Primero el mejor**: asigna el prefijo que corresponda con el primer hueco que se encuentre y sea suficientemente grande. Este es el algoritmo mas rápido de todos.
- **Mejor ajuste**: asigna el prefijo que corresponda con el primer hueco que se encuentre y sea exactamente del tamaño pedido. En caso de que no se encuentre ninguno, se asigna el que más se ajuste a su tamaño. Este algoritmo favorece un mejor aprovechamiento del

espacio de direcciones a costa de una fragmentación con huecos muy pequeños y un mayor tiempo de ejecución.

- **Peor ajuste**: asigna el prefijo que corresponda con el hueco más grande de todos los que haya. Este algoritmo favorece que los huecos de la fragmentación no sean muy pequeños a costa de ser el más lento de todos.

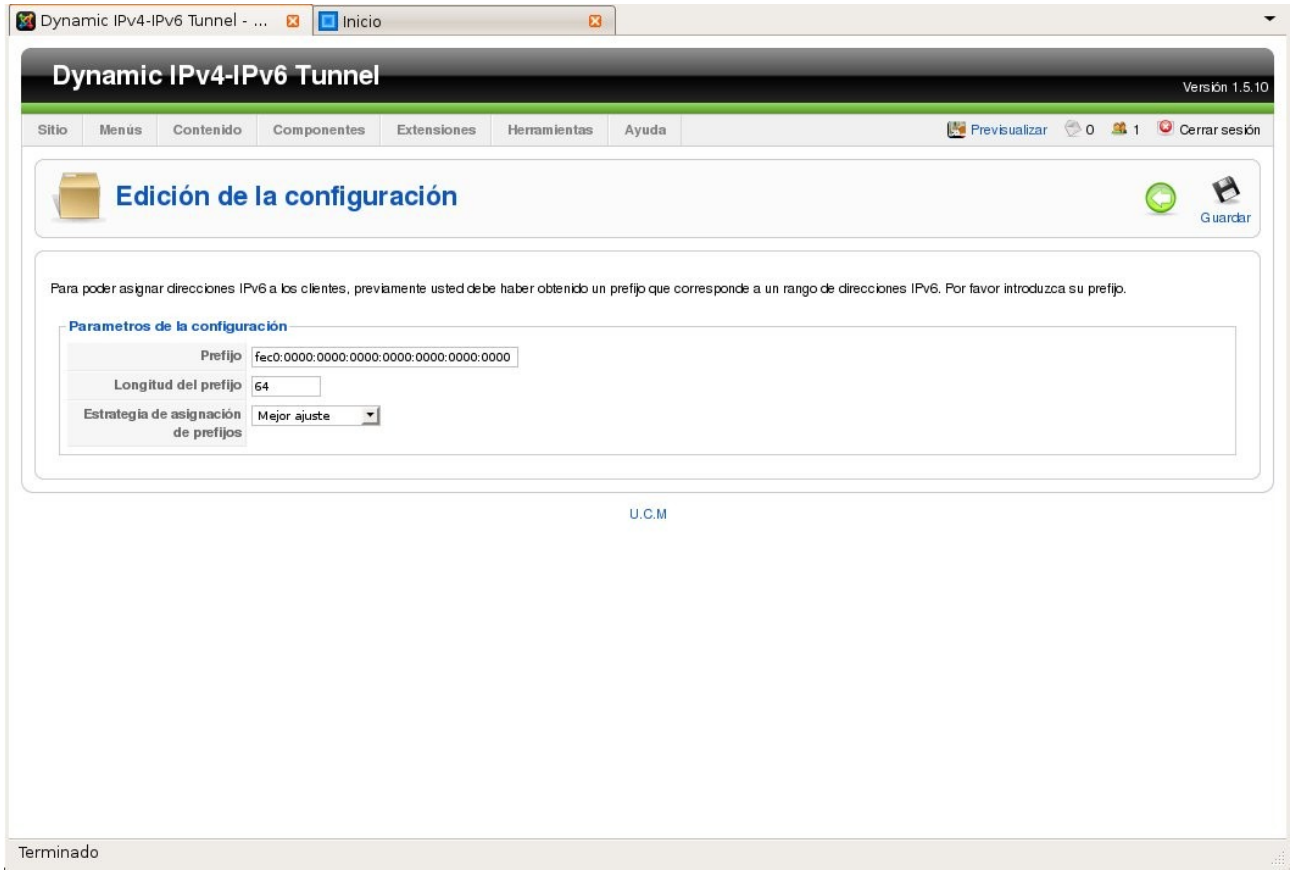


Figura 2.15. Formulario de edición de la configuración.