UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Departamento de Computación

PROYECTO DE FIN DE CARRERA DE INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

Gestor de Ancho de Banda en Linux con control de acceso de usuarios.

Autor: López Pérez, Carlos Alberto
Tutor: Caridad Simón, Serafín
Director: González Penedo, Manuel Francisco

A Coruña, Septiembre de 2009

Gestor de Ancho de Banda en Linux con control de acceso de usuarios.

Tipo de proyecto:

Investigación y desarrollo

Autor:

Carlos Alberto López Pérez

Director:

Manuel Francisco González Penedo

Tutor:

Serafín Caridad Simón

Tribunal:

Serafín Caridad Simón
Manuel Francisco González Penedo

Director Externo:

Eduardo Ozores Baltar (Ubalda Ingeniería SL)

Fecha de Lectura:

30 de Septiembre de 2009

Calificación:

Agradecimientos

Quiero agradecer especialmente a Eduardo Ozores la oportunidad de haber realizado el presente proyecto en la empresa Ubalda Ingeniería, gracias al cual he podido ampliar enormemente mis conocimientos sobre redes de ordenadores y sobre el maravilloso sistema operativo GNU/Linux.

No quisiera dejar pasar la oportunidad para agradecer también a toda la comunidad *“Open Source”* su trabajo desinteresado y altruista gracias al cual ponen a disposición de todo el mundo su conocimiento sobre los sistemas computacionales.

Dedicado:

A mi madre María Teresa
A mi padre José Antonio
A mi hermano Javier

Resumen

El principal objetivo del presente proyecto es el desarrollo de un sistema de gestión de ancho de banda utilizando herramientas "*open source*". Dicho sistema hará de pasarela a Internet para los usuarios que se conecten a través de el, a la vez que limitará el ancho de banda que cada usuario puede utilizar en base al caudal que dicho usuario halla contratado. Además el sistema aplicará diversos métodos de control del tráfico TCP/IP para asegurar que el usuario obtiene la máxima calidad del servicio posible en cada momento.

Otro punto importante del sistema es la gestión de la seguridad del acceso, para ello implementará diversas técnicas de encriptación para asegurar la privacidad y confidencialidad de los datos transmitidos por los usuarios del sistema. Además se implementan diversas técnicas de filtrado que impiden a los usuarios hacer un uso abusivo o mal intencionado de la red. Todos los accesos al sistema quedarán registrados en un *log* de forma que se pueda consultar el mismo en caso de que fuera necesario.

Se pondrá especial énfasis en la configuración e implementación de la red inalámbrica (WiFi) ya que se prevé que la gran mayoría de los usuarios se conecten al sistema de forma inalámbrica.

Los usuarios, además de tener limitado el ancho de banda de subida y bajada tendrán un limite de acceso al sistema, que puede ser una fecha límite hasta la cual pueden tener acceso o bien un consumo tope de tráfico de datos (en Gigabits).

El sistema se diseña con dos conceptos en mente: facilidad de uso y seguridad. El usuario que se conecta a través del sistema no tiene que configurar nada en su equipo, obtiene conectividad automáticamente. El sistema presenta también una interfaz de administración sencilla y amigable que se puede configurar y gestionar desde cualquier PC con navegador Web.

El sistema está pensado para hacer de HotSpot[1] Inalámbrico en lugares públicos, cafeterías, clubes, empresas u organizaciones. El usuario

[1] Un hotspot (en inglés 'punto caliente') es una zona de cobertura Wi-Fi, en el que un punto de acceso (access point) o varios proveen servicios de red a través de un Proveedor de Servicios de Internet Inalámbrico (WISP).

itinerante deberá adquirir una cuenta de usuario para poder conectarse a Internet, para ello puede optar por adquirirla físicamente en la recepción o secretaría del evento o puede comprarla de forma on-line a través del servicio de pago PayPal. También se contempla la posibilidad de instalar una máquina de monedas tipo "*vending*" junto con una impresora de *tickets* para automatizar el proceso de venta de cuentas de usuario.

El proyecto se realiza con una beca FEUGA en la empresa Ubalda Ingeniería SL, la cual lo utilizará como HotSpot en su proyecto de WISP[2].

El proyecto incluye además un estudio sobre la filosofía "*open source*", así como de los diversos modelos de negocio que se engloban dentro de dicho marco

Palabras clave: Linux, iptables, tc, netfilter, hostapd, poptop, radius, ldap, dhcp, named, atheros, wpa, peap, layer7, apache, ebtables, htb, sfq, qdisc, kernel, imq, cansas, openwrt, ubuntu, 802.1x, iproute2, igd, upnp, ipp2p, bind, ubuntu, openwrt, vpn, pptp, ppp, open source, hotspot, wisp, ssl, wifi.

[2] WISP (*Wireless Internet Service Provider*) Proveedor Inalámbrico de Servicios de Internet.

Capitulo 1. Introducción

En los últimos años el acceso a Internet se ha universalizado y ha dejado de ser un privilegio para convertirse en una necesidad. Cada vez somos más dependientes de los servicios que nos ofrece Internet, y muchas veces necesitamos acceder a la red fuera de nuestro hogar.

Actualmente la mayoría de la gente posee un ordenador portátil. De hecho en el último año se ha producido un *boom* de ventas de los denominados *netbooks*, miniportátiles de bajo coste diseñados para navegar por Internet. La tecnología inalámbrica WiFi (IIEE 802.11 [3]) se ha convertido en el estándar de facto para el acceso a Internet en los ordenadores portátiles. Hoy en día todos los operadores de Internet ofrecen routers WiFi al contratar la conexión y en muchas cafeterías, aeropuertos, comercios o edificios públicos se ofrece acceso a Internet gratuito mediante WiFi.

A todo esto tenemos que añadir que los teléfonos móviles cada vez incorporan funciones más avanzadas y no es extraño ver teléfonos con conectividad WiFi y navegador Web incorporado.

Todo esto provoca una demanda real de acceso a Internet en lugares públicos, cafeterías, comercios, aeropuertos, etc. por parte de los usuarios a la vez que plantea serios inconvenientes para el prestador del servicio.

Muchas cafeterías, por ejemplo, optan por dar acceso libre a Internet o protegido mediante una contraseña que facilitan al hacer una consumición. Pero podría ocurrir que uno de los usuarios de la conexión empezara a utilizar el ancho de banda disponible de forma indiscriminada provocando serios problemas de usabilidad del servicio al resto de usuarios. También podría ocurrir que un usuario aprovechara la conexión para hacer actos delictivos en Internet y metiese en serios aprietos legales al dueño del establecimiento. Igualmente es posible que un usuario mal intencionado espiase el trafico que están enviando y recibiendo los otros usuarios en busca de contraseñas o datos privados y confidenciales[4].

[3] http://en.wikipedia.org/wiki/IEEE_802.11

[4] Esto es muy sencillo de realizar en una red inalámbrica cuando el tráfico TCP/IP no esta encriptado o bien lo está con una contraseña conocida. Véase http://aircrack-ng.org/doku.php?id=airodump-ng

1.1. Descripción del problema

El problema que tiene por objetivo solucionar este proyecto es el de implementar un sistema que permita compartir una conexión a Internet entre múltiples usuarios de forma equitativa y segura, tanto para el usuario como para el prestador del servicio.

Desde el punto de vista del prestador del servicio soluciona los siguientes problemas:

- ☒ Posibilidad de identificar a cada uno de los usuarios que hacen uso del sistema.
- ☒ Posibilidad de cobrar el servicio vendiendo diferentes modalidades de acceso: Por velocidad de conexión, por tiempo y por consumo de datos.
- ☒ Posibilidad de automatizar todo el sistema mediante sistemas de pago online como paypal o mediante algún sistema de *vending* o pago por monedas como el de azkoyen[5]

Desde el punto de vista del usuario del servicio soluciona los siguientes problemas:

- ☒ Calidad del servicio: Un usuario siempre tendrá garantizado el ancho de banda que ha contratado y nunca podrá consumir más ancho de banda del que ha contratado.
- ☒ Seguridad de las comunicaciones: Todo el tráfico TCP/IP que envía y recibe el usuario está encriptado. Para ello se le ofrecen al usuario dos opciones de encriptación: Una pasarela VPN encriptada o encriptación WPA2-Enterprise (802.1x/PEAP[6]) en la capa de enlace de la red WiFi.
- ☒ Facilidad de uso: Todo el sistema es *Plug&Play*, el usuario se conecta al sistema y recibe conectividad automáticamente (DHCP) y al tratar de acceder a Internet encuentra una página con las instrucciones para conseguir el acceso deseado.

[5] http://www.azkoyenmediosdepago.com/productos/selectores
[6] http://en.wikipedia.org/wiki/Wi-Fi_Protected_Access

1.2. Motivación del proyecto: Alcance y objetivos

La principal motivación de este proyecto es la de proporcionar un sistema de acceso a Internet que incorpore todas las características deseables tanto para el usuario del mismo como para el prestador del servicio. El objetivo último es desarrollar una forma fiable y robusta de facilitar o bien vender el acceso a Internet en lugares públicos.

La iniciativa de este proyecto surge de la empresa Ubalda Ingeniería SL[7] con el objetivo de vender el sistema a empresas interesadas en el mismo o bien con el objetivo de vender el acceso a Internet a través del mismo.

1.3. Análisis de viabilidad

Cuando se plantea cualquier proyecto, uno de los primeros temas a tratar es la viabilidad: ¿es posible técnicamente llevarlo a cabo?, ¿merece económicamente la pena?

En proyectos de desarrollo como éste, en los que no se conoce a priori las técnicas que se utilizarán para alcanzar la solución, la única forma de responder con certeza a esas preguntas es analizando soluciones similares ya existentes.

Respecto a la cuestión técnica es indudable que es viable técnicamente ya que existen soluciones similares desarrolladas. De hecho, el presente proyecto se basa en uno anterior de Igalia (CANSAS) que se describe en el capitulo 2.

Respecto a la cuestión económica un par de buenos ejemplos son los siguientes:

- La distribución de Linux basada en firmware para dispositivos empotrados tales como routers inalámbricos DD-WRT[8]. Dicha distribución es gratuita en su versión estándar pero no en la versión “profesional” la cual activa características “extras”. Para la activación de la versión profesional hace falta comprar un código que cuesta aproximadamente 20€.

[7] http://www.fic.udc.es/ViewContent.do?location=dean.jobs&contentId=16892&categoryId=181&subcategoryId=181 http://ubalda.com/ofertatrabajo/
[8] http://www.dd-wrt.com/

- El proyecto FON[9] es una iniciativa que nace con el objetivo de crear una comunidad WiFi global. La compañía ofrece "routers" WiFi (inalámbricos) subvencionados, a bajo precio, para que el usuario comparta parte del ancho de banda de su conexión a Internet. Al mismo tiempo, este usuario obtiene el privilegio de conectarse gratuitamente a los routers WiFi de otros usuarios de la comunidad. A cambio de estas facilidades, la compañía se reserva el derecho de cobrar a los no usuarios que quieran conectarse a un punto de acceso compartido a través de PayPal. La mitad de las ganancias se comparten con el propietario del router inalámbrico y la compañía.

En este caso la vía de negocio que se pretende abrir consta de estas dos posibilidades:

- Cobrar al cliente por el hardware, la instalación y el mantenimiento necesario del sistema para que lo explote en su propio beneficio vendiendo las cuentas de acceso al sistema.

- Ofrecer la instalación, configuración y mantenimiento gratuitos del sistema con la condición de que los ingresos generados por la venta de las cuentas necesarias para el acceso a Internet van íntegramente (o parcialmente) destinados a la compañía (Ubalda en este caso).

Además la empresa Ubalda Ingeniería SL, está estudiando la posibilidad de convertirse en operador de Internet, de hecho ya ha solicitado permiso a la CMT (Comisión del Mercado de Telecomunicaciones) para la creación de un wISP[10], que dará servicios de Internet a través de la tecnología inalámbrica de nueva generación WiMAX a empresas alejadas de núcleos urbanos que carezcan de acceso a Internet de alta velocidad.

Por tanto, la empresa puede beneficiarse del presente proyecto para ofrecer servicio a Internet a través de WiFi a usuarios itinerantes en lugares donde no sería posible este acceso debido a su situación remota.

[9] http://www.fon.com/es
[10] Wireless ISP: Proveedor de servicios de Internet a través de tecnologías inalámbricas

Capitulo 2. Estado del arte

Los usuarios itinerantes ya son miembros típicos del entorno de trabajo actual. Ya sea una empresa que quiere diferenciar entre los accesos a su red interna de visitantes de los del personal propio, o usuarios privados accediendo a su proveedor de servicios de Internet, todos ellos comparten una necesidad común, un acceso sencillo y seguro a sus ordenadores y recursos en general. Según ha evolucionado la tecnología, los usuarios móviles han crecido en importancia. Este tipo de usuarios necesitan que los servicios que puedan utilizar estén disponibles desde cualquier lugar desde donde se puedan conectar al sistema asegurando, al tiempo, los recursos de la red contra accesos no autorizados.

Aparte de estas consideraciones desde el punto de vista de los usuarios, los administradores de sistemas tienen una serie de objetivos en la gestión de sus sistemas:

- Facilidad para garantizar un acceso seguro de los usuarios a la red protegiendo la privacidad de las comunicaciones.
- Gestión de frecuentes reorganizaciones y despliegue de nuevos servicios.
- Gestión, monitorización y registro de usuarios itinerantes.
- Autenticación de usuarios antes de acceder a los recursos y servicios de la red.
- Monitorización de accesos a la red, detectando usuarios no autorizados y equipos no autorizados.
- Crear áreas de acceso limitado para trabajadores temporales, vendedores, clientes, visitantes...
- Gestión de equipos utilizados por varios usuarios.

Muchas herramientas usadas actualmente por los administradores de sistemas para la gestión de usuarios solucionan sólo alguna de las cuestiones indicadas anteriormente debido a que estas herramientas se desarrollan teniendo en mente únicamente una parte del cuadro general de la empresa. Hoy por hoy, hay una gran cantidad de herramientas orientadas a la seguridad de las instalaciones, a examinar equipos, gestionar dispositivos de red o analizar tráfico de red donde esté disponible. La especialización de estas herramientas hace que sean capaces de tener una visión muy detallada de ciertos aspectos concretos de la actividad de un

segmento de usuarios particular; pero llevar esta información hasta una visión global de todos los usuarios finales suele ser una tarea del administrador. Todas estas herramientas se basan normalmente en asumir que los usuarios son inmóviles y que los usuarios siempre acceden desde los mismos dispositivos o desde el mismo punto de la red. El cambio en el paradigma de usuario pasando de usuario inmóvil a un usuario itinerante, hace que algunas de estar herramientas no sean válidas y los administradores comiencen a desarrollar sus propias aplicaciones para ciertos aspectos de la gestión; aplicaciones que, como sus predecesoras, no suelen poder relacionarse con el resto de las herramientas, imposibilitando tenerla mencionada visión global

2.1. Punto de partida: CANSAS

El sistema desarrollado se basa en el sistema CANSAS[11] (Ciphered and Authenticated Network Services Access System) desarrollado por Igalia en 2005.

El sistema CANSAS ofrece un sistema de acceso a Internet con control de usuarios y ancho de banda.

2.1.1. Características de CANSAS

- Manejo de los usuarios directamente a través de phpldapadmin (un gestor de bases de datos LDAP genérico escrito en PHP).
- Autenticación de cada conexión con el sistema a través de un servidor RADIUS
- Capacidad de restringir el acceso de los usuarios a determinados rangos de IPs.
- Autenticación basada en Web.
- Soporte de conexiones VPN (protocolo PPTP) para acceso encriptado al sistema.
- Control de ancho de banda por usuario.
- Basado en la distribución KNOPPIX GNU/Linux.
- Generación de logs de uso del sistema

[11] http://community.igalia.com/twiki/bin/view/Cansas/OldHome

2.1.2. Mejoras sobre CANSAS

Las nuevas características que incorpora el sistema estudiado respecto a CANSAS son las siguientes:

- → Distribución base y núcleo del sistema actualizado: CANSAS se basaba en la distribución Knoppix del 2004 con kernel 2.4.x que ha quedado completamente desfasada, mucho del hardware actual no es soportado por el antiguo sistema CANSAS. El sistema actual se basa en la distribución Ubuntu 8.04LTS e incorpora uno de las últimas versiones del núcleo de Linux (2.6.29.1 en concreto)

- → QoS (Calidad de Servicio): CANSAS solo permite control de ancho de banda pero no controla el número de conexiones TCP que un usuario puede crear, de forma que se podría saturar la conexión relativamente fácil con el uso de programas P2P por parte de los usuarios si el gestor de ancho de banda esta detrás de un router con memoria limitada que tiene que hacer NAT provocando la saturación del mismo.

- → LAN Isolation: En el sistema estudiado se ha implementado un firewall en *layer2*[12] que impide que los usuarios conectados al sistema se comuniquen entre si.

- → Firewall avanzado: Gracias al uso de layer7[13] el sistema actual puede clasificar, priorizar y filtrar las conexiones según el tipo (p2p,voip,http,ftp…) sin importar los puertos de origen y destino.

- → Encriptación WPA de la red WiFi: Se ha implementado encriptación WPA Empresarial (WPA-PEAP) integrada completamente en el sistema.

- → Control del consumo de datos: CANSAS no permitía saber cuanto tráfico había consumido cada usuario.

- → Dos modalidades de acceso: Por tiempo o por consumo. CANSAS solo permitía acceso por tiempo.

[12] http://ebtables.sourceforge.net/
[13] http://l7-filter.sourceforge.net/

- → Automatización del sistema: Se permite dar de alta a los usuarios de forma automática mediante paypal o un sistema de control de monedas conectado a una impresora de tickets.

- → Interfaz de administración: Se ha rediseñado completamente toda la Interfaz Web de administración del sistema haciéndola mas amigable e intuitiva. Ahora toda la funcionalidad del sistema es gestionada desde la interfaz de administración. En CANSAS tenias que acceder directamente a la base de datos a través del gestor *phpmyldapadmin* para gestionar los usuarios y las otras partes del sistema eran gestionadas a través de gestores genéricos como *Webmin*[14].

- → Posibilidad de definir sitios de libre acceso: tales como google o PayPal. Puede ser interesante que los usuarios no autentificados puedan navegar por la Web de la empresa o del prestador del servicio.

2.2. Soluciones existentes.

Vamos ahora a hacer un análisis de las principales soluciones existentes hasta la fecha y luego analizaremos detenidamente las herramientas que proporciona Linux para el control del tráfico, herramientas que utiliza el presente proyecto para construir sobre ellas una solución global.

2.2.1. pfSense

pfSense es una versión de la distribución de UNIX FreeBSD configurada especialmente para ser usada como firewall y router. pfSense es un desarrollo muy interesante, a pesar de ser gratuito y OpenSource incluye muchas características avanzadas que solo están disponibles en las soluciones comerciales de más alto nivel:

[14] http://www.webmin.com/

- Firewall
  - Filtrado por IP/MAC y tipo de tráfico.
  - Capacidad de limitar el número de conexiones activas
  - Permite identificar el Sistema Operativo que intenta acceder a la red gracias a un avanzado sistema de reconocimiento de OS *fingerprinting*[15]
  - Registro configurable de conexiones
  - Capacidad de actuar como un firewall Ethernet (en la capa 2 de enlace)
  - Protección contra ataques tipo flood y DoS (Denial of Service)
- Router
  - Modificación activa de la tabla de estados y de conexiones establecidas.
  - Capacidad de limitar el número de conexiones por tipo, destino, origen y por tiempo.
  - Traducción de direcciones de red (NAT)
  - Redundancia de enlace: Permite configurar varias interfaces de red y tiene la capacidad de detectar cuando falla una para configurar automáticamente la otra como backup.
  - Balanceo de carga: Permite usar múltiples interfaces de red y balancea el ancho de banda disponible de forma transparente.
- Seguridad
  - Capacidad de crear enlaces PPP tipo VPN del tipo IPSec, OpenVPN y PPTP.
  - Servidor de conexiones punto a punto sobre Ethernet (PPPoE)
  - Acceso de los usuarios a través de servidor RADIUS
  - Portal cautivo con conexión encriptada mediante SSL.
- Plug&Play
  - Permite configurar servicios DNS para IPs dinámicas
  - Servidor DHCP
  - Generación de gráficos en tiempo real sobre el uso de los recursos de red y del sistema.

[15] El OS Fingerprinting es una técnica que consiste en analizar las huellas que deja un sistema operativo en sus conexiones de red. Está basada en los tiempos de respuesta a los diferentes paquetes, al establecer una conexión en el protocolo TCP/IP, que utilizan los diferentes sistemas operativos. http://lcamtuf.coredump.cx/p0f.shtml

2.2.2. NoCatAuth

NoCatAuth es un software escrito en Perl[16] que permite autenticar el acceso a una red con un portal cautivo. Se usa un portal cautivo para controlar los accesos a redes del tipo 802.11. Está compuesto por un gateway y un servidor de autenticación. El servidor de autenticación muestra al usuario la pantalla de autenticación, busca en la base de datos las credenciales del usuario, notifica de forma segura al gateway el estado del usuario y lo autoriza para posteriores accesos. Por otro lado, el gateway, gestiona las conexiones locales, establece los parámetros de ancho de banda y las reglas del firewall y desconecta a los usuarios inactivos después de un cierto tiempo. Este software está liberado bajo GPL.

2.2.3. WiFiDog

WiFiDog es un sistema de acceso a Internet del tipo portal cautivo[17], ha sido diseñado como un reemplazo de NoCatAuth y podemos considerarlo como uno de los desarrollos más interesantes del tipo OpenSource. Consta de dos partes:

- Servidor de autenticación: Consiste en un servidor que almacena la base de datos de usuarios y se encarga de autenticar a estos cuando alguien solicita acceso a la red. Esta desarrollado en PHP
- Gateway: Consiste en un software desarrollado en C y diseñado para que pueda funcionar en dispositivos embebidos, funciona en cualquier router capaz de ejecutar OpenWRT[18].

2.2.4. ChilliSpot

ChilliSpot es el portal cautivo OpenSource de referencia. Soporta autenticación de usuarios vía Web y es el desarrollo más extendido en los HotSpot[19] públicos. Los firmwares para puntos de acceso inalámbricos más extendidos (OpenWRT, DD-WRT) lo incluyen como portal cautivo por defecto y es el software que ha elegido también el proyecto FON[20] para desarrollar su sistema de acceso.

---

[16] Para sistemas embebidos se ha implementado una versión reducida denominada NoCatAuthSplash escrita en C

[17] Un portal cautivo (o captivo) es un programa o máquina de una red informática que vigila el tráfico HTTP y fuerza a los usuarios a pasar por una página especial si quieren navegar por Internet de forma normal. http://es.wikipedia.org/wiki/Portal_cautivo

[18] OpenWRT es una distribución de Linux basada en firmware usada para dispositivos empotrados tales como routers personales.

[19] Un hotspot (en inglés 'punto caliente') es una zona de cobertura Wi-Fi en lugares públicos.

[20] http://www.fon.com/

2.2.5. Cisco User Registrarion Tool (URT)

Cisco URT[21] es un servicio de autentificación de usuarios en una red de área local que identifica y asigna los usuarios a diferentes VLAN[22] . Cada usuario es asociado a unos servicios y recursos específicos de la red a través de una asignación dinámica a una VLAN. Los usuarios podrán registrarse en la red a través de un sistema Web desde sus clientes Windows, Macintosh o GNU/Linux. El sistema estará integrado con el sistema de autenticación LDAP[23] existiendo un enlace seguro entre el cliente y el *VLAN Police Server (VPS)*. Podrá existir autentificación a nivel de MAC, de este modo se controlarán los accesos de los usuarios sólo desde aquellas máquinas autorizadas

Los componentes de la arquitectura presentada por Cisco son:

- ***URT Administrator Server:*** Es el componente principal de la solución Cisco Secure . Se le envía toda la información de registro de usuarios y de acceso desde todos los VPS que están bajo su control.
- ***URT Management Interface:*** Asigna las VLAN a cada usuario, grupos de usuarios, organizaciones o direcciones MAC y establece las asociaciones a los dominios Windows NT, AD, NDS y RADIUS[24].
- ***URT VPS:*** Es el responsable de asignar cada puerto del switch del cliente en base al registro del nombre de usuario, del nombre del grupo, de la unidad organizativa o de la dirección MAC de usuario. También se usa en la autenticación y asignación de a los usuarios vía Web.
- ***URT Client Module:*** El módulo cliente es automáticamente instalado en la máquina habitual del cliente para permitir la autenticación de cada usuario.
- ***Web Client Module:*** El módulo del cliente web se llama directamente desde el navegador del cliente y proporciona la misma funcionalidad que el URT Client Module. Se trata de un applet que se descargará como un archivo Java. Esta aplicación servirá para que el usuario pueda registrarse mediante su identificador y su contraseña.

[21] http://www.cisco.com/en/US/products/sw/secursw/ps2136/index.html
[22] Virtual Local Area Network (red virtual de área local)
[23] LDAP (Lightweight Directory Access Protocol, Protocolo Ligero de Acceso a Directorios)
[24] Remote Authentication Dial-In User Server

Con un registro de los usuarios vía desde sus clientes Windows, Macintosh o, el usuario puede registrarse de forma segura al VPS sin necesidad de descargarse ninguna aplicación. Cisco Secure proporciona autenticación a través del servidor estándar RADIUS como un control de acceso seguro.

Los aspectos más destacables en el URT de Cisco son:

- **Interfaz de registro web en el cliente.** El sistema URT soporta autenticación vía Web para diferentes plataformas de clientes. Cuando los usuarios lanzan el navegador, son directamente redirigidos a la página de registro hasta que se autentifican correctamente y se le asigna su correspondiente VLAN. Los usuarios podrán elegir si se autentifican en algún dominio de los soportados por la herramienta de administración del URT.
- **Opción de seguridad basada en MAC.** La seguridad basada en MAC proporciona protección en el acceso de usuarios a VLAN desde máquinas no registradas. Es especialmente importante para los consumidores que no desean exponer los registros a VLAN a direcciones no reconocidas.
- **Autenticación RADIUS.** El registro vía Web proporciona autenticación RADIUS.
- **Enlace seguro entre el cliente y el servidor VPS.** Puede establecerse una autenticación segura y cifrando los datos del usuario. De este modo se consigue una protección en el acceso de los usuarios ante posibles ataques al sistema.
- **Soporte LDAP.** A través del *URT Administrator Server* los usuarios, grupos y organizaciones pueden ser recuperados usando LDAP.
- **Más de un usuario por puerto.** En las versiones anteriores de sólo se permitía un usuario por puerto. Si se detectaba más de una dirección MAC por puerto, el puerto se eliminaba de esa VLAN. La última versión permite a múltiples usuarios conectados a un hub acceder por un único puerto del *switch* asignado a la VLAN. El sistema URT permite al administrador asignar un único usuario por puerto o múltiples usuarios por puerto basado en este caso, en el registro de ID y del password.
- **Muestra de los grupos bajo Windows.** El interfaz del sistema del Administrador de URT permite la visualización y la gestión de usuarios pertenecientes a un grupo *Windows NT*. Para los usuarios de un grupo seleccionado, el administrador puede seleccionar el grupo, obtener una lista de usuarios para ese grupo y luego asignar una VLAN para ese grupo o para un usuario dentro de ese grupo.

2.2.6. Cisco Building Broadband Service Manager (BBSM)

Cisco BBSM es una puerta de enlace para el acceso a redes públicas que permite un acceso públicas simple y "*plug-and-play*", múltiples sistemas de autenticación y gestión, generación de informes y configuración basada en web.

Sus principales características son:

- **Acceso.** Cisco BBSM[25] permite el acceso de usuarios sin importar ni configurar los parámetros de red. Los usuarios pueden conectarse y automáticamente reciben la configuración por DHCP[26]. También se pueden configurar direcciones IP estáticas, así como configuraciones de *proxy* estáticas o entradas de DNS. Esta capacidad de *plug-and-play* permite el acceso de una amplia variedad de usuarios y de configuraciones sin requerir ningún tipo de soporte.
- **Flexibilidad y supervisión.** Cisco BBSM tiene características integradas que permiten una gestión efectiva y monitorización de las implementaciones, independientemente de su tamaño. Con su sistema basado en conjuntos de características gestionables, el Cisco se integra en la arquitectura de la empresa existente en sus métodos de supervisión.
- **Portal**. Cisco BBSM redirige a todos los usuarios con dos pasos durante la conexión. Primero, se dirigen a una pantalla de conexión (*Connection Screen*), que se puede modificar según los requisitos particulares para explicarles a los usuarios los servicios disponibles. Esta pantalla también incluye áreas de libre acceso. Con esta funcionalidad, se permite a los usuarios visitar algunos dominios especificados (antes de la autenticación), tales como sitios locales, sitos de información o sitios de comercio electrónico. Una vez que los usuarios seleccionen y compren un servicio concreto de conexión (especificado en la *Connection Screen*), Cisco BBSM los redirige a una dirección configurada como su primera conexión a Internet.

2.2.7. Authenticated Network Access.

ANA es un sistema de autenticación de red desarrollado en la Universidad de Utah que proporciona acceso general a la red en clases,

[25] http://www.cisco.com/en/US/products/sw/netmgtsw/ps533/
[26] Dynamic Host Configuration Protocol

áreas comunes, laboratorios y apartamentos de estudiantes siempre y cuando se disponga de un identificador universitario (uNID) y un password. El uso de está controlado por una serie de políticas.

Las características más importantes de son:

- **Autenticación**. La solución autentica un par login/password independientemente de la dirección MAC del cliente.
- **Rendimiento**. Posibilidad de crear conexiones concretas con un alto ancho de banda para permitir grandes transferencias (aplicaciones multimedia).
- **Funcionalidad**. La funcionalidad de los portátiles no está afectada por la metodología.
- **Escalabilidad**. Permite la creación de un sistema escalable mediante la definición de múltiples redes.
- **Modelo Global de Autenticación** Permite la autenticación utilizando LDAP, plugins...
- **Configuración mínima del cliente**. Permite realizar conexiones sin requerir software específico en el cliente utilizando únicamente DHCP.
- **Multiplataforma**. Soporta múltiples plataformas con una sencilla configuración web.

2.2.8. FirstSpot

FirstSpot es una aplicación basada en Windows que funciona como portal cautivo. Un portal cautivo es un conjunto de webs que saltan en cualquier petición web que hagan sus clientes, en el caso de FirstSpot, para identificar y tener el control sobre los usuarios. Esto lo consigue haciendo pasar todo el tráfico por el ordenador en el que está instalado, de esta manera se crean dos redes de ordenadores distintas, una que va desde un dispositivo de red del ordenador de FirstSpot hacia la conexión a Internet «Red Pública (Public Network)» y la otra que va de otro dispositivo de red distinto del ordenador de FirstSpot hacia los usuarios «Red Privada (Private Network)».

Las principales características de FirstSpot son:

Amigable para con el usuario

- Autenticación basada en web, por lo que no es necesaria la instalación de software extra en el cliente.

- Obtención de cuentas personalizables con tiempo de conexión limitado.
- Pantalla de estado de los parámetros de conexión del usuario (tiempo restante de conexión, botón de desconexión...).

Gestión de red

- Diferentes métodos de autenticación (shadow passwords y RADIUS)
- Limitación del ancho de banda por usuario.
- MACs confiables: se permite la conexión a una lista de MACs sin necesidad de autenticación.
- Aislamiento del cliente.

Sistemas de pago

- Soporte de pago del tiempo limitado de conexión a priori o a posteriori.
- Soporte de cobro por tasa de datos transferida.
- Soporte de PayPal.
- Sistema de creación de facturas a partir de los logs de conexión.

Configuración del sistema de login

- Página de login configurable e integrable en otros sites.
- Posibilidad de diferentes páginas de login para diferentes segmentos de red.

Otras características

- Capacidad para permitir libre acceso a determinadas direcciones.
- Funciona tanto en redes WiFi como en redes Ethernet.

2.2.9. WARTA

WARTA, (Wireless Authentication, Routing, Traffic control and Accounting) es un proyecto que pretende proporcionar acceso wireless a Internet de una forma barata, controlada y autenticada. Para ello han integrado, principalmente, tres «piezas» de software: FreeBSD, PPPoE e IPFW.

Se elige PPPoE debido a que se trata esencialmente de PPP (*Point to Point Protocol*) con la mejora de poder ser utilizado sobre cualquier medio Ethernet, incluyendo el 802.11x wireless. Como toda implementación de PPP tiene autenticación (entre otras) y, en la práctica, funciona creando túneles que separan todos los cliente en interfaces virtuales individuales de nivel 2 (por lo que todo el tráfico pasa a través del *gateway PPP*). Todo el tráfico pasa a través del IPFW, un firewall que puede filtrar por dirección MAC, interfaces virtuales o no y direcciones IP. Finalmente, PPP utiliza RADIUS para la gestión de usuarios.

Capitulo 3. Fundamentos Teóricos y Tecnológicos

En este capitulo se van a describir e introducir las diferentes tecnologías que han sido empleadas para el desarrollo del presente proyecto y se introducirán los conceptos teóricos necesarios. Empezaremos hablando sobre el *Open Source*. Se realizará también un análisis de los diferentes algoritmos de moldeado de tráfico disponibles en Linux gracias a los cuales el presente proyecto consigue controlar el ancho de banda.

3.1. Open Source

A lo largo de la historia de Internet el movimiento open source es uno de los fenómenos más espectaculares que se han producido. Son muchos los juicios, mitos y afirmaciones que circulan entorno al open source a lo largo y ancho de la sociedad de la información. Sin embargo, muchos de estos aspectos son extremadamente superficiales o deben ser matizados.

Sin ninguna duda, la opinión más extendida acerca del open source es que está formado por aplicaciones que no suponen coste económico alguno para el usuario. Aunque en muchos casos las aplicaciones asociadas a este movimiento son gratuitas, esta característica no es una cualidad imprescindible para incluir una aplicación dentro del marco del open source. Una definición que puede servir como punto de partida para comenzar a comprender este fenómeno es la siguiente:

> *"El open source es el movimiento que engloba el software que se desarrolla públicamente y que se encuentra disponible en forma de código fuente nativo."*

Esta breve definición [J1 CAR] pone de manifiesto dos de los pilares fundamentales sobre los cuales se asienta el *open source*. Sin embargo, las consecuencias del desarrollo público y la disponibilidad del código fuente tienen más calado del que se puede apreciar a simple vista.
En primer lugar, el desarrollo público implica la colaboración de agentes externos en la fabricación del producto software, por ejemplo otros programadores o los propios usuarios. Esto influirá de forma trascendental en los mecanismos de desarrollo y la calidad del producto.

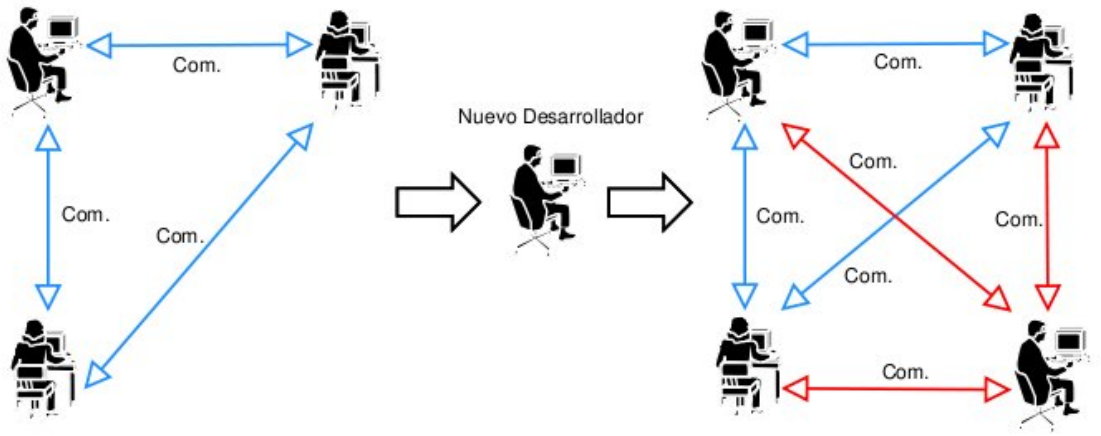
En segundo lugar, el hecho de que el código fuente no se encuentre solamente al alcance de los desarrolladores provoca un salto cualitativo muy importante en el ciclo de vida del software, alterando drásticamente las formas clásicas de mantenimiento y evolución del producto.

Los dos factores anteriores permiten el desarrollo, depuración y mejora del software a velocidades vertiginosas porque se genera una forma de colaboración útil y activa entre los usuarios, desarrolladores y colaboradores. Esta forma de trabajo extiende la filosofía del open source mucho más allá del mero hecho de entregar el código fuente al usuario, enriqueciendo este movimiento con metodologías, mecanismos y reglas de carácter técnico y social.

Los mecanismos de desarrollo de la comunidad open source están desafiando los pensamientos tradicionales acerca de lo que se consideraba como entorno apropiado para la construcción de software exitoso y de calidad.

3.1.1. Ley de Brooks

Hasta el año de la aparición del sistema operativo GNU/Linux (1991), el desarrollo del software parecía gobernado por la Ley de Brooks[27]. Esta ley afirma que según el número de desarrolladores, n, se incrementa, la complejidad del proyecto aumenta n^2. El razonamiento que conduce a esta afirmación se basa en que el número de vías de comunicación entre los desarrolladores se debe incrementar en el número total de los mismos cada vez que se incorpora un nuevo componente al equipo de trabajo, siendo en consecuencia mucho más complicado combinar el trabajo de todos ellos.



[27] http://en.wikipedia.org/wiki/Brooks%27s_law

"Añadir personal a un proyecto
retrasado lo retrasará aún más"

De este modo, un proyecto que involucre demasiados desarrolladores será demasiado complejo para completar a tarea eficiente y efectivamente. Muchos directores de proyecto experimentados han comprobado con el paso de los años que la Ley de Brooks era innegable y han aprendido a base de prueba y error que añadir más desarrolladores a un proyecto retrasado, en lugar de reducir los plazos, a menudo dilata aún más su finalización.

El desarrollo del sistema operativo GNU/Linux demostró que la Ley de Brooks no era inexorable, ya que más de 10.000 desarrolladores contribuyeron activamente a la construcción de este software. A partir del éxito de GNU/Linux, muchos de los proyectos desarrollados bajo el amparo del movimiento open source han desafiado a las prácticas tradicionales de programación, demostrando en cada paso que era posible romper la barrera modelada en la Ley de Brooks.

La red Internet es la principal causante de la nulidad de la Ley de Brooks en el contexto del open source. Gracias a que red Internet pone al alcance de los miembros de un proyecto herramientas que facilitan y agilizan extraordinariamente la comunicación, sucede que los procesos de comunicación n^2 descritos por la Ley de Brooks se llevan a cabo sólo con esfuerzo n. [J1 CAR]

La creación de programas open source lleva implícito el cumplimiento de una serie de principios básicos que gobiernan el desarrollo de los mismos. Estas reglas son la base fundamental que ha conducido a proyectos como GNU/Linux o Apache al éxito.

Estos principios básicos, descritos por la mayoría de los autores expertos en el movimiento open source, por ejemplo en [J1 CAR] y [K1 RAY], pueden agruparse en tres bloques distintos: desarrollo evolutivo, comunidad abierta de desarrolladores y uso de la red Internet.

3.1.2. Desarrollo Evolutivo

El desarrollo evolutivo o iterativo quizás sea una de las características más significativas del open source. Durante el desarrollo de aplicaciones open source se imita el proceso de evolución biológica típico

de la naturaleza. En el mundo natural, la evolución biológica permite la adaptación de las especies a los cambios que se producen en el entorno. Dentro del mundo del software el entorno lo conforma el área de la industria en el que reside una aplicación. Tanto en el sentido biológico como en el informático, la adaptación permite la supervivencia en entornos que se encuentran en permanente cambio. Sin embargo, el mundo de la informática se mueve a una velocidad vertiginosa y un año en esta disciplina equivale a cientos de miles de años de evolución biológica.

Para ilustrar la importancia de la evolución en el software veremos un ejemplo del mundo natural descrito en [J1 CAR]. Imaginemos que un oso de pelo corto que sobrevive en un clima con una temperatura confortable decide invernar durante 500.000 años. Supongamos también que cuando el oso despierta de su letargo el entorno donde habitaba ha cambiado y ahora es un clima frío y cubierto de nieve. Los osos que durante esos 500.000 años permanecieron activos y fueron evolucionando hacia el pelo largo, ayudados por la presión continua aplicada por el cambio gradual del entorno, podrán protegerse del frío, mientras que nuestro oso de pelo corto está avocado a la extinción.

El software cerrado se comporta de la misma forma que el oso de pelo corto. Dicho software desaparece durante meses y a veces incluso años para ser desarrollado de forma secreta. Cuando reaparece nuevamente tiene que ser probado dentro del entorno actual y este puede haber variado bruscamente. El éxito en la supervivencia del software propietario en estos casos depende fundamentalmente de dos factores: la habilidad de los desarrolladores para predecir el entorno emergente y la calidad del software en dicho entorno emergente. En algunos casos, las aplicaciones pueden sobrevivir sin cumplir ninguno de los dos requisitos anteriores si la competencia es baja. Esto puede hacernos comprender el sentido de las agresivas estrategias monopolistas aplicadas por algunas de las grandes multinacionales del software propietario. Si la competencia es baja, el producto sobrevive aunque se adapte mal a las necesidades del usuario o aunque su calidad sea baja.

La supervivencia del software no es el único argumento a favor del desarrollo evolutivo de aplicaciones. Otra razón poderosa a favor de este principio reside en el hecho de que en muchos casos para comprender completamente un problema es necesario implementar una solución errónea. Esto quiere decir que las soluciones mejores e innovadoras a menudo aparecen cuando los desarrolladores detectan que el concepto del problema que tenían en mente era erróneo.

Al contrario de lo que pueda parecer a primera vista, desarrollar una solución errónea porque no se comprende correctamente el problema a resolver solamente es una situación crítica si se detecta en fases muy avanzadas del proceso. Sin embargo, si la detección es temprana el impacto sobre el proyecto es muy reducido.
Para lograr reducir este impacto, en la práctica es imprescindible realizar entregas del producto muy frecuentes, así como disponer de una comunidad abierta de desarrolladores. De este modo, el open source se refina, corrige y actualiza constantemente, logrando niveles de adaptación que el software propietario no puede alcanzar.

3.1.3. Comunidad abierta de desarrolladores

El open source logra imitar los procesos biológicos evolutivos descritos en el apartado anterior mediante una presión constante hacia la selección. El carácter público y abierto de estas aplicaciones permite que los usuarios y desarrolladores encuentren disponibles constantemente nuevas versiones de las mismas.

Dentro de la filosofía del open source cualquier persona del mundo puede examinar el código fuente de una aplicación y enviar correcciones y sugerencias[28]. Dentro del software propietario existe una clara línea que separa a los desarrolladores de los usuarios y no puede existir la figura del desarrollador colaborador porque el código fuente es secreto. En el mundo del open source esta línea es borrosa, siendo habitual que grupos de personas no directamente relacionadas con el proyecto contribuyan en partes fundamentales del mismo. Este tipo de desarrollo abierto y democrático proporciona una presión constante y continua sobre la comunidad open source hacia la selección del mejor código. Por contra, el paradigma del software propietario es totalmente opuesto, siendo una práctica habitual que el producto permanezca oculto durante el tiempo en el cual es desarrollado.

La colaboración es posible porque el equipo de desarrollo responsable del proyecto escucha con interés real las opiniones y aportaciones de los usuarios y desarrolladores colaboradores, siendo esta la ruta más rápida hacia la mejora y depuración efectiva de las aplicaciones. Con esta conducta los proyectos open source logran el apoyo de programadores con talento esparcidos por todo el mundo que, mediante su

[28] La correcciones a fallos son conocidas con el término patch (parche) dentro de la comunidad open source.

colaboración, logran infundir un sentimiento de propiedad global sobre el proyecto. Tratar a los colaboradores como si fuesen el recurso más valioso del proyecto hará que estos respondan transformándose realmente en el recurso más valioso.

Para evitar los posibles conflictos que puedan surgir entre colaboradores bajo esta filosofía, los miembros de la comunidad aplican un principio técnico que permite solventar este tipo de problemas: ([D2 PAV])

"Dejad que el mejor código gane."

Bajo este lema no importa quién envíe la solución para un problema dado, lo realmente importante es comprobar si es la mejor solución. Principios como este favorecen una competición sana entre desarrolladores y aceleran extraordinariamente los avances tecnológicos.

Mediante la implicación de una comunidad abierta de desarrolladores y usuarios en todo el ciclo de fabricación es posible lograr que las desviaciones sobre los objetivos sean corregidos antes que la aplicación se encuentre demasiado avanzada. Así mismo, disponer de una base de beta-testers y co-desarrolladores lo suficientemente grande logrará que cada problema o desviación sea caracterizado rápidamente y que la solución sea obvia para alguien[29]. En la mayoría de las aplicaciones open source lo más importante, aparte de tener buenas ideas, es reorganizar las buenas ideas de los usuarios y colaboradores. De hecho, en muchas ocasiones lo último es incluso mejor.

3.1.4. Uso de la red Internet

Las cuestiones analizadas en los dos apartados anteriores serían muy difíciles de llevar a cabo si la comunicación entre los miembros componentes de un proyecto (colaboradores y usuarios) no es fluida y eficaz. En la actualidad, la red Internet es el medio utilizado por la comunidad open source para lograr estos objetivos. Sin ninguna duda, el desarrollo de Internet ha tenido mucho que ver con la gran popularidad que ha alcanzado el movimiento open source durante los últimos años. A diferencia de los comienzos de la era informática, ahora el gran público

[29] En honor a Linus Torvalds, a este principio se lo conoce como la Ley de Linus porque es la base del desarrollo del sistema operativo GNU/Linux ([K1 RAY]).

dispone de un medio rápido y sencillo para acceder a los proyectos open source. Adicionalmente, Internet ha permitido que la comunicación entre colaboradores, en muchos casos separados por miles de kilómetros, sea más directa y efectiva que nunca a lo largo de la historia.

Los miembros de la comunidad open source no sólo han hecho uso de la red Internet, también han contribuido activamente en su propio desarrollo y evolución. Con el fin de facilitar los procesos asociados al desarrollo open source, la comunidad ha construido multitud de protocolos, lenguajes de programación, herramientas y servicios que han enriquecido enormemente las capacidades de la red Internet. Muchas de estas contribuciones ya forman parte del día a día de los usuarios de la red, incluso de aquellos que ni siquiera tienen conocimiento de la existencia del movimiento open source. Podríamos decir que el movimiento open source y la red Internet se han nutrido mutuamente de las potencialidades del otro, dando lugar a una simbiosis enormemente beneficiosa para ambas partes.

Este entorno de comunicación global, rápido y efectivo hace que el contexto de desarrollo de las aplicaciones open source sea muy distinto al de las aplicaciones propietarias. Una forma de aproximarse a la naturaleza real de tal contexto es utilizar la metáfora de la catedral y el mercadillo descrita en [K1 RAY]:

A lo largo de la historia, las catedrales siempre han sido construidas por un conjunto reducido de artesanos en completo aislamiento, de tal modo que el usuario no tenía acceso al producto final hasta que se encontraba finalizado. El desarrollo del software propietario se comporta en gran medida de este modo, siendo casi imposible que los usuarios puedan intervenir de forma activa y útil en el desarrollo del producto antes su finalización.

Por contra, el contexto de desarrollo de las aplicaciones open source es muy similar al que se puede encontrar durante las compras en un mercadillo. Un mercadillo se compone de vendedores y clientes muy diversos y heterogéneos. En los mercadillos, los vendedores logran el éxito cuando consiguen comercializar sus productos con el beneficio más alto posible. Por contra, los compradores fijan su objetivo en lograr adquirir buenos productos a bajo coste. Los mercadillos funcionan porque se favorece la competición entre los propios vendedores y entre los propios compradores. Los vendedores con precios elevados y baja calidad no obtienen beneficios. De forma análoga, los compradores *“tacaños”* no consiguen realizar ninguna compra.

Al igual que en un mercadillo, el contexto que rodea al software open source es altamente competitivo gracias a la fácil y ágil comunicación propiciada por la red Internet. Si trasladamos la metáfora del mercadillo al open source, los vendedores jugarían el papel de los desarrolladores y los clientes el de los usuarios, algunos de los cuales también son desarrolladores de software. De forma similar a los productos de un mercadillo, las buenas aplicaciones open source continúan adelante, mientras que las malas son corregidas o eliminadas.

Adicionalmente, la facilidad para intercambiar datos a través de Internet permite que el software se entregue a la comunidad frecuentemente, logrando de este modo un testeo continuo muy efectivo. Habitualmente, los desarrolladores publican el código fuente, la documentación y los bugs[30] en un repositorio central al que cualquier persona puede acceder desde cualquier parte del mundo.

Toda la comunicación entre usuarios y desarrolladores es pública y abierta, de tal modo que cuando una persona desea realizar una pregunta puede dirigirse al repositorio para comprobar si ya había sido respondida en otra ocasión. Tratar la información mediante mecanismos como este permite eliminar los esfuerzos duplicados. La capacidad de la red Internet para evitar los esfuerzos de comunicación innecesarios hace posible la ruptura de la Ley de Brooks. Si el coordinador de un proyecto sabe manejar las herramientas de este tipo que Internet pone a su alcance, y es capaz liderar sin coaccionar, varias cabezas son inevitablemente mejores que una, en contra de lo postulado por la Ley de Brooks.

3.1.5. Estrategias de negocio

Actualmente, el open source es una estrategia comercial que se está adoptando en pequeñas y grandes empresas. Multinacionales como Sun Microsystems, IBM o SAP han invertido parte de sus recursos dentro del open source, utilizando en muchos casos una estrategia híbrida junto con modelos propietarios.

[30] Término utilizado para describir los fallos en los programas. Su origen proviene de la época de los primeros ordenadores, máquinas que ocupaban un edificio completo y funcionaban mediante válvulas de vacío. En estos equipos era frecuente que las cucarachas (bugs) se posaran sobre los contactos de las válvulas de vacío, provocando fallos en los equipos.

Las compañías dedicadas exclusivamente al open source, por ejemplo Red Hat, Caldera o SuSE han tenido un gran éxito inicial, obteniendo valoraciones significativas dentro de la bolsa. También son numerosas las pequeñas empresas que han florecido proporcionando soluciones empresariales basadas en open source.

Este nuevo entorno está provocando un salto en la forma de obtener beneficios por parte de las compañías dedicadas a la fabricación de sistemas informáticos. En lugar de comercializar licencias de software, las compañías open source comercializan el soporte y las soluciones personalizadas.

Dicho de otro modo, la industria está evolucionando desde la venta de bits hacia la venta de servicios que hacen que esos bits funcionen.

En este ámbito el estudio más pormenorizado hasta el momento lo ha realizado [K1 RAY], aunque es posible encontrar alguna información de carácter más genérico sobre el tema en [J1 CAR], [K63 SAN] y [D2 PAV]. En los siguientes apartados se resumirán y agruparán las ideas que los autores más relevantes tienen sobre las estrategias de negocio open source.

3.1.5.1. Valor de venta y valor de uso

Los programas de ordenador, al igual que otros bienes, poseen dos tipos distintos de valor económico: el valor de uso y el valor de venta.

- El valor de uso de una aplicación lo conforma su valor como herramienta, es decir, como multiplicador de la productividad.
- El valor de venta está asociado a su comercialización como mercancía. Si utilizásemos la jerga de los economistas, podríamos decir que el valor de venta se asocia a un bien final y el valor de uso a un bien intermedio.

La mayoría de las personas asumen que el valor del software se rige por un modelo de fabricación tradicional. Este modelo se apoya en dos premisas:

1. El valor de venta del software debe cubrir la mayor parte del salario de los desarrolladores de software.
2. El valor de venta del software debe ser proporcional a su coste de desarrollo y a su valor de uso.

Dicho con otras palabras, la mayoría de las personas tienden a asumir que el software tiene las mismas características, en cuanto a su valor, que cualquier otro bien fabricado. Sin embargo, es fácil demostrar que estas premisas son falsas.

Respecto a la primera afirmación, es necesario apuntar que el software desarrollado para ser comercializado es sólo una pequeña parte de todo el software producido en el mundo. De hecho la gran mayoría del software desarrollado en el mundo se fabrica internamente dentro una organización y no es comercializado. En la mayoría de los casos, este software desarrollado a media no se comercializa por que sus funcionalidades son demasiado específicas y se encuentran demasiado vinculadas a la organización. Con estos datos en la mano es posible afirmar que el valor de venta del software no tiene necesariamente que financiar los salarios de los desarrolladores, ya que todas las organizaciones que desarrollan software para su uso interno retornan este coste por otras vías. Por tanto, la primera premisa expuesta anteriormente podría considerarse una estrategia utilizada por algunas de las empresas fabricantes de software propietario para justificar los elevados precios del software.

La segunda afirmación es fácilmente cuestionable si se tiene en cuenta que el mantenimiento de un producto software supone el 75 % del coste de su ciclo de vida [D29 BOE]. Si el valor de venta del producto fuese proporcional a su coste de fabricación, esto supondría que el precio en el momento de la compra sería el 25 % del coste y el precio del servicio post-venta, o mantenimiento, sería el 75 %. En la realidad esto no sucede, ya que generalmente la compra de un producto propietario requiere una fuerte inversión inicial, mientras que los servicios de soporte en muchos casos tienen coste cero y en los que supone algún coste nunca se mueve en los porcentajes descritos anteriormente. Esto hace que los fabricantes de software propietario aumenten sus esfuerzos en la venta inicial y no en el servicio de soporte posterior, hecho que sin duda supone perjuicios evidentes a sus clientes.

La última parte de la segunda premisa, la cual indica que el valor de venta también es proporcional al valor de uso, es igualmente cuestionable. El valor de uso de una aplicación informática está ligada directamente a su funcionalidad actual y su capacidad de continuidad futura. Si el fabricante de un producto cesa su actividad, o el producto no se continúa desarrollando, su valor de uso cae drásticamente y termina siendo cero. Es un hecho empíricamente comprobable que el software pierde valor de uso con el paso del tiempo, ya que sus funcionalidades son mejoradas por nuevas versiones del mismo producto o por aplicaciones equivalentes.

Con los argumentos esgrimidos en este apartado es fácil concluir que la industria del software no puede comportarse en función de un modelo de fabricación tradicional. Ha de cambiar drásticamente su base de negocio para que el producto también sea satisfactorio para el usuario final. En caso contrario, la gran expansión de aplicaciones open source equivalentes a sistemas propietarios finalmente terminará por ocupar el mercado de estos últimos.

3.1.5.1.1. Modelos de negocio sobre el valor de uso

La idea fundamental sobre la cual se asientan estos modelos de negocio reside en que es necesario desplazar el valor de venta del software hacia el valor de uso. En estos modelos subyace la idea de que se pueden obtener multitud beneficios indirectos utilizando y desarrollando software open source.

A continuación se describirán dos modelos que pretenden este objetivo. El primero de ellos, el modelo de coste compartido, ha sido utilizado con éxito en el desarrollo del servidor Web Apache, que en la actualidad es el más utilizado del mundo.

El segundo modelo, basado en la distribución de riesgos, fue aplicado con éxito por Cisco para evitar los riesgos derivados del abandono de la compañía por parte del personal que desarrollaba el servidor de impresión CiscoPrint [K1 RAY] [TRO 08].

3.1.5.1.1.1. Modelo de coste compartido

Este modelo de negocio se fundamenta en la idea de compartir el coste de desarrollo de una aplicación con la comunidad open source. Imaginemos que una organización necesita una aplicación crítica para su negocio que debe ser fiable, rápida y personalizable. En esta situación existen tres posibilidades distintas para lograr dicha aplicación:

1. Comprar una aplicación propietaria. En este caso la organización tiene que sincronizar su agenda con la del fabricante. Este, a su vez, tiene que disponer de la suficiente competencia técnica para implementar la aplicación correctamente. Aunque ambas premisas se cumplan, la aplicación resultante será poco personalizable, ya que la organización sólo podrá modificar la aplicación a través de los mecanismos establecidos por el fabricante.

2. Desarrollar una aplicación propia. Esta posibilidad no es una opción real si se desea disponer de la aplicación en un corto plazo de tiempo. Siguiendo este camino, la organización logrará las características y capacidad de personalización deseadas, pero tendrá que pagar por ello en tiempo de desarrollo. Adicionalmente, cuando el equipo de desarrollo abandone la organización puede surgir un serio problema.

3. Unirse a un proyecto open source. Mediante este camino la organización colabora con un proyecto open source que se encuentre dentro del dominio de la aplicación deseada. De este modo se capturan las ventajas de un desarrollo propio y el poder de la comunidad open source para potenciar y depurar la aplicación, logrando un producto mejor y con menor coste.

El modelo de coste compartido amplifica el valor de uso de la aplicación porque tiene como objetivo máximo lograr las funcionalidades que la organización necesita.

3.1.5.1.1.2. Modelo de distribución de riesgos

El modelo de distribución de riesgos tiene como objetivo fundamental minimizar la dependencia de una organización respecto del equipo de desarrollo que fabrica sus productos software, aunque también es posible utilizarlo para mitigar cualquier riesgo inherente al desarrollo de una aplicación.

La estrategia básica de este modelo consiste en desarrollar software distribuyendo los riesgos dentro de la comunidad open source, de modo que la responsabilidad del proyecto no sólo recae sobre la organización que lo impulsa.

Aunque en cierto sentido este modelo es semejante al de coste compartido, la diferencia entre ambos radica en que el modelo de distribución de riesgos no se centra en la reducción de los costes de desarrollo sino en la reducción de los riesgos que conlleva.

3.1.5.1.2. Modelos de negocio sobre el valor de venta indirecto

El software open source hace muy difícil la captura directa del valor de venta. Esta dificultad no es técnica porque el código fuente de una aplicación no se copia más fácilmente que los binarios. Además, las leyes del copyright y licencia no son de elaboración más compleja en el open source.

La dificultad reside en la naturaleza del contrato social que soporta el desarrollo open source. La mayoría de las licencias open source restringen ampliamente los usos, redistribuciones y modificaciones que facilitan la captura directa del valor de venta. Para comprender este comportamiento es necesario analizar el contexto social en el que se desarrollaron dichas licencias: la cultura de los *hackers* [31] de Internet (no confundir con *cracker*[32])

Al contrario de lo que se piensa popularmente, la mayoría de los hackers no son hostiles al mercado del software. Este hecho es fácilmente demostrable observando la gran cantidad de hackers que colaboran con las distribuciones comerciales de GNU/Linux. Las verdaderas razones que llevan a los hackers hacia la obstrucción de la captura directa del valor de venta son mucho más sutiles.

En primer lugar, los hackers pretenden que la comercialización de aplicaciones open source se lleve a cabo en igualdad de condiciones. En general, los desarrolladores de la comunidad open source no se oponen a que otras personas u organizaciones obtengan beneficios económicos a partir de su trabajo, pero si de mandan que ninguna de las partes tenga una posición privilegiada para obtener estos beneficios.

En segundo lugar, la comunidad open source pretende evitar los efectos colaterales del intento de obtener beneficios directos a partir del valor de venta. Por ejemplo, las licencias que incluyen restricciones o cuotas para el uso comercial o venta son muy perjudiciales porque limitan y complican la distribución del software en recopilaciones, por ejemplo las distribuciones de GNU/Linux.

---

[31] "hacker: [...] Una persona que se divierte explorando los detalles de los sistemas programables y averiguando como potenciar sus capacidades, contrariamente a la mayoría de los usuarios, que prefieren aprender lo mínimo necesario. [...] Alguien que programa de forma entusiasta (incluso obsesivamente) o que se divierte programando en lugar de teorizar sobre la programación. [...] Una persona que es buena programando rápidamente. [...] Un experto en un programa particular, o alguien que frecuentemente realiza trabajo con él, o sobre él; por ejemplo un hacker de UNIX [...]" [K1 RAY]

[32] "cracker: Alguien que rompe la seguridad de un sistema [...]" [K1 RAY]

Los hackers consideran que las restricciones de este tipo van en contra de la propia filosofía del open source, en la cual se busca una amplia distribución y uso de las aplicaciones. Por ello, existe una presión social fuerte dentro de la comunidad open source por mantener las licencias simples y libres de restricciones.

Sin embargo, el principal motivo de oposición a la captura directa del valor de venta reside en el intento de preservar la capacidad de ramificación de los proyectos. Las licencias que capturan directamente el valor de venta prácticamente hacen imposible que el proyecto pueda dividirse en otras líneas de trabajo. Como ya se ha expuesto en varias ocasiones, la capacidad de personalizar una aplicación y ponerla a disposición de la comunidad es una de las bases del open source y el mero hecho de impedir estas ramificaciones sobre un proyecto choca de frente con la propia filosofía del movimiento.

A pesar de los impedimentos que el movimiento open source impone a la captura directa del valor de venta, existen modelos de captura indirecta que pueden ser aplicados de forma efectiva. En los siguientes apartados se describirán siete modelos de este tipo, de los cuales dos son de carácter especulativo.

3.1.5.1.2.1. Modelo de posicionamiento basado en software propietario

En este modelo el open source se utiliza para crear o mantener la posición de mercado de un software propietario que genera flujos de beneficio directos. En la variante más común de esta estrategia, una aplicación cliente open source permite la venta de software servidor o la suscripción a servicios de un portal.

El ejemplo más conocido de esta estrategia es la apertura del código fuente del navegador Web de Netscape. Cuando Microsoft lanzó al mercado su navegador Web Internet Explorer, la cuota de mercado de Netscape se redujo considerablemente y Microsoft empezó a lograr un control de facto sobre el lenguaje HTML y el protocolo HTTP. Al transformar su navegador Web en un proyecto open source, Netscape impidió de forma efectiva que Microsoft monopolizase el mercado de los navegadores. Con este movimiento, Netscape esperaba que el apoyo de la comunidad open source acelerase el desarrollo y depuración de su navegador, evitando que Microsoft definiera de forma exclusiva las siguientes versiones del lenguaje HTML.

3.1.5.1.2.2. Modelo de posicionamiento basado en servicios

En este modelo, una organización desarrolla software open source para crear una posición de mercado para los servicios que desea ofrecer. Al contrario del modelo anterior, en este caso no se pretende posicionar software propietario.

Este modelo fue utilizado por primera vez en la empresa Cygnus Solutions[33] en el año 1989. En estas fechas, las herramientas del proyecto GNU proporcionaban un entorno de desarrollo común sobre varios tipos de máquinas distintos, pero cada herramienta requería un proceso de configuración y una serie de parches específicos para ejecutarse en cada plataforma. Cygnus facilitó este proceso mediante la creación de la utilidad configure, que unificaba el proceso de construcción y compilación de las herramientas GNU, y para la cual comercializaban servicios de soporte. De acuerdo con la licencia GPL, Cygnus permitía a sus clientes que usasen, distribuyesen y modificasen libremente el software, pero el contrato de servicios de soporte terminaba, o debía abonarse una cuota mayor, cuando había más usuarios utilizando los servicios de soporte que los acordados en el contrato.

Políticas similares a la de Cygnus son las que siguen las distribuciones de GNU/Linux. Lo que se comercializa en estos casos no son los bits de las aplicaciones, sino el valor añadido de recopilar y testear un sistema operativo completo. A esto hay que añadir otros servicios como el soporte de instalación y diversas opciones de mantenimiento continuado del producto.

La estrategia de posicionamiento entorno a servicios es extraordinariamente poderosa para aquellas empresas que comienzan a desarrollar su actividad bajo este modelo. Un ejemplo muy reciente lo encarna Digital Creations[34], una empresa especializada en el diseño de sitios Web que comenzó su actividad en 1995. Su aplicación más importante es una herramienta que permite publicar diversos tipos de objetos en la WWW y que después de ser denominada de diversas formas en la actualidad se llama Zope[35]. Zope fue liberado a la comunidad open source por dos motivos fundamentales. El primero señalaba que el valor de la herramienta estaba ligado fuertemente a las habilidades de sus desarrolladores. El segundo motivo estimaba que Zope sería más valiosa como constructor de mercado que como una herramienta secreta.

[33] http://www.cygnus-solutions.com/
[34] http://www.zope.com/
[35] http://en.wikipedia.org/wiki/Zope

Imaginemos que Zope hubiese permanecido como arma secreta de Digital Creations y que fuese una herramienta muy efectiva que permitiese desarrollar sitios Web de altísima calidad. El problema residiría en que nadie lo sabría. Zope lograría gran satisfacción entre sus clientes, pero construir una base de clientes sobre la cual comenzar sería muy complicado. Al proporcionar Zope a la comunidad open source, Digital Creations pretendía que los posibles clientes que evaluasen Zope considerasen más eficiente contratar a Digital Creations para personalizar la aplicación que desarrollarla en la propia organización. Esto era especialmente importante en las organizaciones no dotadas de personal informático de desarrollo.

La estrategia de Digital Creations está funcionando en la actualidad de forma extraordinaria, ya que ha conseguido un equilibrio adecuado entre el apoyo a la comunidad open source y sus objetivos de negocio. Sin ninguna duda, Zope es un claro ejemplo de que el open source puede ser una alternativa de negocio poderosa.

3.1.5.1.2.3. Modelo consolidación de hardware

Como su propio nombre hace intuir, este modelo suele ser aplicado por los fabricantes de hardware. En este contexto, las presiones del mercado han forzado a las compañías de hardware a escribir y mantener software (controladores de dispositivos, herramientas de configuración, etc) para diversas plataformas tecnológicas. Sin embargo, este tipo software en si mismo no supone beneficio alguno y a la larga se convierte en una sobrecarga.

Tradicionalmente, los fabricantes de hardware han sido muy reticentes a la apertura del código fuente de sus controladores de dispositivo. Esta reticencia se basa en el hecho de que este código fuente puede desvelar características internas del producto a sus competidores.

Hace tiempo, cuando los ciclos de desarrollo del hardware eran de tres a cinco años, este podía ser un argumento válido. Sin embargo, en la actualidad el hardware se desarrolla con ciclos mucho más cortos y los competidores no disponen de tiempo material para copiar el hardware de otro fabricante.

Abrir este tipo de software a la comunidad open source no tiene ningún efecto negativo, ya que no se corre el riesgo de perder beneficio alguno. Al contrario, el fabricante obtiene drásticamente un grupo de

desarrolladores de tamaño considerable que pueden cubrir de forma más rápida y flexible las necesidades de sus clientes y que portarán de forma gratuita el software a varias plataformas tecnológicas.

Adicionalmente, el usuario sale altamente beneficiado de esta política, ya que cuando la fabricación de un determinado modelo de hardware finalice el software asociado podrá seguir siendo corregido y ampliado.

Uno de los casos mas recientes de éxito con este tipo de modelo es el Digium[36], empresa creadora del famoso software open source de telefonía IP Asterisk. Digium fabrica diverso hardware, como tarjetas PCI diseñadas para trabajar con Asterisk y obtiene grandes beneficios de la venta de este hardware, el cuales no tendría el éxito que tiene de no ser por que Asterisk[37] se ha convertido en el software de referencia en centralitas telefónicas de VozIP.

3.1.5.1.2.4. Modelo de accesorios

Dentro de este modelo la clave de la estrategia reside en la comercialización de accesorios relacionados con la aplicación open source. Estos accesorios pueden ser desde camisetas hasta documentación editada y producida profesionalmente.

O'Reilly & Associates Inc[38] . es uno de los ejemplos más significativos del modelo basado en accesorios. Se trata de una editorial que dedica su actividad a la publicación de excelentes volúmenes de referencia sobre aplicaciones open source. En la actualidad O'Reilly contrata a hackers muy conocidos como mecanismo para construir una buena reputación dentro de su mercado.

Otro ejemplo es la distribución de GNU/Linux Ubuntu también recurre a este sistema como una parte de su negocio vendiendo diferentes artículos como camisetas, memorias USB y diversos accesorios personalizados con los logotipos de la distribución.[39]

[36] http://en.wikipedia.org/wiki/Digium
[37] http://en.wikipedia.org/wiki/Asterisk_PBX
[38] http://oreilly.com/
[39] http://shop.canonical.com/

3.1.5.1.2.5. Modelo de licencia cerrada expirable

En este modelo el software se entrega al usuario en forma binaria y fuente con una licencia cerrada que incluye una fecha de expiración para las cláusulas cerradas. Por ejemplo, podría utilizarse una licencia que permitiese las redistribución gratuita, prohibiese el uso comercial sin abonar una cuota y que garantizase que el software entraría dentro la licencia GPL un año después de la entrega del software al cliente, o si el fabricante cesa su actividad.

Bajo este modelo, los clientes se aseguran de que el producto podrá ser ajustado a sus necesidades porque disponen del código fuente. Adicionalmente, la continuidad del proyecto está garantizada en caso de que el fabricante deje de mantenerlo, ya que pasa automáticamente a la comunidad open source.

Dado que bajo este modelo las revisiones de código que realiza la comunidad open source se realizan sobre código antiguo, existe una penalización sobre el potencial total de depuración que la comunidad puede ofrecer. De hecho, este modelo puede ser un inconveniente en fases iniciales de un proyecto, cuando la capacidad de apoyo de la comunidad open source es más necesaria. Sin embargo, las organizaciones que emplean el modelo todavía obtienen una reducción del 75 % en las tareas de mantenimiento ([K1 RAY]).

Este modelo ha sido utilizado por Aladdin Enterprises[40], fabricantes del popular programa Ghostscript, un intérprete del lenguaje PostScript[41] que puede comunicarse con múltiples impresoras.

3.1.5.1.2.6. Modelo de marcas de certificación

Este es un modelo de negocio especulativo en el cual la tecnología open source retiene un conjunto de tests o criterios de compatibilidad. De este modo, los usuarios reciben una marca de certificación que asegura que la implementación es compatible con las otras que disponen de la misma marca.

[40] http://www.aladdin.com
[41] http://www.adobe.com/products/postscript/index.html

Este modelo es utilizado por Sun Microsystems para certificar las líneas de desarrollo de StarOffice[42] surgidas a partir de la liberación del código fuente de este paquete de ofimática.

3.1.5.1.2.7. Modelo de contenidos

Se trata de otro modelo de negocio especulativo. Se fundamenta en el hecho de comercializar la suscripción a los contenidos almacenados en aplicaciones de servicios, mientras que el software cliente y servidor se distribuyen bajo open source. De este modo, la comunidad open source mejorará y portará el software a diversas plataformas, logrando una expansión del mercado automática.

Un ejemplo de este modelo lo constituyen las aplicaciones clientes de AOL[43] .

3.1.6. Criterios para el desarrollo de un proyecto Open Source

A lo largo de los apartados anteriores se han descrito varios modelos de negocio que pueden ser aplicados con éxito sobre la base del open source. Sin embargo, la cuestión más importante quizás radique en saber cuándo una aplicación debe ser desarrollada de forma cerrada y cuándo debe hacerlo sobre la filosofía del open source. La respuesta a esta cuestión es sencilla. Debe elegirse el camino del open source cuando los beneficios obtenidos a partir de la comunidad que soporta este movimiento son superiores a los beneficios obtenidos mediante la comercialización cerrada.

La sencilla respuesta anterior plantea ahora un nuevo dilema. ¿Cómo se pueden medir los posibles beneficios de la comunidad open source sobre un proyecto software?. Realmente, estos beneficios son de muy difícil cuantificación si se compara con la medición de los beneficios de comercialización del software propietario. Quizás, la única posibilidad disponible en la actualidad para cuantificar las aportaciones de la comunidad open source sea observar qué proyectos han tenido éxito y bajo qué condiciones.

[42] http://es.wikipedia.org/wiki/StarOffice
[43] http://www.aol.com

En este sentido, es posible definir cinco criterios básicos que pueden ser tenidos en cuenta a la hora de decidir la implementación de una aplicación bajo el prisma del open source:

1. La fiabilidad, estabilidad y escalabilidad son críticas. Si es necesaria cualquiera de estas características, la capacidad de mejora y revisión aportada por la comunidad open source es fundamental.

2. La corrección del diseño y la implementación no pueden ser verificadas fácilmente sin una revisión independiente de los desarrolladores. Realmente, este criterio es necesario prácticamente en todas las aplicaciones que no sean triviales.

3. El software es crítico para la gestión de la organización. Muchas de las organizaciones que optan por el open source intentan evitar la dependencia respecto de un fabricante concreto. De hecho, cuanto más crítico es el software para la organización, más tendencia hay hacia la búsqueda de soluciones que eviten el monopolio de un proveedor.

4. El software establece o permite una infraestructura de comunicación y computación común. Se ha observado que la filosofía open source tiene un mayor ratio de éxito en la implantación de tecnologías de este tipo. Tanto es así, que muchos de los proyectos open source de éxito se mueven dentro de este dominio.

5. Los procesos clave son parte de la base de conocimientos común de ingeniería. La filosofía open source es extremadamente efectiva cuando la algoritmia utilizada es global y comúnmente conocida dentro del campo de la ingeniería informática. Si las funcionalidades a lograr requieren algoritmos poco conocidos, las potencialidades de la comunidad open source son mermadas significativamente.

3.1.7. Open Source frente a software cerrado

Uno de los mitos sobre el open source es que el software propietario resuelve mejor las necesidades de los usuarios. Nada más lejos de la realidad. Adquirir costosas licencias de software cerrado sólo garantiza que se gastará el dinero y no que los problemas serán resueltos. El open source, al ser explorado por un gran número de expertos de todo el mundo, mejora notablemente la calidad del producto final respecto al software cerrado. Es

imposible que una multinacional disponga del tal número de expertos y colaboradores para participar en el desarrollo, depuración y ampliación de los productos. Esto hace que las grandes multinacionales del software cerrado encuentren graves dificultades para mantener al día sus productos y competir con sus equivalentes open source.

En la actualidad, el código abierto, y el nulo o bajo coste de las aplicaciones open source, están eliminando las barreras de aprendizaje y uso de los sistemas informáticos. Muchos desarrolladores que desean obtener formación en plataformas propietarias se topan de frente con precios de varias cifras. En consecuencia, estos desarrolladores se desplazan sistemáticamente hacia alternativas open source de calidad.

Como ya se ha mencionado en numerosas ocasiones a lo largo de todo el capítulo, el open source cambia por completo los mecanismos de desarrollo y distribución de las aplicaciones. Este paradigma se encuentra en el lado opuesto al empleado por los fabricantes de software propietario. En lugar de mantener el software en secreto para los usuarios, el open source establece un proceso basado en una comunidad pública de usuarios. Esto hace que muchas organizaciones, que en la actualidad se encuentran en permanente lucha con los sistemas propietarios, estén dando el salto hacia el open source porque encaja mejor con sus objetivos y les permite alcanzar el éxito más fácilmente.

3.1.7.1. Beneficios del Open Source

Con el fin de ordenar, matizar y resumir muchas de las diferencias que han quedado patentes entre ambos paradigmas a lo largo de este capítulo, los siguientes apartados describirán los beneficios del open source respecto del software propietario

3.1.7.1.1. Eliminación de los ciclos de actualización innecesarios

Una de las estrategias más populares de las compañías de software propietario consiste en el desarrollo de productos en múltiples generaciones de versiones. Esta estrategia es adecuada, ya que es justamente después de la creación de un producto cuando se recibe la mayor retroalimentación por parte los usuarios. Adicionalmente, también puede ser necesario adaptar el software a los nuevos avances de la tecnología.

Añadir funcionalidades en este sentido producirá aplicaciones mejores de forma gradual. El problema reside cuando los avances y mejoras se suprimen intencionadamente de una versión con el fin de prolongar los ciclos de actualización del producto, estrategia que aumenta los beneficios de la multinacional. En este caso, los usuarios deben esperar de forma intencionada por parte del fabricante meses e incluso años para disponer de las nuevas funcionalidades. Los fabricantes de software cerrado toman este tipo de decisiones basándose en sus objetivos de negocio y en la mayoría de las ocasiones tales objetivos no coinciden con los de los usuarios.

El software open source elimina los ciclos de actualización innecesarios porque se añaden nuevas características de forma constante basándose en las demandas de los usuarios. Esto hace que los proyectos open source tengan un mayor número de ciclos de vida pequeños en lugar de los ciclos de vida largos del software propietario. Las mejoras son puestas a disposición de los usuarios rápidamente, permitiendo que realicen más productivamente su trabajo.

3.1.7.1.2. Personalización del software ilimitada

Esta es una gran ventaja si la organización en la que se utiliza open source tiene el deseo y la experiencia suficiente como para adaptar el software. En muchas ocasiones puede lograrse una mayor eficiencia y efectividad de las aplicaciones realizando pequeñas adaptaciones que lo ajusten a las necesidades de la organización. Mediante el acceso al código fuente inherente en el open source este tipo de actuaciones se convierten en una realidad.

Si una organización desea realizar ajustes sobre software propietario, en la mayoría de los casos se encontrará con una negativa rotunda del fabricante. En los pocos casos en que esto sea posible, el fabricante nunca permitirá que las modificaciones sean realizadas por el cliente y por tanto este deberá abonar cuantiosas cifras para que el fabricante realice las adaptaciones oportunas.

3.1.7.1.3. Aplicaciones diseñadas y probadas por una comunidad mundial

Este beneficio no se sustenta solamente en el bajo precio del software open source, sino también en el ahorro que se produce como consecuencia del uso de productos fiables y que tienen una base de usuarios importante. Otro de los factores de ahorro en el open source reside en la baja barrera de entrada de los empleados de nuevas tecnologías en el uso de este tipo de software. Esto crea una fuerza de trabajo sobre el producto mucho mayor y experta que la lograda por los sistemas propietarios.

3.1.7.1.4. Eliminación de las restricciones en las licencias

Las licencias de los sistemas propietarios pueden llegar a ser una verdadera pesadilla. Por ejemplo, existen licencias que sólo permiten la realización de copias por motivos de seguridad. Otras sólo sirven para desarrollo y a la hora de poner el sistema en producción deben adquirirse nuevas y caras licencias. Licencias de este tipo son habituales en el mundo de las aplicaciones propietarias por que intentan maximizar el beneficio del fabricante, aumentando en consecuencia el coste para el cliente. Los problemas con las licencias han ralentizado muchos proyectos porque la incapacidad para obtenerlas ha dilatado su desarrollo y fecha de entrega.

El open source elimina todas estos problemas con las licencias porque es mucho más flexible y consciente de las necesidades reales del usuario. Por ejemplo, es posible comprar una copia de SuSE Linux por su documentación impresa y facilidad de uso, e instalarla en toda la organización o incluso regalarla a sus amigos.

3.1.7.2. Riesgos del Open Source

En el mundo moderno, las personas tienen tendencia a buscar soluciones mágicas que curen todos los males. Sin embargo, todos sabemos que dichas soluciones no existen. Del mismo modo, el open source no resuelve instantáneamente todos los problemas tecnológicos. En este apartado analizaremos los riesgos que se corren cuando se toma la decisión de utilizar software open source.

3.1.7.2.1. No entender qué problemas puede resolver una solución tecnológica

Este es un riesgo inherente tanto en el software open source como en el propietario. Sin embargo, es necesario aclarar que la elección de una solución open source siempre se basa en sus propios méritos y no en las expeditivas campañas publicitarias que acompañan a numerosos productos propietario

3.1.7.2.2. Uso indiscriminado porque es gratis

El hecho de que un determinado producto sea open source no indica necesariamente que sea bueno. Esto mismo sucede con el software propietario. Algunos sistemas propietarios funcionan correctamente y otros no. Una empresa o proyecto sólo debe aceptar soluciones open source fiables, seguras y robustas. Salvo que una organización pretenda finalizar o personalizar algún proyecto open source, sólo debe utilizar aplicaciones que se encuentren listas para producción.

3.1.7.2.3. Software no debidamente documentado

Existen miles de proyectos open source. La mayoría son aplicaciones y utilidades simples que han sido desarrolladas por una única persona. En general los desarrolladores de software son conocidos por no documentar su trabajo. Los desarrolladores de open source no son diferentes. Aunque una aplicación sea open source, la documentación proporciona una reducción sustancial de la curva de aprendizaje, siendo difícil comprender las características y limitaciones de la aplicación si carecemos de la correspondiente documentación.

Tradicionalmente, la mayoría de las aplicaciones propietarias han dispuesto de una documentación adecuada. Pero también es cierto que en los últimos años el intento de reducción de costes para competir con el open source ha llevado a muchos de los fabricantes a minimizar esta documentación o a proporcionarla por otras vías de negocio, por ejemplo los cursos de formación.

Las consecuencias de implantar en una organización sistemas open source pésimamente documentados pueden ser nefastas. Sin embargo, hay que decir a favor del open source que la mayoría de sus proyectos de calidad están bien documentados.

3.1.7.2.4. Soporte no adecuado

Es parte del camino que lleva a la implantación de un sistema informático encontrar dificultades. El soporte experto es fundamental para lograr una correcta velocidad y eficiencia de los recursos durante la implantación, sobre todo cuando el equipo de trabajo se queda bloqueado con un determinado problema.

En la mayoría de los casos los sistemas propietarios llevan asociado un mínimo de soporte, aunque en los últimos tiempos hay una cierta tendencia en los fabricantes a que el soporte avanzado sea adquirido por separado del producto.

Esta seguridad en la capacidad de recibir soporte de los productos propietarios es uno de los aspectos que más temor infunde a los usuarios que desean utilizar open source.

Hay dos soluciones potenciales para obtener soporte adecuado de aplicaciones open source. La primera de ellas es utilizar los foros de mensajes que generalmente se encuentran disponibles en los sitios Web del proyecto open source. Generalmente, en estos foros se pueden encontrar personas competentes que pueden proporcionar la ayuda necesaria. Sin embargo, estos participantes tendrán distintos niveles de experiencia y las respuestas obtenidas pueden no resolver los problemas planteados.

La segunda forma de resolver incidencias de soporte con aplicaciones open source es a través de compañías que ofrecen soporte comercial. Generalmente, las mejores compañías a las que se puede recurrir son aquellas que esponsorizan el proyecto open source que se está implementando. En la mayoría de los casos el coste de este soporte comercial es inferior al tiempo invertido por el equipo de implementación en resolver el problema por si mismos.

3.1.7.2.5. Software no testado adecuadamente

El aseguramiento de la calidad es uno de los aspectos más ignorados en el proceso de ingeniería de software. En muchas ocasiones, los fabricantes de software propietario entregan al cliente productos pobremente testados para cumplir los plazos o los objetivos comerciales.

El software open source también adolece de este problema. Algunos de estos proyectos se centran en escribir código y en ocasiones no son probados adecuadamente en varias plataformas. Esto puede resultar en aplicaciones que funcionan correctamente en una plataforma determinada y pésimamente en otra. Sin embargo, en el caso del open source esto solamente suele ocurrir en proyectos pequeños o que se encuentran en sus fases iniciales de desarrollo.

3.1.8. Licencias Open Source

El progresivo desarrollo y aumento de popularidad de las aplicaciones open source ha conducido a la aparición de múltiples licencias de software que acompañan a los productos de este tipo. La mayoría de las licencias asociadas a proyectos open source son similares, aunque es posible realizar una clasificación general según la capacidad de contaminación de la licencia en desarrollos futuros sobre un producto ya licenciado:

- **Licencias virales**. Este tipo de licencias requieren que todos los trabajos derivados a partir de un software licenciado también continúen bajo la misma licencia.
- **Licencias no virales**. En contraste a las licencias virales, las licencias no virales no contaminan las licencias de los desarrollos derivados del software original. Dicho de otro modo, permiten establecer para el software derivado una licencia distinta de la original.

Las licencias open source más importantes y conocidas en la actualidad son las siguientes:

- **GNU General Public License (GPL).** Sin ninguna duda esta es la licencia open source más popular. Fue creada originalmente por Richard Stallman y la Free Software Fundation. Esta licencia viral protege las libertades de usuarios y desarrolladores asegurando que el software seguirá siendo libre en sus futuras encarnaciones. En contraste con otras licencias open source elaboradas por los departamentos legales de algunas compañías, la licencia GPL es de fácil lectura y compresión

- **GNU Affero General Public License (Affero GPL).** Es una versión de la GPL pensada para el software que se ejecuta en servidores Web. La Affero GPL es íntegramente una GNU GPL con una cláusula nueva (sección 2(d)) que añade la obligación de distribuir el software si éste se ejecuta para ofrecer servicios a través de una red de ordenadores. Al igual que GNU GPL es una licencia viral.

- **GNU Library General Public License (LGPL).** Esta licencia, también conocida como Lesser GPL, es similar a la GPL. La mayor diferencia con la anterior reside en el hecho de que los programas que utilicen la licencia LGPL no tienen que continuar con la licencia LGLP. Dicho de otro modo, la LGPL es una licencia no viral. Por ejemplo, usar librerías licenciadas con GPL en una nueva aplicación hace que esta pase automáticamente a ser licenciada también bajo la GPL. Sin embargo, con la licencia LGPL esto no sucede.

- **Mozilla Public License.** Esta licencia fue creada por Netscape para cubrir la apertura del código fuente de su navegador Web dentro del proyecto Mozilla. Se trata una licencia viral de difícil lectura y comprensión debido a su gran complejidad legal.

- **Apache Software License**. Esta licencia de tipo viral es utilizada por la Fundación Apache, origen de numerosos proyectos de gran éxito dentro del mundo empresarial y la comunidad open source.

- **Artistic License**. En esta licencia no viral se proporciona al poseedor del copyright el control artístico sobre el desarrollo del producto, mientras se protege la libertad de los usuarios. Existen numerosas variantes de esta licencia, por ejemplo la que acompaña al lenguaje perl.

- **Python Software Fundation License**. Se trata de una licencia no viral que acompaña al lenguaje Python y es utilizada en muchos programas open source realizados con este lenguaje de programación.

- **Zope Public License (ZPL).** Esta licencia no viral acompaña al servidor de aplicaciones Zope y a la gran mayoría de los módulos que colaboradores de todo el mundo desarrollan para este software.

- **BSD Copyright.** Esta licencia constituye la espina dorsal de los sistemas operativos BSD. Se trata de una licencia no viral que no excluye la posibilidad de incluir el software dentro de un producto cerrado.

- **Aladdin Free Public License (AFPL).** Esta licencia se utiliza para la distribución del producto Aladdin Ghostscript. Se trata de una licencia muy peculiar (apartado 4.6.3.5) porque libera el software bajo la GPL una vez transcurrido cierto tiempo.

3.2. GNU/Linux.

El nombre GNU viene de las herramientas básicas de sistema operativo creadas por el proyecto GNU[44], iniciado por Richard Stallman en 1983 y mantenido por la FSF. El nombre Linux viene del núcleo Linux, inicialmente escrito por Linus Torvalds en 1991.

La contribución de GNU es la razón por la que existe controversia[45] a la hora de utilizar Linux o GNU/Linux para referirse al sistema operativo formado por el conjunto de las herramientas de GNU y el núcleo Linux en su conjunto.

Su desarrollo es uno de los ejemplos más prominentes de software libre; todo el código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL.

[44] http://www.gnu.org/
[45] http://www.gnu.org/gnu/why-gnu-linux.es.html

El proyecto GNU, iniciado en 1983 por Richard Stallman[46], tiene como objetivo el desarrollo de un sistema operativo Unix completo compuesto enteramente de software libre. La historia del núcleo Linux está fuertemente vinculada a la del proyecto GNU. En 1991 Linus Torvalds empezó a trabajar en un reemplazo no comercial para MINIX[47] que más adelante acabaría siendo Linux.

Cuando la primera versión del núcleo Linux fue liberada el proyecto GNU ya había producido varios de los componentes fundamentales del sistema operativo, incluyendo un intérprete de comandos, una biblioteca C y un compilador, pero su núcleo "*Hurd*" no estaba lo suficientemente maduro como para completar el sistema operativo.

Entonces, el núcleo creado por Linus Torvalds, quien se encontraba por entonces estudiando en la Universidad de Helsinki, llenó el "espacio" final que en sistema operativo de GNU.

Las variantes de este sistema se denominan distribuciones y su objetivo es ofrecer una edición que cumpla con las necesidades de determinado grupo de usuarios. Algunas distribuciones son especialmente conocidas por su uso en servidores y supercomputadoras. No obstante, nos centraremos principalmente en las dos distribuciones empleadas en el presente proyecto: **Ubuntu** y **OpenWRT.**

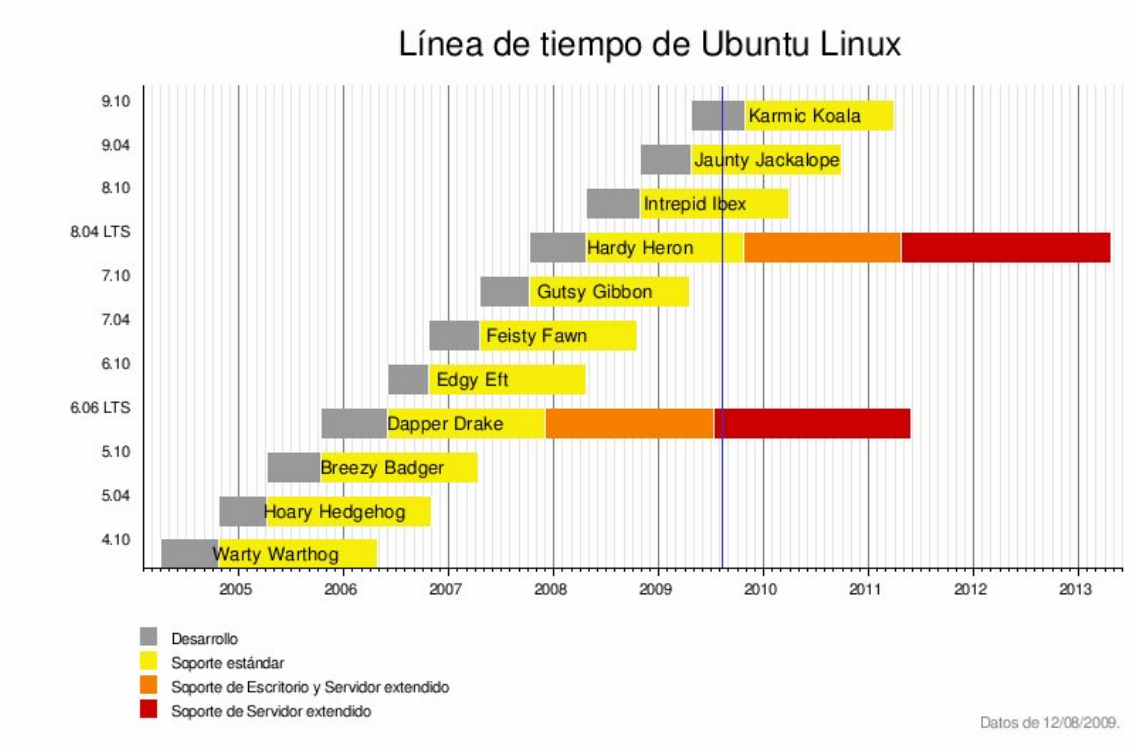
3.2.1. Ubuntu

Ubuntu Linux, es una distribución GNU/Linux basada en Debian GNU/Linux, cuyo nombre proviene de la ideología sudafricana *Ubuntu*[48] que se podría resumir como "*humanidad hacia otros*"

Ubuntu está patrocinando por Canonical Ltd.[49], una compañía británica propiedad del empresario sudafricano Mark Shuttleworth que se financia por varios de los medios antes estudiados en modelos de negocio basados en software libre, por ejemplo por medio de servicios vinculados al sistema operativo y vendiendo soporte técnico, además de *merchandising* como camisetas, tazas y diversos objetos con el logo de Ubuntu.

[46] http://www.gnu.org/gnu/initial-announcement.es.html
[47] Minix es un clon del sistema operativo Unix distribuido junto con su código fuente y desarrollado por el profesor Andrew S. Tanenbaum en 1987. Fue creado para enseñar a sus alumnos el diseño de sistemas operativos en la Vrije Universiteit de Ámsterdam.
[48] https://help.ubuntu.com/9.04/about-ubuntu/C/about-ubuntu-name.html
[49] http://www.canonical.com/

Desde su inicio proyecto Ubuntu se ha comprometido a un ciclo de lanzamiento y ha logrado cumplir con ese compromiso sin falta. Es la regularidad y la fiabilidad de estas versiones de Ubuntu que hace una gran opción para los usuarios y las empresas que pueden planear las actualizaciones y las nuevas instalaciones con una fiabilidad que es muy inusual en el mercado de los sistemas operativos.



Cada seis meses se libera una nueva versión de Ubuntu la cual recibe soporte por parte de Canonical, durante dieciocho meses, por medio de actualizaciones de seguridad, parches para errores críticos y actualizaciones menores de programas. Las versiones LTS (Long Term Support), que se liberan cada dos años, reciben soporte durante tres años en los sistemas de escritorio y cinco para la edición orientada a servidores.

En el presente proyecto se ha utilizado Ubuntu Server 8.04 LTS, que tiene el soporte asegurado hasta abril de 2013 tal y como se puede observar en el gráfico de arriba.

3.2.2. OpenWRT

OpenWRT es una distribución de Linux basada en firmware usada para dispositivos empotrados tales como routers personales. El soporte fue limitado originalmente al modelo Linksys WRT54G, pero desde su rápida expansión se ha incluido soporte para otros fabricantes y dispositivos.

En el presente proyecto se ha evaluado con éxito la instalación y configuración de OpenWRT como punto de acceso inalámbrico del gestor de ancho de banda en los siguientes dispositivos: La Fonera[50] y Ubiquiti PowerStation y NanoStation[51].

Los tres routers inalámbricos mencionados tienen dos características esenciales que los hacen aptos para ser usados como puntos de acceso del gestor de ancho de banda.

- Se puede instalar OpenWRT en ellos sin ningún problema
- La tarjeta inalámbrica (WiFi) que incorporan está basada en un chip Atheros[52].

Esta última característica que podría parecer trivial a simple vista no lo es en absoluto, ya que gracias a los proyectos MadWifi[53] y Ath5k[54] se disponen de unos controladores (drivers) para tarjetas WiFi basadas en chip Atheros que permiten configuraciones únicas que no soportan otro tipo de drivers/tarjetas. En concreto me estoy refiriendo a:

- Posibilidad de poner la tarjeta en modo AP (“punto de acceso”).
- Posibilidad de crear múltiples APs virtuales (VAP)

Gracias a los puntos de acceso virtuales, con un solo dispositivo de este tipo se pueden crear dos redes inalámbricas con diferentes SSIDs[55].

En concreto en el presente proyecto se crean las siguientes redes inalámbricas:

[50] http://es.wikipedia.org/wiki/La_Fonera
[51] http://www.ubnt.com/products/ps.php
[52] http://www.atheros.com/
[53] http://madwifi-project.org/
[54] http://linuxwireless.org/en/users/Drivers/ath5k
[55] El SSID (Service Set IDentifier) es un código incluido en todos los paquetes de una red inalámbrica (Wi-Fi) para identificarlos como parte de esa red.

- Red WiFi pública: sin encriptación, es la red a la que se conectará por primera vez un usuario ya que no hace falta ninguna contraseña para conectarse. Al intentar acceder a Internet el usuario recibirá un mensaje de que necesita comprar un “paquete de acceso” al sistema.

- Red WiFi con encriptación WPA: En esta red todo el tráfico se envía y recibe encriptado, para acceder a ella el usuario ha de introducir sus credenciales previamente (nombre de usuario y contraseña). La autenticación del usuario se realiza usando el protocolo PEAP (*Protected Extensible Authentication Protocol*)

Un potencial usuario que busque redes inalámbricas encontrara dos redes diferentes con diferentes nombres una encriptada y otra no (por ejemplo: *clubnautico* y *clubnautico_wpa*). Sin embargo no hay dos puntos de acceso, solamente hay uno que gestiona ambas redes, de ahí la elección de puntos de acceso con chip Atheros y que estén soportados por OpenWRT.

3.2.2.1. HostAPd

HostAPd[56] es un *demonio*[57] para puntos de acceso y servidores de autenticación. Implementa los protocolos de autenticación IEEE 802.1X/WPA/WPA2/EAP, incorpora un cliente RADIUS, servidor EAP y un servidor de autenticación RADIUS. La actual versión funciona con los drivers inalámbricos madwifi, Prism54 y los basados en mac80211.

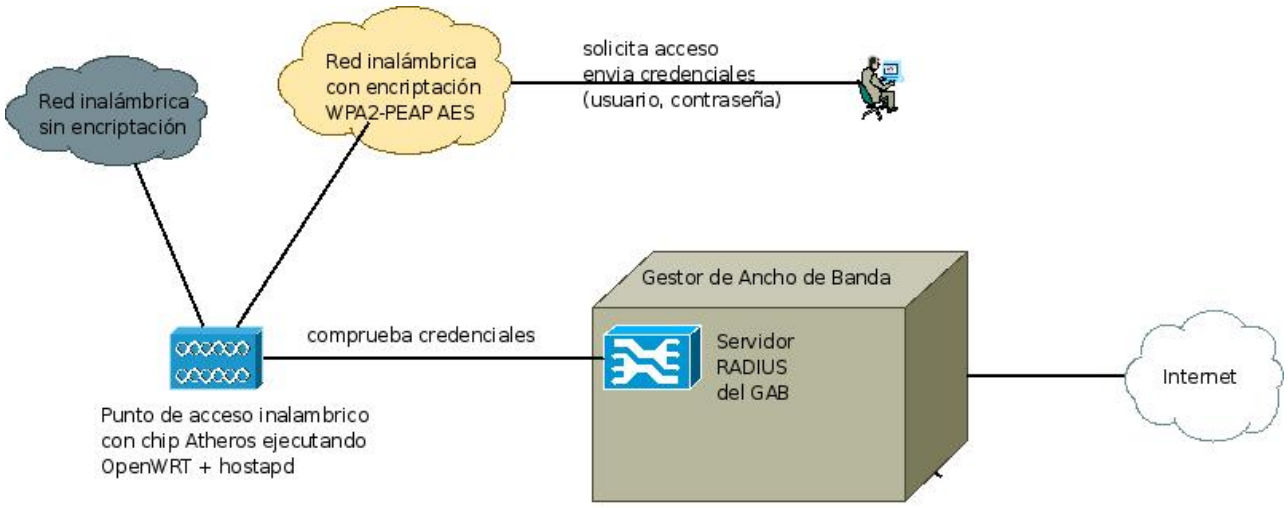
Características de WPA/IEEE 802.11i/EAP/IEEE 802.1X soportadas:

- WPA-PSK ("WPA-Personal")
- WPA con EAP (con el servidor EAP integrado o mediante un servidor RADIUS de autenticación externo) ("WPA-Enterprise")
- Manejo de claves para CCMP, TKIP, WEP104, WEP40
- WPA y soporte completo de IEEE 802.11i/RSN/WPA2
- RSN: cacheo de PMKSA, pre-autenticación

[56] http://hostap.epitest.fi/hostapd/
[57] Un demonio, daemon o daemon (de sus siglas en inglés Disk And Execution MONitor), es un tipo especial de proceso informático que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario (es un proceso no interactivo).

- IEEE 802.11r
- IEEE 802.11w
- Wi-Fi Protected Setup (WPS)

En el presente proyecto se ha configurado HostAPd para el estándar conocido como "WPA-Enterprise", que consiste en encriptación WPA (TKIP o AES[58]) bajo autenticación PEAP (Protected EAP) contra un servidor de autenticación RADIUS alojado en el gestor de ancho de banda.



Esquema de la autenticación WPA2 y del despliegue de la red inalámbrica

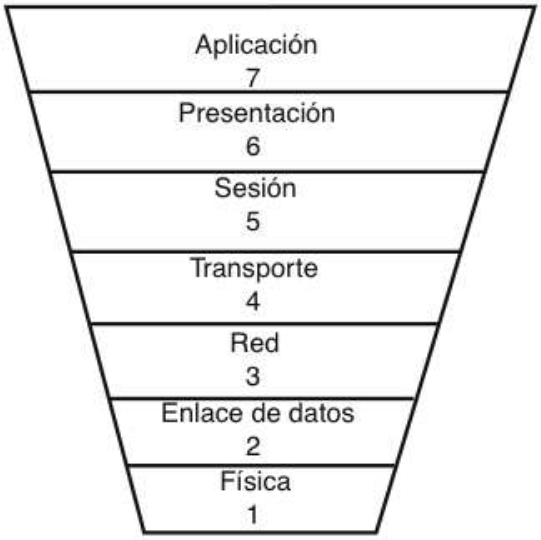
3.2.2.1.1. 802.1X y EAP

802.1X[59], es un mecanismo estándar para autenticar centralmente estaciones y usuarios. Es un estándar abierto que soporta diferentes algoritmos de encriptación. Se apoya en el protocolo de autenticación EAP (Extensible Authentication Protocol), aunque en realidad es EAPoL (EAP over LAN) de forma que se puede usar en redes ethernet, 802.11, Token-Ring y FDDI.

[58] La encriptación AES es mucho más robusta y segura que TKIP, pero presenta el inconveniente de que el hardware inalámbrico viejo puede no soportarlo adecuadamente debido a que se requiere un chip especializado para tal fin en la controladora MAC de la tarjeta inalámbrica.
No obstante, se ha optado por utilizar AES debido a que se han descubierto recientemente diversos fallos de seguridad en el algoritmo TKIP y la mayoría del hardware inalámbrico disponible hoy en día soporta el algoritmo AES.
http://jwis2009.nsysu.edu.tw/location/paper/A%20Practical%20Message%20Falsification%20Attack%20on%20WPA.pdf

[59] http://ditec.um.es/~ocanovas/papers/802.1X.pdf

Su uso requiere de un cliente (Xsupplicant en Linux), un punto de Acceso y un servidor de autenticación RADIUS. EAP es soportado por muchos Puntos de Acceso y por HostAPd. Antes de la autenticación sólo se permite tráfico 802.1X (petición de autenticación).



802.1X funciona en la capa 2

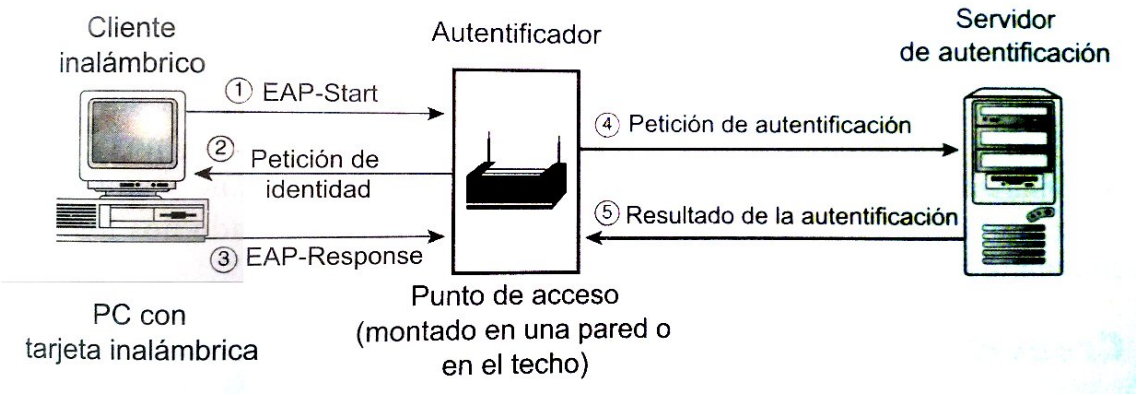
El Protocolo de Autentificación Extensible (EAP) es un protocolo general para la autenticación que soporta múltiples mecanismos. EAP no selecciona un mecanismo de autenticación específico en la fase de Control de Enlace, sino que pospone esta decisión hasta la fase de Autenticación. Esto permite que el Autenticador solicite más información antes de determinar el mecanismo de autenticación específico. También permite el uso de un servidor de soporte que implemente los diversos mecanismos (RADIUS por ejemplo) mientras que el Autenticador PPP se limita a transportar el intercambio de autenticación.

Después de que se haya establecido el enlace, se realiza la autenticación de EAP de la siguiente forma:

- Inicialmente el Autenticador envía peticiones para autenticar al par. La petición tiene un campo de tipo para indicar que es lo que se solicita. Algunos ejemplos de tipos de petición son la identidad, el desafió MD5, contraseñas de un solo uso, tarjetas de elementos genéricos y elementos del estilo. El autenticador enviará una petición de identidad inicial seguida de una o más peticiones de información de autenticación.

- Mas tarde, el par envía una respuesta correspondiente a cada petición. Al igual que el paquete de petición, el paquete de respuesta contiene un campo tipo que se corresponde con el campo del tipo del paquete de petición.

- El autenticador termina el proceso de autenticación con un paquete de éxito o fallo



EAP tiene muchas ventajas, incluyendo soporte de múltiples métodos de autenticación sin tener que establecer un mecanismo en particular durante la fase de Control de Enlace. Además, el punto de acceso no necesita comprender cada uno de los tipos de petición y puede actuar simplemente como un agente de redirección para un servidor RADIUS de soporte. Así el dispositivo sólo necesita controlar las respuestas de éxito o fallo para determinar el resultado del proceso de autenticación [C2 VLA].

En el siguiente proyecto se utilizará una variante de EAP, conocida como **PEAP** (*Protected Extensible Authentication Protocol, Protected EAP, or simplemente PEAP*). PEAP es un método para transmitir información de forma segura, incluyendo las contraseñas, sobre redes cableadas o inalámbricas. Fue desarrollada conjuntamente por Cisco Systems, Microsoft, y la Agencia de Seguridad RSA. PEAP no es un protocolo de encriptación, si no un método de autenticar clientes en una red. Podemos resumir el funcionamiento de PEAP de la siguiente forma.

- El servidor envía su llave pública SSL al cliente.
- El cliente cifra las comunicaciones con dicha llave pública del servidor.
- El servidor las cifra con su llave privada de forma que se establece entre ambos un túnel encriptado[60].
- El cliente envía sus datos de autenticación (nombre de usuario y contraseña) a través del túnel encriptado.

[60] http://es.wikipedia.org/wiki/Criptografía_asimétrica

- El servidor comprueba los datos (generalmente los envía a un servidor RADIUS para que realice dicha comprobación).
- Si los datos son correctos el servidor establece la conexión a través del túnel encriptado, si no lo son el cliente es desconectado.

La principal ventaja de PEAP es que no requiere instalación de certificados en el cliente ni ninguna configuración adicional, motivo por el cual lo hacen especialmente indicado para el presente proyecto

3.3. PoPToP

Tras haber visto como se ha configurado el sistema para permitir el acceso encriptado mediante WPA, estudiaremos el componente que permite crear la pasarela VPN, también encriptada.

PoPToP es el servidor PPTP OpenSource más conocido, es una de las mejores soluciones existentes para sistemas Linux y UNIX.

PPTP es un protocolo de Túneles Punto a Punto (Point-to-Point Tunneling Protocol) propietario desarrollado por Microsoft, pensado para comunicaciones del estilo VPN. Ofrece autenticación de usuarios utilizando protocolos como MS-CHAP, CHAP, SPAP y PAP. Su principal ventaja consiste en su facilidad de uso, sobretodo en clientes Windows, donde se puede configurar una VPN bajo este protocolo con muy pocos "clicks". Este ha sido el principal motivo de su elección como protocolo para la implementación VPN. Otros protocolos como IPSec ofrecen mejores características de seguridad y rendimiento, pero su configuración resulta algo compleja para el usuario novel, motivo por el cual se ha optado por la solución más sencilla dado que el escenario sobre el que trabajamos requiere que todo funcione con la mínima configuración por parte del cliente, al más puro estilo "Plug&Play", y dado que la mayoría de los clientes usarán con toda seguridad alguna variante de Windows la elección de PPTP como protocolo para la VPN parece razonable.

El servidor PoPToP aceptará peticiones PPTP y validará las credenciales del usuario (nombre y contraseña) contra el servidor RADIUS instalado en el gestor de ancho de banda. Si la autenticación de las credenciales es correcta se crea un túnel punto a punto entre el cliente y el

gestor de ancho de banda encriptado con el algoritmo MPPE[61] (*Microsoft Point-to-Point Encryption*), basado en RSA RC4.

Dicha encriptación no es tan robusta como la proporcionada por AES[62] (WPA2), no obstante nada impide que un determinado usuario se conecte al gestor de ancho de banda mediante WPA2 y luego cree una VPN contra el gestor, de forma que conseguiría un acceso doblemente encriptación (MPPE+DES).

3.4. FreeRADIUS

FreeRADIUS es un servidor de autenticación basado en el protocolo RADIUS. Se define en su documentación como el servidor RADIUS mas popular, en términos de números de usuarios que se autentican con el. FreeRADIUS es escalable, pudiendo funcionar en dispositivos embebidos (como pequeños routers inalámbricos) hasta sistemas con millones de usuarios. Soporta diversos métodos y protocolos de autenticación.

El diseño de FreeRADIUS es modular, diversos módulos gestionan los métodos de autenticación de usuarios. Por ejemplo, hay módulos para gestionar las bases de datos LDAP o la autenticación a través de EAP (con soporte para PEAP y EAP-TTLS).

En el presente proyecto el servidor FreeRADIUS se encargará de autenticar y validar el acceso de los usuarios comprobando sus credenciales contra una base de datos LDAP.

3.4.1. RADIUS

RADIUS (acrónimo en inglés de Remote Authentication Dial-In User Server). Es un protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP. Utiliza el puerto 1813 UDP para establecer sus conexiones.

Cuando se realiza la conexión con un Proveedor de Internet mediante módem, ADSL, cable módem, Ethernet o Wi-Fi, se envía una información que generalmente es un nombre de usuario y una contraseña. Esta

[61] http://www.networksorcery.com/enp/protocol/mppe.htm
[62] http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

información se transfiere a un dispositivo NAS (Servidor de Acceso a la Red o Network Access Server (NAS)) sobre el protocolo PPP, quien redirige la petición a un servidor RADIUS sobre el protocolo RADIUS. El servidor RADIUS comprueba que la información es correcta utilizando esquemas de autenticación como PAP, CHAP o EAP. Si es aceptado, el servidor autorizará el acceso al sistema del Proveedor de Internet y le asigna los recursos de red como una dirección IP, y otros parámetros.

La estructura fundamental detrás del servicio RADIUS se conoce como AAA (Autenticación, Autorización y Administración de uso). Dicha estructura puede interpretarse como una estructura para el control de acceso a recursos informáticos, la imposición de políticas, el análisis de recursos y la obtención de información necesaria para cobrar por este servicio. Estos procesos se consideran vitales para la administración eficaz de redes y la imposición de medidas de seguridad.

3.4.1.1. Autenticación

La autenticación (o autentificación[63]) es el proceso que proporciona un método para identificar a los usuarios mediante la petición y comparación de un conjunto válido de credenciales. La autenticación se basa en los criterios únicos que posee cada usuario para conseguir el acceso. El servidor conforme con AAA compara las referencias de autenticación del usuario con información almacenada en una base de datos. Si las credenciales se corresponden, se le permite el acceso a los recursos de red solicitados; de no ser así, el proceso de autenticación falla y se niega el acceso a la red.

3.4.1.2. Autorización

La autorización va después de la autentificación y es el proceso por el cual se determina si el usuario puede solicitar o utilizar ciertas tareas, recursos de red u operaciones. Normalmente la autorización se produce en el contexto de la autentificación y, una vez que se aprueba al usuario, este puede pasar a utilizar los recursos solicitados. Por ello, la autorización es un aspecto vital para la sana administración de una política de acceso.

[63] Autenticar. ‘Certificar la autenticidad [de algo, especialmente un documento]’: «Se requiere un documento sellado y autenticado con todas las de la ley». Este verbo se documenta ya en el español medieval y sigue plenamente vigente en el lenguaje legal y administrativo, especialmente en América. Con este mismo sentido se ha creado modernamente el verbo autentificar, que se considera también válido y es el usado con preferencia en el español de España y de buena parte de América: «Si el cuadro se compra a un particular, debe autentificarlo un experto» [http://buscon.rae.es/]

3.4.1.3. Administración de uso

El aspecto final de la estructura AAA es la “contabilidad”, que se describe mejor como el proceso de medida y grabación del consumo de los recursos de red. Esto permite la monitorización y la generación de informes sobre eventos y su uso para distintos propósitos, incluidos la presentación de facturas, el análisis de tendencias, el uso de recursos, la planificación de capacidad y el mantenimiento activo de la política.

3.5. OpenLDAP

OpenLDAP es una implementación libre y de código abierto del protocolo *Lightweight Directory Access Protocol* (LDAP) desarrollada por el proyecto OpenLDAP. Está liberada bajo su propia licencia *OpenLDAP Public License*[64]

3.5.1. Servicio de directorio

Un directorio es una estructura de base de datos que suele optimizarse para leer, buscar y recorrer las entradas. Los directorios suelen contener información basada en atributos y una descripción, y suelen disponer de capacidades de filtrado para permitir una obtención más rápida y precisa de los resultados de las búsquedas. Los directorios deberían ajustarse para proporcionar una rápida respuesta a búsquedas muy habituales. Puede que tengan la capacidad de duplicar información entre servidores similares para incrementar la disponibilidad y fiabilidad del servicio ofrecido. Cuando se replica la información de la base de datos entre servidores, pueden producirse inconsistencias temporales entre las réplicas, que deberían sincronizarse en muy poco tiempo para conservar la fiabilidad de la información.

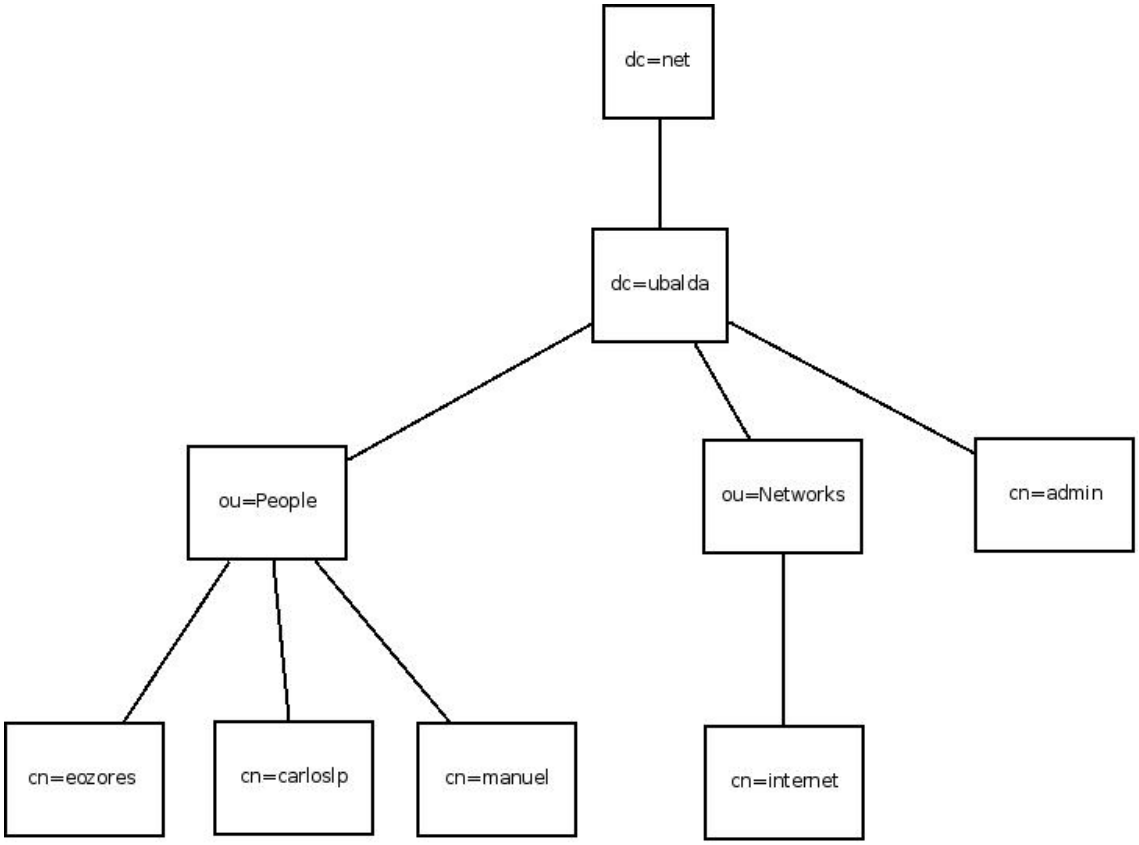
[64] http://www.openldap.org/software/release/license.html

3.5.2. LDAP

LDAP son las siglas de *Lightweight Directory Access Protocol (Protocolo de Acceso Ligero a Directorio)*. LDAP se ha venido utilizando durante bastante tiempo y se ha vuelto muy popular dentro de la comunidad de las redes. Existen muchas razones para esta situación, como el soporte de desarrollo cada vez mayor en forma de software procedente de la comunidad opensource y de organizaciones de software propietario.

El modelo de información de LDAP contiene un cierto número de registros individuales conocidos como entradas, que representan un conjunto de atributos que tienen un nombre distintivo (DN) globalmente único. Cada uno de los atributos de la entrada tiene un tipo y uno o más valores. Los tipos son habitualmente cadenas, como ***uid*** para la identificación de usuario, ***cn*** para el nombre común, ***sn*** para el apellido o ***mail*** para la dirección de correo electrónico. La sintaxis de los valores depende del tipo de atributo. Por ejemplo, un atributo ***cn*** puede contener el valor *carloslp*, y el atributo ***mail*** podía contener el valor carloslp@ubalda.com

En LDAP, las entradas de directorio se organizan de acuerdo con una estructura jerárquica en árbol. La siguiente figura muestra la estructura del directorio de la base de datos implementada en este proyecto:



Estructura del directorio LDAP implementado en el presente proyecto

Tradicionalmente, esta estructura refleja divisiones geográficas o de organización. Las entradas que representan países aparecen en lo más alto del árbol y bajo ellas se encuentran las entradas que representan a las organizaciones nacionales. Por debajo podría haber entradas que representan a las unidades de la organización, a gente, a impresoras, documentos o cualquier otra cosa que pueda imaginarse.

Alternativamente, la estructura del directorio puede basarse en nombres de dominio, algo que cada vez es más popular. Como puede ver en la figura anterior, la estructura del directorio de la organización se basa en el nombre del dominio (*dc=ubalda,dc=net*).

Una entrada se referencia en la estructura de directorio mediante su ***DN***, que se forma tomando el nombre de la propia entrada y añadiéndole los nombres de sus entradas paternas, de sus antecesores. Por eso, para acceder a la entrada de carloslp del ejemplo anterior deberíamos utilizar *cn=carloslp,ou=People,dc=ubalda,dc=net*

3.6. Apache

Apache es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual (*virtual host*[65]). Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que Behelendorf eligió ese nombre porque quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de Internet. El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años.

Apache es el servidor web más usado en sistemas Linux. Los servidores web se usan para servir páginas web solicitadas por equipos cliente. Los clientes normalmente solicitan y muestran páginas web mediante el uso de navegadores web como Firefox, Opera o Internet Explorer.

Los usuarios introducen un Localizador de Recursos Uniforme (*Uniform Resource Locator,* ***URL***) para señalar a un servidor web por medio de su Nombre de Dominio Totalmente Cualificado (**Fully Qualified Domain Name,** ***FQDN***) y de una ruta al recurso solicitado. Por ejemplo, para ver la página web del sitio web de Ubuntu, un usuario debería introducir únicamente el FQDN (*ubuntu.com*). Para solicitar información específica acerca de dicho sitio web, un usuario deberá introducir el FQDN seguido de una ruta.

[65] Gracias a los virtual hosts se pueden tener diferentes sitios web alojados en un mismo servidor. Por ejemplo, los dominios *ubalda.com gestordeanchodebanda.es* y *conchouso.net* pueden estar alojados en el mismo servidor (la misma IP para todos ellos) y es Apache quien se encarga de mostrar uno u otro sitio según la información que envía el navegador Web en las cabeceras HTTP.
http://httpd.apache.org/docs/2.0/es/vhosts/name-based.html

El protocolo más comúnmente utilizado para ver páginas Web es el Hyper Text Transfer Protocol (HTTP). Protocolos como el Hyper Text Transfer Protocol sobre Secure Sockets Layer (HTTPS), y File Transfer Protocol (FTP), un protocolo para subir y descargar archivos, también son soportados.

Los servidores web Apache a menudo se usan en combinación con el motor de bases de datos MySQL, el lenguaje de scripting PHP, y otros lenguajes de scripting populares como Python y Perl. Esta configuración se denomina ***LAMP*** (*Linux, Apache, MySQL y Perl/Python/PHP*) y conforma una potente y robusta plataforma para el desarrollo y distribución de aplicaciones basadas en la web.

3.7. Bind

BIND es un software open-source que implementa el protocolo de traducción de nombres (dominios) a direcciones IP conocido como DNS (acrónimo en ingles de Domain Name System). El protocolo DNS es parte de los protocolos "estándar" de Internet y es soportado por todos los sistemas operativos y fabricantes.

BIND (Berkeley Internet Name Domain, es el servidor de DNS más comúnmente usado en Internet, especialmente en sistemas Unix, en los cuales es un Estándar de facto. Es patrocinado por la Internet Systems Consortium. BIND fue creado originalmente por cuatro estudiantes de grado en la University of California, Berkeley y liberado por primera vez en el 4.3BSD. Paul Vixie comenzó a mantenerlo en 1988 mientras trabajaba para la DEC.

La nueva versión de BIND (BIND 9) fue escrita desde cero en parte para superar las dificultades arquitectónicas presentes anteriormente para auditar el código en las primeras versiones de BIND, y también para incorporar DNSSEC (DNS Security Extensions[66]). BIND 9 incluye entre otras características importantes: TSIG, notificación DNS, nsupdate, IPv6, rndc flush, vistas, procesamiento en paralelo, y una arquitectura mejorada en cuanto a portabilidad. Es comúnmente usado en sistemas Linux.

[66] http://www.dnssec.net/

El presente proyecto implementa su propio servidor DNS gracias al uso de BIND9. Esta característica, la incorporación al sistema de un servidor DNS, le permite dos características esenciales:

- **Tolerancia a fallos**. La estructura de funcionamiento del protocolo DNS es de tipo árbol, donde existe un servidor central para cada TLD (.es,.com,.net,org…) que replica su información en servidores secundarios, de forma que se consigue que en caso de avería del servidor central para dicho TLD el sistema no deje de funcionar, y de paso se consigue evitar "cuellos de botella" ya que se puede consultar cualquiera de esos servidores secundarios. El servidor DNS incluido en el gestor de ancho de banda está configurado para obtener y replicar en caché dicha información de los servidores raiz (*root servers*) del protocolo DNS[67], de forma que no depende del servidor DNS del proveedor a Internet contratado, consiguiendo efectivamente seguir dando servicio aun en el hipotético caso de que el servidor DNS del proveedor a Internet dejase de funcionar.

- **Resolución de dominios no existentes**. Cuando el usuario teclea un dominio que no existe (por ejemplo gogle.esx) el sistema resuelve dicho nombre a su propia IP, de forma que el usuario accede al gestor de ancho de banda donde se le muestra un aviso de que la dirección tecleada no existe y se le muestra una barra de búsqueda en Google.

3.8. dhcpd

DHCPD es el servidor DHCP por excelencia de Linux, se encarga de atender las peticiones DHCP y asignar direcciones IP a los clientes. Es una parte básica del funcionamiento del presente proyecto ya que permite a los usuarios que se conectan obtener una IP para su posterior conexión a Internet de forma automática

DHCP significa Protocolo de configuración de host dinámico (*dynamic host configuration protocol)*. Es un protocolo que permite que un equipo conectado a una red pueda obtener su configuración (principalmente, su configuración de red) en forma dinámica (es decir, sin intervención

[67] http://www.root-servers.org/

particular). El servidor DHCP se encarga de asignarle al equipo que lo solicita una dirección IP de manera automática. El objetivo principal es simplificar la administración de la red.

3.8.1. Funcionamiento del protocolo DHCP

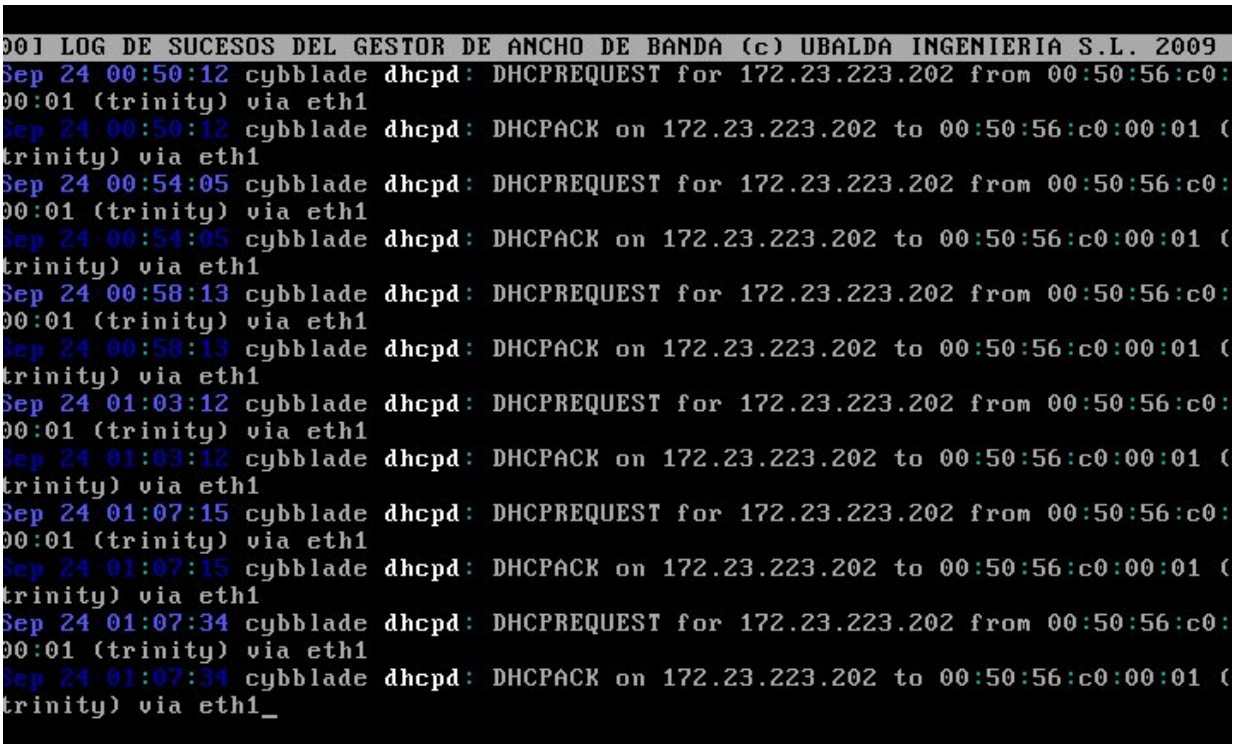
Primero, se necesita un servidor DHCP que distribuya las direcciones IP. Este equipo será la base para todas las solicitudes DHCP por lo cual debe tener una dirección IP fija. Por lo tanto, en una red puede tener sólo un equipo con una dirección IP fija: el servidor DHCP.

El sistema básico de comunicación es BOOTP (con la trama UDP). Cuando un equipo se inicia no tiene información sobre su configuración de red y no hay nada especial que el usuario deba hacer para obtener una dirección IP. Para esto, la técnica que se usa es la transmisión: para encontrar y comunicarse con un servidor DHCP, el equipo simplemente enviará un paquete especial de transmisión (transmisión en 255.255.255.255 con información adicional como el tipo de solicitud, los puertos de conexión, etc.) a través de la red local. Cuando el DHCP recibe el paquete de transmisión, contestará con otro paquete de transmisión (no olvide que el cliente no tiene una dirección IP y, por lo tanto, no es posible conectar directamente con él) que contiene toda la información solicitada por el cliente.

Se podría suponer que un único paquete es suficiente para que el protocolo funcione. En realidad, hay varios tipos de paquetes DHCP que pueden emitirse tanto desde el cliente hacia el servidor o servidores, como desde los servidores hacia un cliente:

- DHCPDISCOVER (para ubicar servidores DHCP disponibles)
- DHCPOFFER (respuesta del servidor a un paquete
- DHCPDISCOVER, que contiene los parámetros iniciales)
- DHCPREQUEST (solicitudes varias del cliente, por ejemplo, para extender su concesión)
- DHCPACK (respuesta del servidor que contiene los parámetros y la dirección IP del cliente)
- DHCPNAK (respuesta del servidor para indicarle al cliente que su concesión ha vencido o si el cliente anuncia una configuración de red errónea)

- DHCPDECLINE (el cliente le anuncia al servidor que la dirección ya está en uso)
- DHCPRELEASE (el cliente libera su dirección IP)
- DHCPINFORM (el cliente solicita parámetros locales, ya tiene su dirección IP)



El primer paquete emitido por el cliente es un paquete del tipo DHCPDISCOVER. El servidor responde con un paquete DHCPOFFER, fundamentalmente para enviarle una dirección IP al cliente. El cliente establece su configuración y luego realiza un DHCPREQUEST para validar su dirección IP (una solicitud de transmisión ya que DHCPOFFER no contiene la dirección IP) El servidor simplemente responde con un DHCPACK con la dirección IP para confirmar la asignación. Normalmente, esto es suficiente para que el cliente obtenga una configuración de red efectiva, pero puede tardar más o menos en función de que el cliente acepte o no la dirección IP...

3.9. Linux UPnP Internet Gateway Device

El servidor UPnP IGD para Linux es el más completo de los servidores IGD-UPnP existentes para dicho sistema operativo.

UPnP (Universal Plug and Play) soporta el trabajo de una red sin configurar y automáticamente detecta cualquier dispositivo que puede ser incorporado a esta, obtiene su dirección IP, un nombre lógico, informando a los demás de sus funciones y capacidad de procesamiento, y le informa, a su vez, de las funciones y prestaciones de los demás.

IGD UPnP, es una parte del protocolo UPnP que se utiliza para *"natear"* puertos de forma automática. En muchas situaciones un usuario conectado a Internet a través de un router que hace traducción de direcciones (NAT es acrónimo de *Network Adress Translation*) necesita que los puertos TCP o UDP de su PC sean accesibles desde el exterior, desde Internet. Tradicionalmente para hacer esto posible había que entrar en la configuración del router y definir y asignar los puertos a mano.

IGD UPnP soluciona este problema y realiza de forma automática y dinámica la asignación de puertos. Cuando un programa en el PC del usuario necesita que uno de los puertos TCP o UDP que tiene abiertos sea accesible desde Internet, por ejemplo para recibir una videoconferencia o para intercambiar archivos mediante algún protocolo P2P este envía una petición UPnP al router, si el router tiene soporte para IGD UPnP le asignará automáticamente un puerto accesible desde el exterior.

Aplicaciones del tipo P2P, juegos en red, asistencia remota se benefician de este protocolo. Sin IGD el usuario tiene que configurar manualmente el router para permitir el tráfico entrante hacia determinados puertos, un proceso que puede llegar a ser complejo. Gracias al uso del protocolo IGD de UPnP esto se automatiza. IGD permite:

- Notificar cual es la IP externa en Internet
- Enumerar las redirecciones de puertos existentes
- Añadir o eliminar redirecciones
- Reservar una determinada redirección durante cierto tiempo

Por resumirlo en una frase, IDG UPnP facilita la configuración automática de la redirección de puertos de la misma forma que DHCP facilita la configuración automática de direcciones IP.

3.10. Herramientas de Linux para el control de tráfico.

El control del tráfico de red, es una característica que ha estado presente en el kernel de Linux desde las versiones 2.2. Es quizá el producto más completo disponible, y ofrece una gran variedad de métodos de planificación de paquetes y moldeado. Linux incorpora las herramientas necesarias para gestionar el ancho de banda en formas comparables a los sistemas dedicados de alto nivel.

Lamentablemente el uso de estas herramientas que Linux incorpora es realmente complejo y nada intuitivo, es por ello que se hace necesario aplicaciones de terceros que gestionen estas herramientas para construir un sistema. El presente proyecto tiene esa finalidad, construir un sistema sencillo de usar y administrar que sea capaz de sacar todo el rendimiento que Linux puede ofrecer.

3.10.1. iproute2

Iproute2 es una colección de utilidades que reemplazan los viejos comandos de Linux que permitían controlar la configuración TCP/IP. Su sintaxis trata de imitar la del Sistema Operativo de Red CISCO IOS[68].

Estas son las utilidades que reemplaza y su función:

* Dirección y configuración del enlace: ifconfig → ip addr, ip link
* Tablas de rutas: route → ip route
* Vecinos: arp → ip neigh
* Túneles: iptunnel → ip tunnel
* Multicast: ipmaddr → ip maddr
* Estadísticas de red: netstat → ss

La funcionalidad de Iproute2 va mucho más allá, por ejemplo la utilidad *tc* permite implementar disciplinas de encolado de paquetes, priorización y QoS realmente impresionantes. Tales son sus capacidades que merece que le dediquemos una sección aparte.

[68] http://en.wikipedia.org/wiki/Cisco_IOS

3.10.1.1. TC – QoS[69] y Algoritmos de moldeado de tráfico.

Tc es la útilidad útilizada para controlar el tráfico de red en el kernel de Linux. Dicho control de tráfico se divide en las siguientes posibilidades:

➔ SHAPING (MODELADO)

Cuando el tráfico es moldeado, se mantiene bajo control su tasa de transmisión. El moldeado de tráfico es más que disminuir el ancho de banda disponible - también se usa para limitar las ráfagas de paquetes para un mejor comportamiento de la red entre otras cosas.

➔ SCHEDULING (PLANIFICACIÓN)

Mediante la planificación de la transmisión de paquetes es posible mejorar la interactividad para el tráfico que lo necesite mientras se garantiza el resto del ancho de banda para las grandes transferencias de datos que no necesitan necesariamente de esa interactividad o respuesta instantánea.

➔ POLICING (VIGILANCIA)

Por defecto el kernel de Linux solo permite aplicar el moldeado y la planificación al tráfico saliente pero no al entrante. Mediante el *policing* se pueden aplicar algunas reglas al tráfico entrante aunque no son tan potentes como las de *shapping* y *scheduling*. Mas adelante hablare de ello y de cómo se puede solucionar esto.

➔ DROPPING (DESCARTAR)

El tráfico que excede determinado ancho de banda puede ser directamente descartado.

El procesamiento del tráfico es controlado por tres tipos de objetos: qdiscs, clases y filtros.

[69] QoS: Acrónimo de Quality of Service (calidad de servicio)

➲ Qdiscs

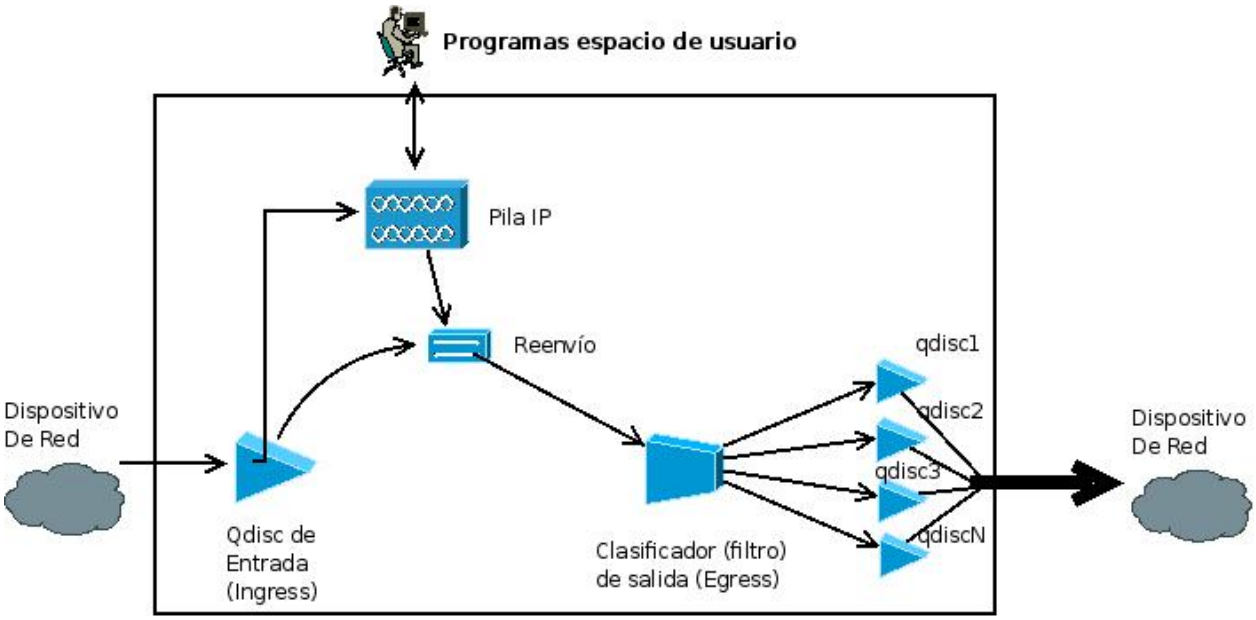
Qdisc es un acrónimo de *'queueing discipline'* (disciplina de cola) y es el componente básico del control del tráfico en Linux. Cada vez que el kernel de Linux necesita enviar un paquete a la red, este es primero encolado en la qdisc para ello definida. Inmediatamente después el kernel intenta desencolar los paquetes de la qdisc para enviarlos a la red a través del adaptador de red. Existen diferentes tipos de qdisc cada una con un algoritmo de procesado diferente, la mas sencilla es la denominada *'pfifo'*, que no hace ningún tipo de procesamiento, simplemente aplica la regla FIFO (el primero en entrar es el primero en salir), por lo que es una sencilla forma de almacenar el tráfico cuando el enlace está saturado y no puede manejarlo de momento.

➲ Clases

Algunas Qdiscs pueden contener clases, las cuales a su vez pueden contener otras qdiscs que están dentro de las clases. De esta forma cuando el kernel decide desencolar un paquete para enviarlo a la red desde una qdisc con clases este puede venir de cualquiera de las clases. Una qdisc puede por ejemplo priorizar cierto tipo de tráfico intentando desencolar primero de cierto tipo de clases en vez de otras de más baja prioridad

➲ Filtros

Un filtro es usado por una qdisc con clases para determinar en que clase un determinado paquete debe ser encolado. Cada vez que llega tráfico a una qdisc con clases este necesita ser clasificado para decidir en que clase se encolará. Esto puede ser programado mediante diversos métodos, uno de ellos es el uso de los filtros. Cuando un paquete va a ser encolado se comprueban todos los filtros, el primero que sea capaz de clasificar al paquete define en que clase se encolará. Si ningún filtro de los definidos es capaz de clasificar al paquete entonces se encolará en la clase por defecto.



El gran bloque representa al núcleo. La flecha de la izquierda es el tráfico entrando en la máquina desde la red. Entonces se le pasa a la Qdisc de Entrada que puede aplicar filtros a los paquetes, y decidir descartarlos. A esto se le llama «Policing». Esto ocurre en una etapa muy temprana, antes de que se haya visto mucho del núcleo. Por tanto, es un buen lugar para descartar tráfico sin consumir mucho tiempo de CPU. Si se le permite continuar al paquete, puede estar destinado a una aplicación local, en cuyo caso entra en la pila IP para ser procesado, y se envía a un programa en espacio de usuario. El paquete también puede ser reenviado sin pasar por una aplicación (el sistema puede estar enrutando paquetes), en cuyo caso se destina a la salida (egress). Los programas de espacio de usuario también pueden distribuir datos, que serán examinados y reenviados por el Clasificador de Salida. Ahí es observado y encolado por cualquiera de varias qdisc. En el caso por defecto, sin configurar, sólo hay instalada una qdisc de salida, pfifo_fast, que siempre recibe el paquete. A esto se le llama «encolado» (enqueueing). Ahora el paquete está en la qdisc, esperando a que el núcleo pida que sea retransmitido por la interfaz de salida. A esto se le llama «desencolado».

Esta figura también funciona en el caso de que haya un único adaptador de red (las flechas de entrada y salida al núcleo no se deben tomar demasiado literalmente). Cada adaptador de red tiene ranuras tanto para entrada como para salida.

3.10.1.1.1. Disciplinas de cola (qdiscs) y algoritmos de moldeado de tráfico

En las siguientes páginas se describen los algoritmos de “traffic shapping”, habitualmente traducido como moldeado o perfilado de tráfico, y planificación de tráfico (“traffic sheduling”) que implementa el núcleo de Linux. Según la qdisc que asignemos a un dispositivo de red se aplicará uno u otro algoritmo.

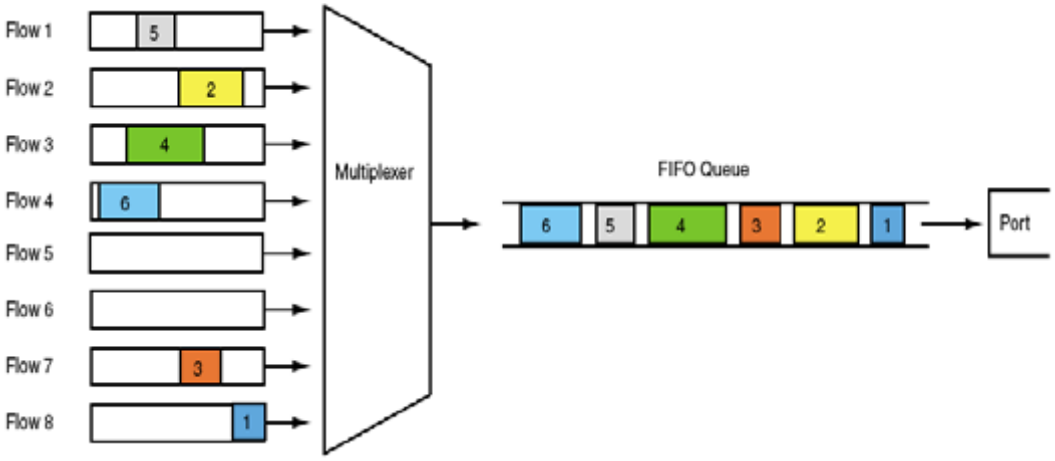
Debemos distinguir entre disciplinas de cola (qdiscs) con clases o sin clases, en las primeras podremos insertar disciplinas de cola secundarias formando un árbol tan complejo como deseemos. Las qdiscs sin clases no permiten esto por lo que serán únicas para cada dispositivo de red. No obstante podemos utilizar una qdisc con clases de forma que los nodos hoja del árbol que formemos sean qdiscs sin clases.

- Tipos de Disciplinas de Cola (Qdiscs)
  - Sin clases
    - Pfifo
    - RED
    - SFQ
    - TBF
  - Con clases
    - PRIO
    - CBQ
    - HTB

3.10.1.1.1.1. Pfifo

Primero en entrar, primero en salir (FIFO), es la estrategia más simple de planificación de tráfico. Los paquetes se encolan según van llegando al dispositivo y se desencolan según pueden ser enviados a la red, por tanto no se consideran prioridades para los paquetes dentro de estas colas

Las colas FIFO poseen varias limitaciones (descritas en detalle por Chuck Semeria[70]), no siendo adecuadas para el tráfico a ráfagas, ante el cual producen un in cremento de la latencia.

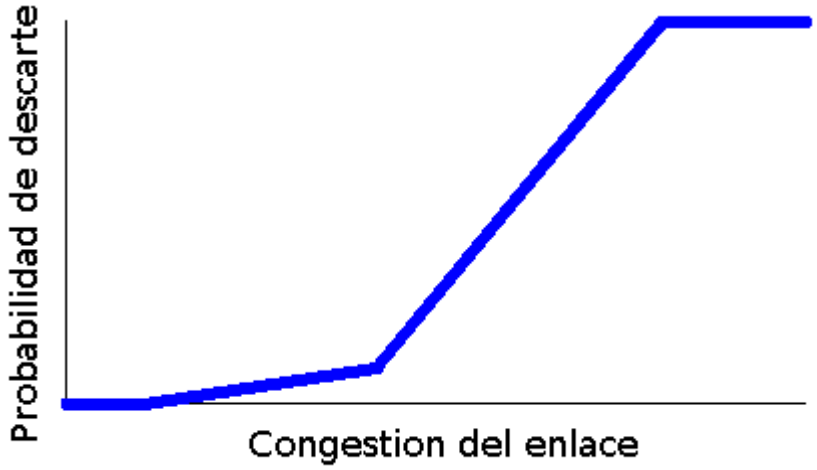


Cabe destacar que FIFO es el algoritmo utilizado por defecto por Linux y la mayoría de los routers y es una buena elección cuando la CPU puede no tener capacidad suficiente para procesar todos los paquetes que pasan a través del dispositivo.

3.10.1.1.1.2. RED (Random Early Detection)

La detección temprana aleatoria, tal y como describe Floyd[71], trata de evitar la congestión en la puerta de enlace. Lo consigue descartando paquetes cuando la congestión del enlace es inminente, o sea, cuando el tráfico se acerca al límite de ancho de banda establecido en la qdisc. Esto permite que protocolos como TCP ajusten la velocidad del tráfico antes de que la congestión sea excesiva. La selección de los paquetes a descartar es aleatoria, para evitar perjudicar a un flujo concreto de tráfico. Algunas variantes utilizan pesos para decidir qué tipos de tráfico tiene más probabilidad de sufrir descarte de paquetes.

[70] http://marco.uminho.pt/disciplinas/ST/packethandling.pdf
[71] http://www.icir.org/floyd/papers/early.pdf



A diferencia de las colas FIFO, no se produce un incremento en la latencia debido a que se descartan paquetes antes de que se congestione el enlace, pero la facilidad con que se descartan los paquetes hace poco adecuado su uso con ciertos tipos de tráfico (protocolos sin retransmisiones como UDP).

3.10.1.1.1.3. SFQ (Stochastic Fairness Queueing)

El Encolado Justo *(Fair queuing)* (FQ) fue propuesto por John Nagle en 1987. Su funcionamiento es la de un conjunto de colas de encolado que están diseñadas para asegurar que cada flujo de datos tiene un acceso a la red justo de forma que se previene que un determinado flujo masivo de datos consuma mas ancho de banda que los otros flujos de datos. Los paquetes se clasifican en flujos de datos por el sistema y luego son asignados a las colas específicamente dedicadas a dichos flujos.

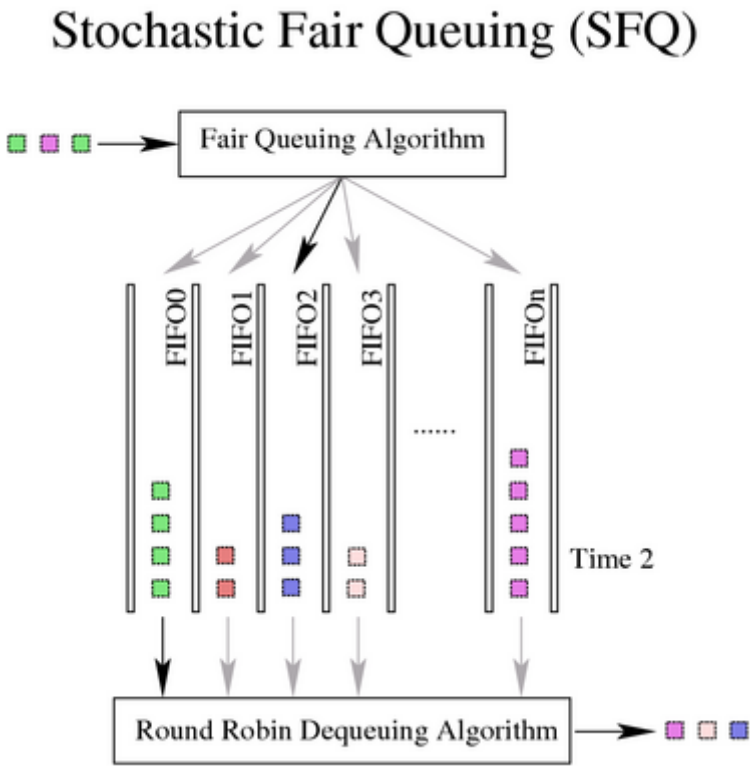
Las colas de flujos sirven entonces un paquete por petición en un esquema round-robin. Las colas vacías son ignoradas. Está basado en el concepto de Round Robin y su la idea central es la siguiente:

- El enrutador posee múltiples colas (FIFO) asignadas al enlace: una por cada usuario (o flujo).

- Cuando el enlace está disponible (se pueden transmitir más datos) el enrutador examina las colas cíclicamente (Round Robin), extrayendo el primer paquete de cada cola (de aquella que tiene el turno).

- Con n usuarios compitiendo por la misma línea se garantiza que cada usuario (o flujo) pueda enviar al menos un paquete por cada n.

Por supuesto, la versión propuesta por Nagle tiene ciertas deficiencias, como no tener en cuenta la longitud de los paquetes; lo que implica que un flujo con paquetes bastante más grandes que los de otros, injustamente transmitirá mayor cantidad de datos con el mismo número de paquetes.



La versión estocástica, Stochastic Fairness Queueing (SFQ) [72] originalmente propuesta por McKenney y que es la que se implementa en el núcleo de Linux es una implementación sencilla de la familia de algoritmos de colas justas (fair queueing). Es menos preciso que los otros, pero también necesita menos cálculos mientras que resulta ser casi perfectamente justo. En lugar de asignar una cola para cada clase de tráfico, utiliza un algoritmo de hashing para dividir el tráfico entre un número limitado de colas. Esto supone que múltiples clases de tráfico (se correspondan éstas a sesiones, flujos o usuarios) podrían ser asignadas al mismo grupo; aunque el algoritmo ha sido desarrollado de forma que las colisiones sean mínimas.

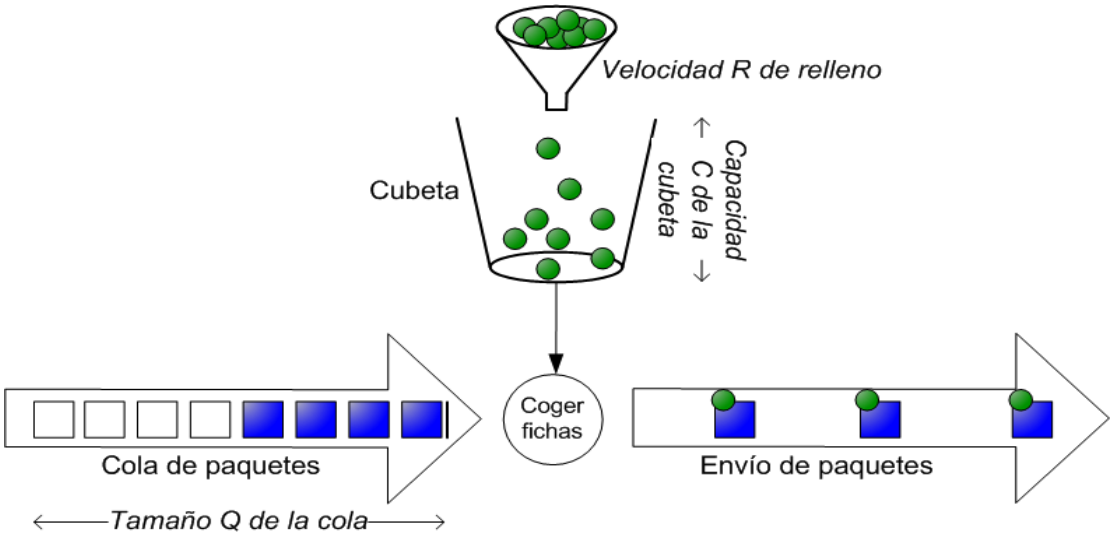
La palabra clave en SFQ es conversación (o flujo), que se corresponde en su mayoría a una sesión TCP o a un flujo UDP. El tráfico se divide en un número bastante grande de colas FIFO, una por cada conversación. Entonces se envía el tráfico de una manera parecida a round robin, dando a cada sesión por turnos la oportunidad de enviar datos.

[72] http://www.gulic.org/comos/LARTC

Esto lleva a un comportamiento bastante equitativo y evita que una única conversación ahogue a las demás. SFQ se llama «estocástica» porque realmente no crea una cola para cada sesión, sino que tiene un algoritmo que divide el tráfico en un número limitado de colas usando un algoritmo de hash (troceo). Debido al hash, varias sesiones pueden acabar en la misma cola, lo que dividirá por dos las posibilidades de cada sesión de enviar un paquete, reduciendo a la mitad de esta forma la velocidad efectiva disponible. Para evitar que esta situación acabe siendo detectable, SFQ cambia a menudo su algoritmo hash de manera que dos sesiones sólo colisionen durante unos pocos segundos.
Es importante tener en cuenta que SFQ sólo es útil en caso de que la interfaz real de salida esté realmente llena. Si no lo está, entonces la máquina Linux no encolará paquetes y no se producirá efecto alguno.

3.10.1.1.1.4. TBF (Token Bucket Filter)

El Token Bucket Filter (TBF) (*filtro de la cubeta de fichas*) es una cola de encolado *qdisc* sencilla que se limita a dejar pasar los paquetes que llegan a una tasa que no exceda la que se ha definido previamente, pero con la posibilidad de permitir ráfagas cortas que excedan esta tasa. TBF es muy preciso, amigable para la red y el procesador. Sin duda es la mejor opción cuando sólo se pretende ralentizar un dispositivo de red.
La implementación de TBF consiste en una "*cubeta*" (*el bucket*), que se llena constantemente con "*fichas virtuales*" denominadas *tokens*, a una tasa o velocidad específica (*token rate*). El parámetro más importante del bucket (*la cubeta*) es su tamaño, es decir, el número "*fichas*" (*tokens*) que puede almacenar.



Su funcionamiento es muy intuitivo, los paquetes que llegan a la qdisc TBF, para poder pasar (ser transmitidos) han de retirar una ficha (token). Las fichas se almacenan en una cubeta (bucket) que se rellena a una velocidad constante.

El comportamiento ante la llegada de datos viene dado por una de las siguientes situaciones:

- Los datos llegan a TBF a una tasa que es igual a la de tokens (*fichas*) entrantes. En este caso, cada paquete entrante tiene su token correspondiente y pasa a la cola sin retrasos.

- Los datos llegan al TBF a una tasa menor a la de relleno de *fichas* (token). Sólo una parte de las fichas (tokens) son usados con el envio de cada paquete que sale fuera de la cola, de manera que se acumulan las fichas, hasta llenar la cubeta (bucket). Las fichas (tokens) sin usar se pueden utilizar en determinado momento para enviar datos a velocidades mayores de la tasa de relleno de fichas (tokens), en cuyo caso se produce una corta ráfaga de datos.

- Los datos llegan al TBF a una tasa mayor que la de relleno de fichas (tokens). Esto significa que la cubeta (bucket) se quedará pronto sin fichas (tokens), lo que causará que TBF se acelere a sí mismo por un rato. Esto se llama una «situación sobrelímite». Si siguen llegando paquetes, empezarán a ser descartados. Esta última situación es muy importante, porque permite ajustar administrativamente al ancho de banda disponible a los datos que están pasando por el filtro. La acumulación de fichas (tokens) permite ráfagas cortas de datos extralimitados para que pasen sin pérdidas, pero cualquier sobrecarga restante causará que los paquetes se vayan retrasando constantemente, y al final sean descartados.

Puede apreciarse que el algoritmo ofrece tres parámetros principales fáciles de interpretar:

  - **R**, normalmente denominado *Rate*, regula la tasa de relleno de la cubeta, y permite ajustar administrativamente el ancho de banda (la velocidad media) disponible a los datos que pasan el filtro.

  - **C**, la capacidad de la cubeta, llamada frecuentemente *Burst*, permite establecer el tamaño máximo de las ráfagas.

- **Q**, el tamaño de la cola o *Capacity*, especifica el tamaño de la cola de espera. Nota: el tamaño de la cola puede ser cero, indicando que los paquetes no-conformes han de ser descartados. Cabe destacar que este valor influye directamente en la latencia máxima para los paquetes en espera.

- Existe un parámetro adicional en la implementación de Linux denominado *peakrate* (tasa máxima instantánea): establece una velocidad máxima para el vaciado de la cubeta.

3.10.1.1.1.5. Colas de encolado (*qdiscs*) con clases

Hasta ahora hemos visto las colas de encolado sencillas (que no permiten contener clases y subcolas). Veremos ahora una introducción a las colas con clases antes de explicar en detalle cada una de ellas

Las colas de encolado (qdisc) con clases son muy útiles si tiene diferentes tipos de tráfico a los que quiere dar un tratamiento separado.

Cuando entra tráfico dentro de una cola de encolado (qdisc) con clases, hay que enviarlo a alguna de las clases que contiene (para ello se necesita *«clasificarlo»*). Para determinar qué hay que hacer con un paquete, se consulta a los «*filtros*».

Los filtros asociados a esa *qdisc* devuelven entonces una decisión, y la *qdisc* la usa para encolar el paquete en una de su clases asociada. Cada clase puede contener subclases y puede probar otros filtros para ver en cual de ellas se subclasifica. Se trata de una estructura en forma de árbol, los nodos hojas del árbol son alguna de las *qdiscs* sencillas de las vistas anteriormente.

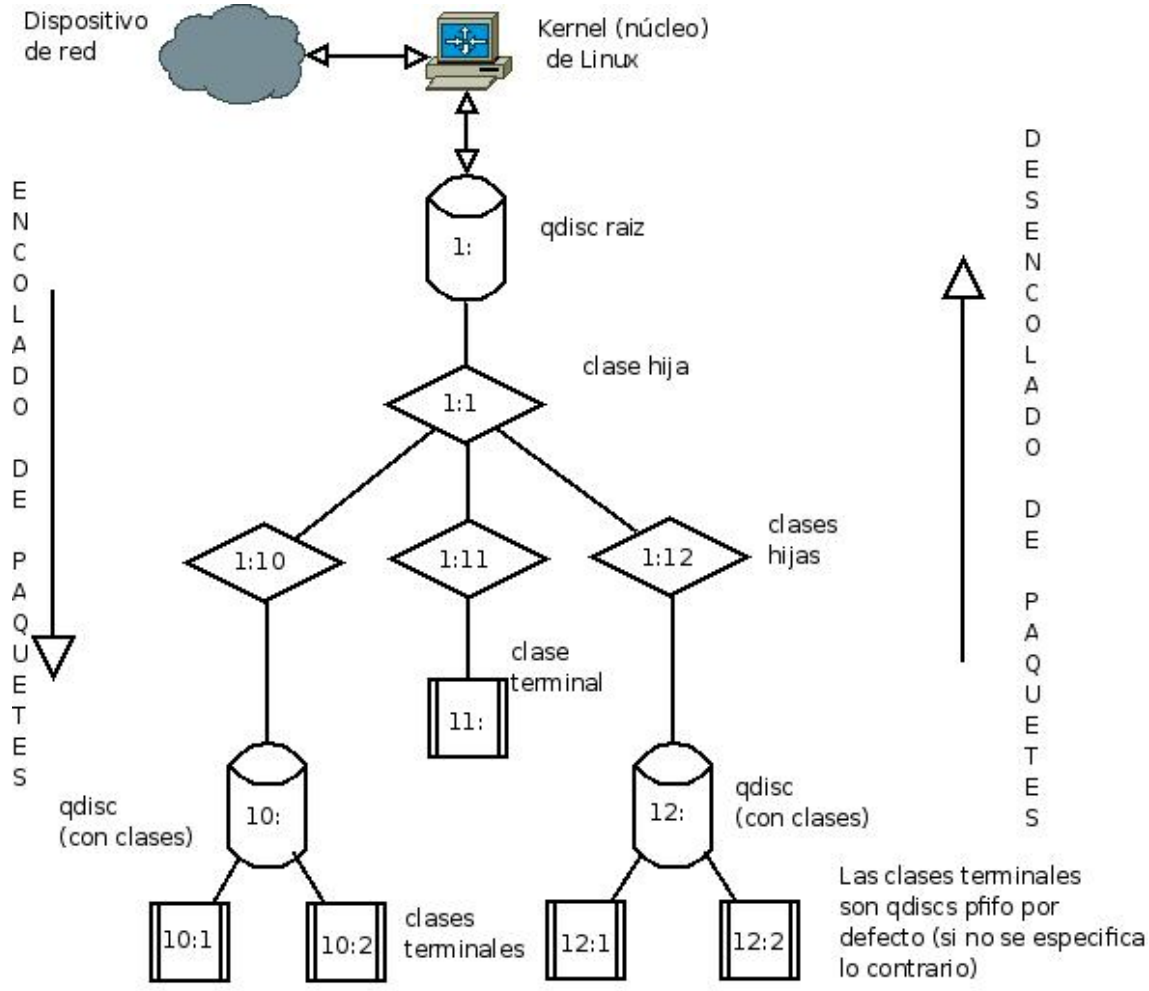
La mayoría de las qdisc con clases no se limitan a contener otras *sub-qdiscs* dentro de sus clases, si no que también aplican algún algoritmo de moldeado de tráfico, es decir, también realizan *«shaping»*. Esto es útil tanto para reordenar paquetes (con SFQ, por ejemplo) como para controlar tasas (con TBF por ejemplo).

3.10.1.1.1.5.1. Funcionamiento

Cada interfaz (*dispositivo de red*) tiene una «*qdisc raíz*» de salida. Por defecto, es la disciplina de colas pfifo sin clases que mencionamos anteriormente. A cada qdisc y clase se le asigna un controlador (*handle*), que puede usar en posteriores sentencias de configuración para referirse a la qdisc. Aparte de la qdisc de salida, la interfaz también puede tener una de entrada, que dicta las normas sobre el tráfico que entra. Los controladores de estas qdisc consisten en dos partes, un número mayor y un número menor:

<mayor>:<menor>

Es costumbre darle a la qdisc de raíz el nombre «1:», que es lo mismo que «1:0». El número menor de una qdisc siempre es 0. Las clases de una qdisc deben tener el mismo número mayor que sus padres. Este número mayor tiene que ser único dentro de una configuración de salida o entrada. El número menor debe ser único dentro de una qdisc y sus clases.



Ejemplo de una cola de encolado (qdisc) con clases

Pero no deje que este árbol le engañe, No debe imaginarse que el núcleo está en la cima del árbol y la red abajo, ya que no es el caso. Los paquetes se encolan y desencolan en la qdisc raíz, que es la única cosa con la que "*habla*" el núcleo.

Supongamos que llega un paquete y se clasifica en una cadena como ésta:
1: ⇨1:1 ⇨ 12: ⇨ 12:2
Ahora el paquete reside en una cola de una qdisc asociada a la clase **12:2**. En este ejemplo, se asocia un filtro a cada *«nodo»* del árbol, y en base a esos filtros se escoge por que rama del árbol descenderá un determinado paquete.
Sin embargo, también es posible esta situación:
1: ⇨ 12:2
En este caso, un filtro asociado a la raíz decidió enviar el paquete directamente a **12:2**.

¿Cómo se desencolan los paquetes para enviarlos al hardware?

Cuando el núcleo decide que necesita extraer paquetes para enviarlos a la interfaz (dispositivo de red), la **qdisc 1: raíz** recibe una petición de desencolar, que se pasa a **1:1**, que a su vez la pasa a **10:**, **11:**, y **12:**, cada una de las cuales consulta a sus descendientes, e intenta hacer *dequeue()* sobre ellos. En este caso, el núcleo necesita recorrer todo el árbol, porque sólo **12:2** contiene un paquete.

Las clases anidadas **solamente** hablan con sus qdisc paternas, y nunca con el dispositivo de red. Sólo la qdisc raíz recibe peticiones de desencolado por parte del núcleo para enviar paquetes al dispositivo de red. La consecuencia de esto es que las clases nunca desencolan más rápido de lo que sus padres permiten. Y esto es exactamente lo que queremos: de esta manera, podemos tener SFQ como clase interna (nodo hoja), que no hace ajustes, sólo reordena, y tenemos una qdisc padre, que es la que hace los ajustes.

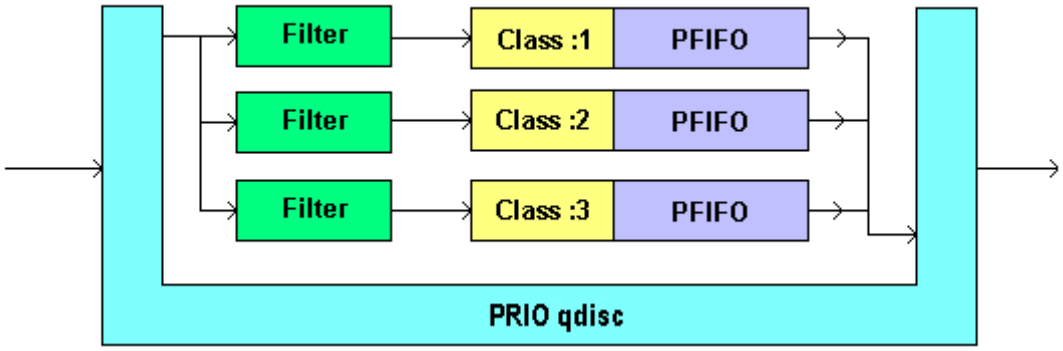
3.10.1.1.1.6. PRIO

La qdisc con clases PRIO en realidad no hace ajustes, sino que sólo subdivide el tráfico basándose en cómo haya configurado los filtros. Puede considerar la qdisc PRIO como una cola del tipo pfifo con clases, en la que cada banda es una clase separada, en lugar de una simple FIFO.

Cuando se encola un paquete a la qdisc PRIO, se escoge una clase basándose en las órdenes de filtrado que hayan definido. Por defecto, se crean tres clases. Estas subclases contienen una qdisc terminal que son puras FIFO sin estructura interna, pero puede sustituirlas por cualquier qdisc que haya disponible.

Siempre que se necesite desencolar un paquete, se intenta primero con la clase :1. Las clases más altas sólo se usan si no se ha conseguido el paquete en las clases más bajas.

Esta qdisc es muy útil en caso de que quiera dar prioridad a cierto tráfico Hablando formalmente, la qdisc PRIO es un reorganizador conservativo.



Cada vez que un paquete necesita ser desencolado se intenta primero en la clase **1:** Cuando se ha vaciado la clase **1:** se intenta con la clase **2:** y finalmente con la clase **3:**. Es evidente que siempre que halla paquetes en la clase **1:** estos serán desencolados antes que los otros y solo cuando se halla vaciado la clase **1:** se desencolaran las otras clases. Esto es útil para priorizar cierto tipo de tráfico como la VozIP. Pero a la vez puede suponer un problema si no se definen correctamente los filtros y la clase **1:** se llena mas rápido de lo que se vacía ya que en ese caso nunca se enviaría a la red el trafico de las clases **2:** y **3:**

3.10.1.1.1.7. CBQ (Class Based Queueing)

CBQ es la qdisc más compleja disponible, la más publicitada, la menos comprendida, y probablemente la más difícil de configurar correctamente. Esto no se debe a que los autores sean malvados o incompetentes, ni mucho menos, sino sólo que el algoritmo CBQ no es tan preciso y realmente no se ajusta del todo a la manera de trabajar de Linux.
Además de tener clases CBQ también es moldeadora del tráfico (shaper) y es en este aspecto es en el que no funciona del todo bien. Debería funcionar de esta manera:

- Si intenta ajustar a 1mbit/s una conexión de 10mbit/s, el enlace debería estar ocioso el 90% del tiempo. Si no lo está, necesitamos acelerar de manera que realmente ESTE ocioso el 90% del tiempo.

Esto es bastante difícil de medir, de manera que en su lugar CBQ deriva el tiempo ocioso del número de microsegundos que se tarda entre cada petición de más datos por parte de la capa de hardware. Combinados, se pueden usar para aproximar cómo de lleno o vacío está el enlace.

Esto es bastante tortuoso y no siempre lleva a resultados adecuados. Por ejemplo, ¿qué pasaría si la verdadera velocidad del enlace no es capaz de transmitir realmente todos los 100mbit/s de datos, quizá debido a un driver mal implementado? Una tarjeta de red PCMCIA tampoco alcanzará nunca los 100mbit/s debido a la manera en que está diseñado el bus (de nuevo, ¿cómo calculamos el tiempo?)

Se vuelve aún peor si consideramos dispositivos de red no del todo reales, como PPP sobre Ethernet (PPPoE) o PPTP sobre TCP/IP (una VPN por ejemplo). El ancho de banda efectivo en ese caso probablemente se determina por la eficiencia de los canales al espacio de usuario (que es enorme).

La gente que ha hecho mediciones ha descubierto que CBQ no es siempre muy preciso, y a veces se pierde del todo[73]. Sin embargo, en muchas circunstancias funciona bien. Con la documentación que proporcionamos aquí, debería ser capaz de configurarlo para que funcione bien en la mayoría de los casos.

[73] http://linuxreviews.org/man/tc-cbq-details/

CBQ trabaja asegurándose de que el enlace está ocioso sólo lo necesario para reducir el ancho de banda real hasta la tasa configurada. Para hacerlo, calcula el tiempo que debería pasar entre los paquetes medios. Mientras opera, se mide el tiempo ocioso efectivo usando una media de movimiento por exponencial proporcional (EWMA - exponential weighted moving average), que considera los paquetes recientes exponencialmente más importantes que los pasados. La media de carga de UNIX (loadaverage) se calcula de la misma manera.

El tiempo ocioso calculado se resta al medido mediante EWMA, y el número resultante se llama *«avgidle»*. Un enlace cargado perfectamente tiene un *avgidle* de cero: los paquetes llegan exactamente una vez cada intervalo calculado.

Un enlace sobrecargado tiene un *avgidle* negativo y si se vuelve muy negativo, CBQ lo cierra durante un rato y entonces se produce un *«sobrelímite»*.

Por el contrario, un enlace ocioso puede amasar un *avgidle* enorme, lo que permitiría anchos de banda infinitos tras unas horas de silencio. Para evitarlo, *avgidle* se trunca en *maxidle.*

Si hay un sobrelímite, en teoría, la CBQ debería acelerarse a sí misma durante exactamente el tiempo que se ha calculado que pasa entre paquetes, entonces pasa un paquete, y se acelera de nuevo.

CQB está diseñado para reducir el flujo a la velocidad requerida mediante la introducción de periodos inactivos para las colas de tráfico y el enlace. Calcula el tiempo medio que debería darse entre la transmisión de paquetes, para obtener la velocidad deseada. Luego utiliza una promedio exponencial móvil para calcular el tiempo inactivo actual: paquetes nuevos se consideran exponencialmente más importantes que los anteriores, de forma que el tráfico reciente tiene más influencia que el antiguo, pero al tiempo refleja la evolución del tráfico hasta ese instante. El valor calculado permite conocer si hay una infrautilización del enlace, que indica un exceso de ancho de banda; o una sobreutilización, que indicaría que es necesario reducir la velocidad forzando tiempos de inactividad para la clase de tráfico.

Aparte del ajuste, usando las aproximaciones de tiempo ocioso ya mencionadas, CBQ también actúa igual que la cola PRIO en el sentido de que sus clases pueden tener diferentes prioridades y que los números pequeños de prioridad se examinan antes que los grandes.

Como explica Floyd[74], a la hora de dar servicio a las clases, CBQ utiliza una implementación de weighted round-robin (WRR) con déficit, en el selector/planificador. Cada vez que la capa de hardware pide un paquete para enviarlo a la red, se inicia un proceso de round robin por pesos («WRR»), comenzando por las clases de prioridad con menor número. Estas se agrupan y se les pregunta si tienen datos disponibles. En tal caso, se devuelven. Después de que se haya permitido desencolar una serie de bytes a una clase, se prueba con la siguiente clase de esa misma prioridad.

Los siguientes parámetros controlan el proceso WRR:

- **allot**

Cuando se le pide a la CBQ externa un paquete para enviar por la interfaz, buscará por turnos en todas sus qdisc internas (en las clases), en el orden del parámetro de «prioridad». Cada vez que le toca el turno a una clase, sólo puede enviar una cantidad limitada de datos. «Allot» es la unidad básica de esta cantidad

- **prio**

La CBQ también puede actuar como un dispositivo PRIO. Primero se prueba con las clases internas de mayor prioridad y mientras tengan tráfico, no se mira en las otras clases.

- **weight**

Weight ayuda en el proceso de Weighted Round Robin. Cada clase tiene una oportunidad por turnos para enviar. Si tiene una clase con un ancho de banda significativamente mejor que las otras, tiene sentido permitirle enviar más datos en su ronda que a las otras. Una CBQ suma todos los pesos bajo una clase, y los normaliza, de manera que puede usar números arbitrarios: sólo son importantes las equivalencias. La gente viene usando «tasa/10» como regla general y parece que funciona bien. El peso renormalizado se multiplica por el parámetro «allot» para determinar cuántos datos se envían en una ronda.

Podría pensarse que CBQ es una solución ideal para el problema considerado. Sin embargo existen varios problemas derivados de la complejidad del algoritmo. La gran cantidad de variables que entran en juego hacen difícil configurar el reparto de ancho de banda (como muestran

[74] http://www.icir.org/floyd/cbq.html

las configuraciones de ejemplo mostradas en la página de Lartc [75]). [LARTC]. Establecer el tiempo inactivo adecuado para el enlace no es sencillo; y los complejos cálculos no se ajustan o no están optimizados para las necesidades y características de los ordenadores personales.

La qdisc CBQ tiene muchos mas parámetros que no entraremos a describir debido a su complejidad, su uso esta desaconsejado debido a la complejidad del algoritmo, la enorme cantidad de parámetros que poca gente comprende del todo bien y a los problemas anteriormente mencionados.

3.10.1.1.1.8. HTB (Hierarchical Token Bucket)

Martin Devera se dio cuenta de que CBQ es complejo y que no parece óptimo en varias situaciones típicas. Su enfoque Jerárquico se ajusta bien a configuraciones donde se tiene una cantidad fija de ancho de banda a dividir para diferentes propósitos, dándole a cada propósito un ancho de banda garantizado, con la posibilidad de especificar cuánto ancho se puede tomar prestado.

HTB funciona igual que CBQ, pero no recurre a cálculos de tiempo ocioso para los ajustes. En su lugar, es un Token Bucket Filter con clases (de ahí el nombre). Sólo tiene unos pocos parámetros que están bien documentados en su página web[76].

HTB es mucho más sencillo de configurar que CBQ y escala bien a configuraciones más complejas. Con CBQ ya es complejo incluso en casos simples.

Como el nombre implica, HTB consiste en una colección de token bucket filters (TBF), podríamos definirlo como un TBF con clases.

Funcionalmente HTB persigue el mismo objetivo que CBQ pero lo hace de una forma más sencilla y eficaz. HTB nace como un reemplazo, mas efectivo e intuitivo de la qdisc CBQ. Tanto CBQ como HTB permiten controlar el ancho de banda de un determinado dispositivo de red. Ambos permiten simular dentro de un enlace físico varios *"subenlaces"* mas lentos (las clases) y enviar diferentes tipos de tráfico por cada *"subenlace"* de

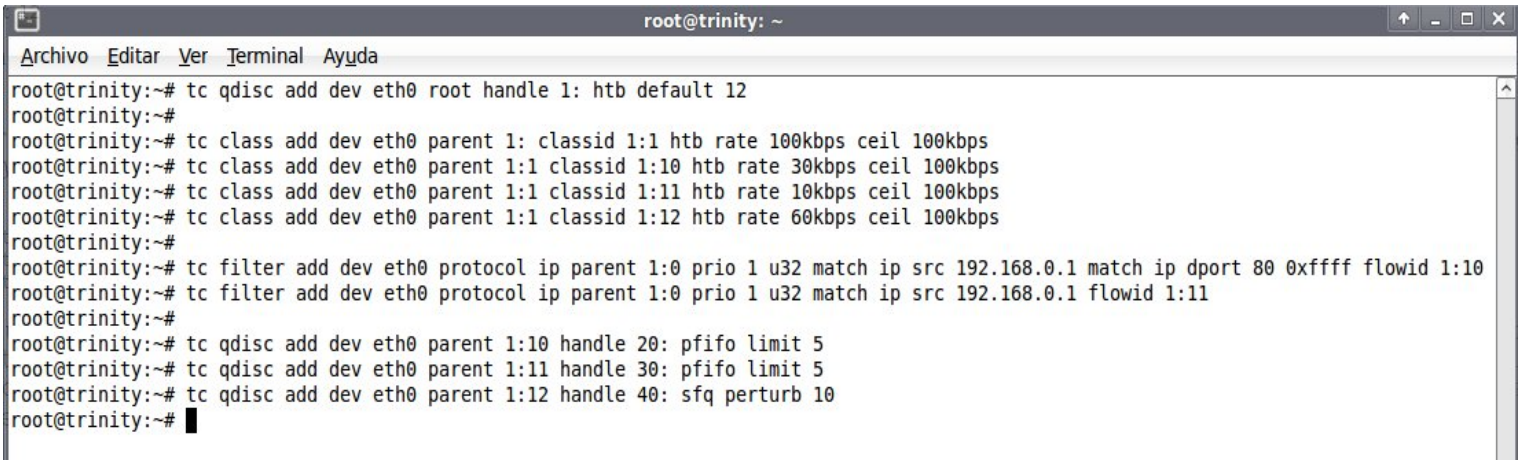
[75] http://lartc.org/
[76] http://luxik.cdi.cz/~devik/qos/htb/

forma que se adjudique mas ancho de banda a un determinado tipo de tráfico que a otro.

HTB asegura que la cantidad de ancho de banda disponible para cada clase es siempre el mínimo entre lo que esta solicita y lo que se le ha asignado previamente. Cuando una clase solicita menos ancho de banda de lo que tiene asignado el ancho de banda que sobra es distribuido entre las otras clases si estas lo están solicitando.

En la literatura[77] se le denomina a esto "tomar prestado" el ancho de banda que sobra, sin embargo este término no es del todo correcto ya que no existe la necesidad de *"devolver el prestamo"*.

Veamos como se configura una qdisc con HTB:



La primera línea crea la qdisc HTB **1:** y la adjunta al dispositivo de red *eth0*. La segunda línea crea una clase **1:1** con una velocidad de 100kbps. Las líneas 3,4,5 crean tres subclases dentro de la clase **1:1**. Esto lo hacemos así porque la clase principal no puede *"tomar prestado"* de otras clases principales. Una clase en una qdisc HTB permite a sus hijos *"tomar prestado"* de los otros, pero la clase principal no puede tomar prestado de otra clase principal ya que su padre es la propia qdisc. De esta forma configuramos tres subclases a 30kbps, 10kbps y 60kbps dentro de una clase principal a 100kbps. Las tres subclases pueden *"tomar prestado"* ancho de banda unas de otras cuando a una le sobre y las demás lo necesiten.

Las siguientes líneas definen los filtros que harán que el tráfico se clasifique en una clase u en otra. La primera línea de filtrado define que todos los paquetes que provengan desde 192.168.0.1 y se dirijan al puerto 80 (http) se clasificaran en la clase **1:10** (la de 30kbps). La siguiente línea

[77] http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm

define que el resto del trafico desde 192.168.0.1 se clasificara en la qdisc **1:11** (la de 10kbps). El tráfico que no es clasificado por ninguno de los filtros va a la clase por defecto **1:12** que se definió en la primera línea al crear la clase HTB.

Por defecto se asigna la qdisc pfifo a las clases terminales, las ultimas tres líneas cambian esa asignación por defecto asignando una qdisc pfifo con limite de 5 paquetes a **1:10** y **1:11** y asignando una qdisc sfq a **1:12**

El esquema quedaría de esta forma:

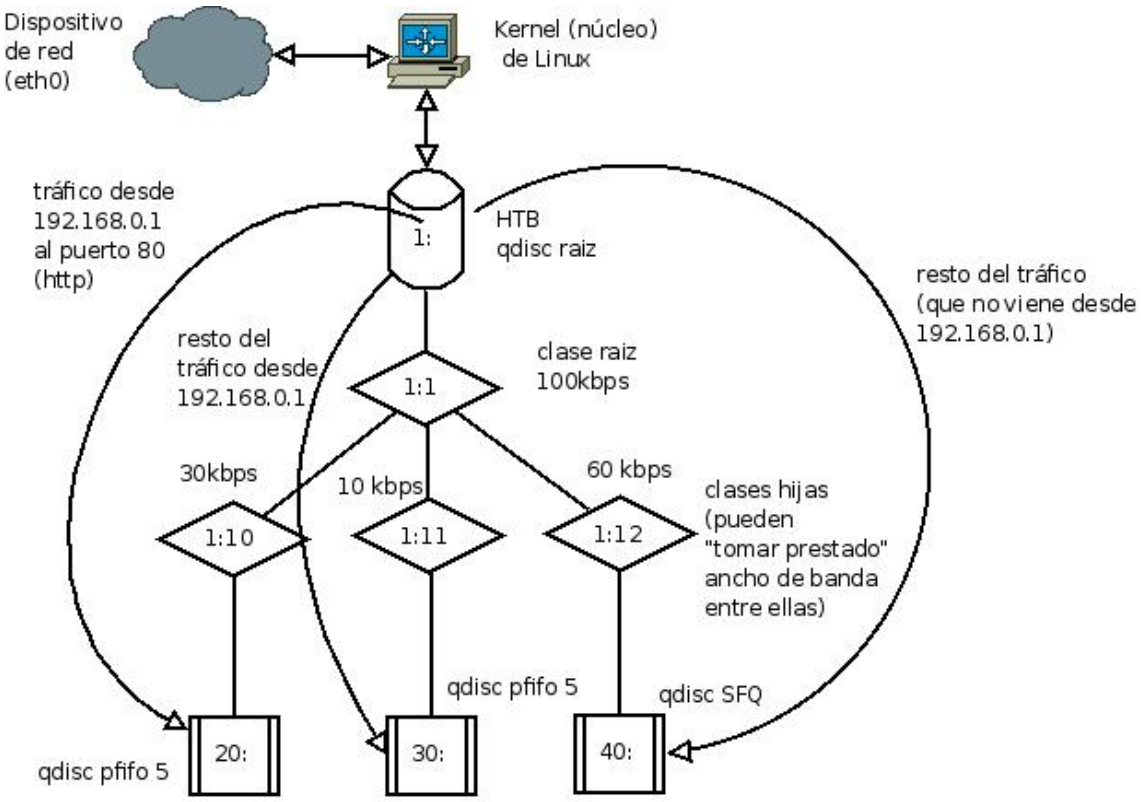


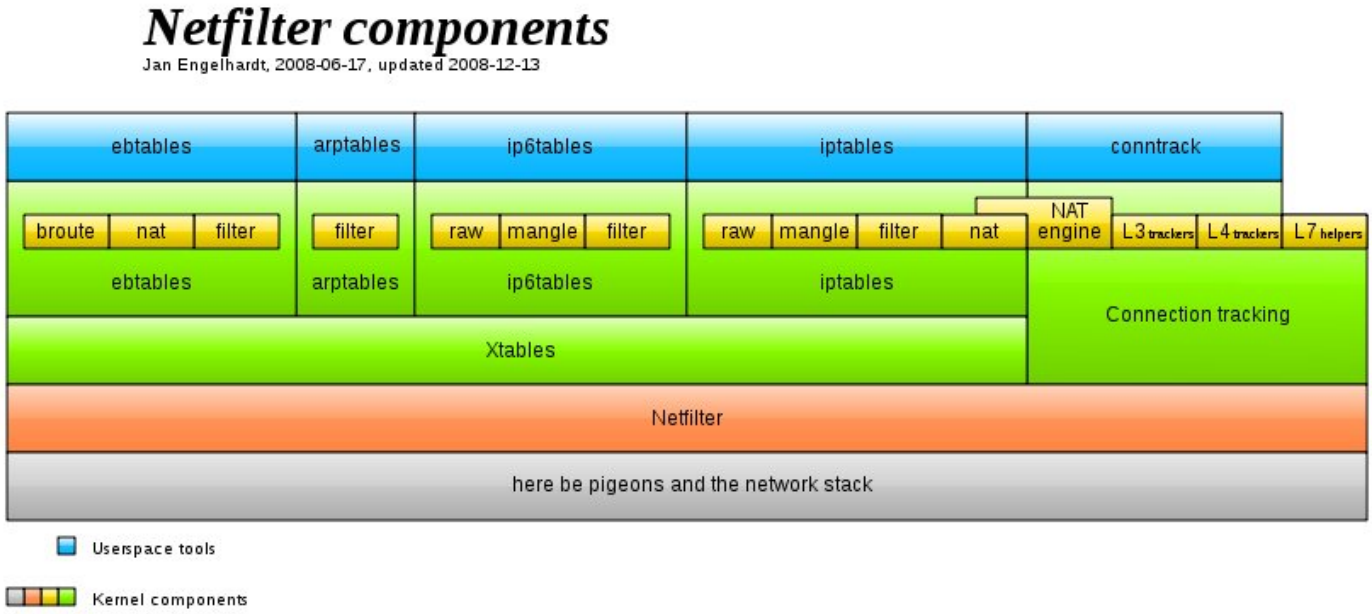
Gráfico de la configuración de ejemplo creada para ilustrar HTB

Este esquema define que el máximo ancho de banda que enviaremos por la interfaz eth0 es de 100kbps, de los cuales 30kbps estan reservados para el tráfico http de 192.168.0.1 y 10kbps reservados para el resto del tráfico desde esta IP. El resto del tráfico (que no proviene desde 192.168.0.1) tiene 60kbps reservados. Si alguna de las clases no esta usando todo el ancho de banda que tiene reservado entonces sus clases hermanas pueden "*tomar prestado*" el exceso pero el conjunto global nunca superará los 100kbps.

También conviene resaltar que la cantidad que pueden tomar prestada es proporcional a la proporción que tienen reservada. Esto es, si la clase **1:12** no esta usando nada de los 60kbps que tiene reservados y las otras clases están pidiendo más ancho de banda se asignará en la proporción 3 a 1 ya que la clase **1:10** tiene el triple de ancho de banda reservado que la clase **1:11**. Es decir, en este caso la clase **1:10** conseguirá 45kbps extra (el 75%) y la **1:11** los 15kbps restante (25%). De esta forma está asegurado el reparto equitativo del ancho de banda.

3.10.1.2. Netfilter.

Netfilter es un framework disponible en el kernel de Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento.



El componente más popular construido sobre Netfilter es iptables, una herramientas de cortafuegos que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT), mantener registros de log y muchas otras cosas que exceden las capacidades de los sistemas de más alto rendimiento. Realmente se pueden hacer cosas realmente espectaculares con iptables si se tiene la paciencia necesaria para aprender a manejar todas sus posibilidades. 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.

Otro componente muy interesante es ebtables que permite filtrar y manejar el tráfico directamente en la capa 2 del modelo TCP/IP, lo cual es muy útil cuando nuestro sistema esta haciendo de *"bridge"* o *"switch"* de la red.

Todos estos componentes están llamados a desaparecer en el futuro próximo en favor de Xtables que los reemplazará e integrará en una sola herramienta de diseño modular que permitirá añadir nuevas características y reglas de filtrado simplemente cargando los módulos sin necesidad de recompilar el código como ocurre ahora mismo. El soporte para Xtables[78] se incorpora al núcleo de Linux a partir de las versiones 2.6.30 y superiores.

3.10.1.3. Iptables

Iptables es un conjunto de herramientas que le permiten al usuario enviar mensajes al kernel de Linux para controlar la pila TCP/IP. El kernel tiene todo el manejo de paquetes TCP/IP metido dentro de él, no es algo aparte como en otros sistemas operativos.

Los paquetes de red tienen muchas características, algunas pueden ser los valores que tienen en sus encabezados (a donde se dirigen, de donde vienen, números de puertos, etc., etc.), otra puede ser el contenido de dicho paquete (la parte de datos), y existen otras características que no tienen que ver con un paquete en particular sino con una sumatoria de ellos. La idea es lograr identificar un paquete y en base a ello hacer una u otra cosa con el.

[78] http://freshmeat.net/projects/xtables-addons/

Por lo tanto, voy a definir a iptables como *"un conjunto de comandos que permiten decirle al kernel qué hacer con ciertos paquetes que cumplan con ciertas características"*.

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

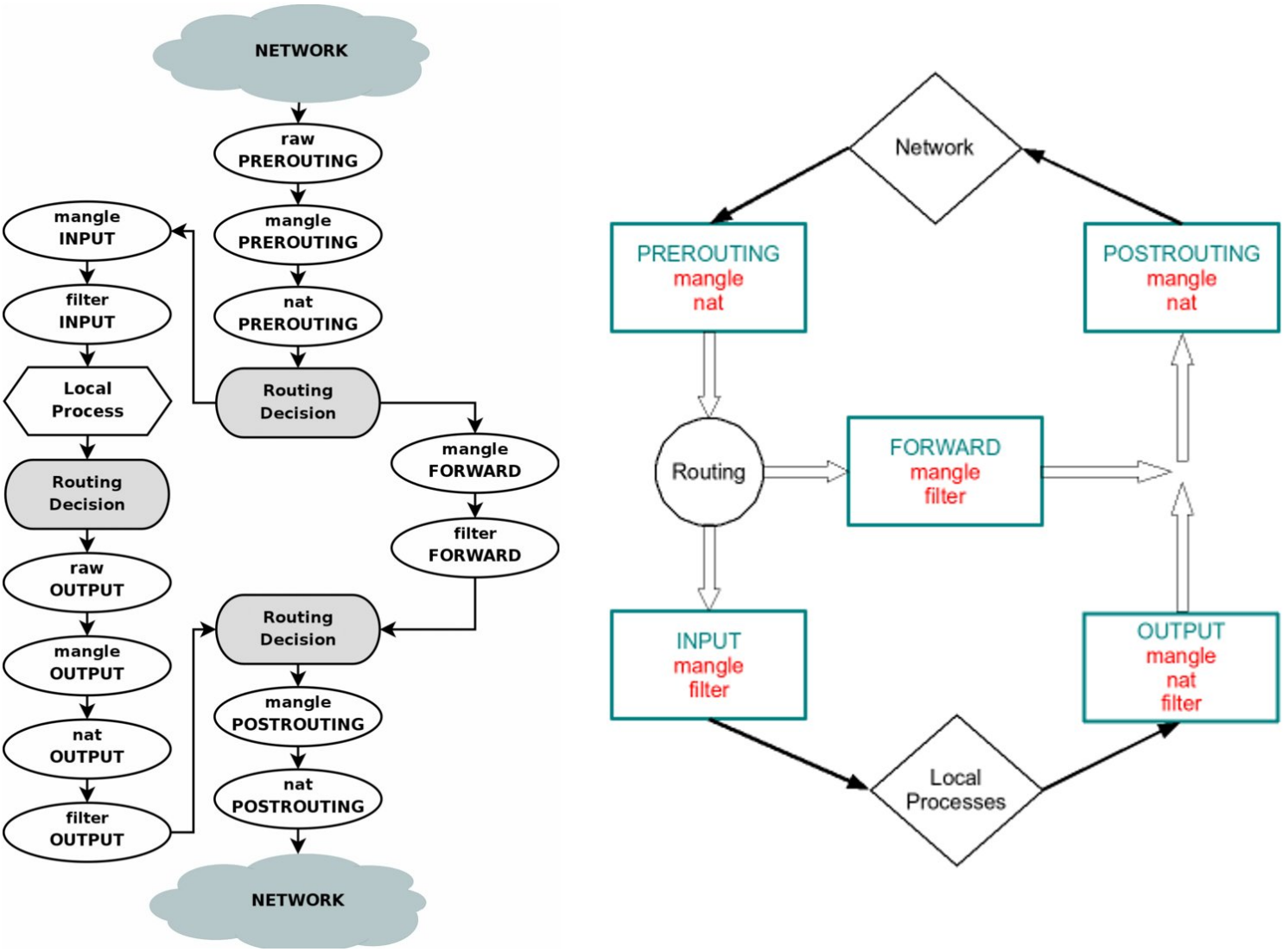
Reglas *(Rules)* $\in$ **Cadenas** *(Chains)* $\in$ **Tablas** *(Tables)*

Cada regla especifica qué paquetes la cumplen (*match*) y un destino que indica qué hacer con el paquete si éste cumple la regla. Cada paquete de red que llega a una computadora o que se envía desde una computadora recorre por lo menos una cadena y cada regla de esa cadena se comprueba con el paquete. Si un paquete verifica determinada regla, se detiene la comprobación de las sucesivas reglas ya que esa regla determinará que hacer con el paquete. Si tras comprobar todas las reglas el paquete no ha podido ser verificado por ninguna de ellas entonces la *política* por defecto de la cadena dicta qué hacer con el paquete.

Estas son las tablas definidas en iptables y las cadenas que contiene cada una de ellas. El usuario puede crear nuevas cadenas si lo desea, cada una de estas cadenas tiene definida una política por defecto que indica que hacer con el paquete si este no verifica ninguna regla de la cadena.

- FILTER
  - INPUT
  - FORWARD
  - OUTPUT
- NAT
  - PREROUTING
  - POSTROUTING
  - OUTPUT
- MANGLE
  - PREROUTING
  - INPUT
  - FORWARD
  - OUTPUT
  - POSTROUTING
- RAW
  - PREROUTING
  - OUTPUT

Las acciones que se pueden tomar sobre un paquete son diversas: aceptarlo, rechazarlo, modificarlo, redirigirlo…



La anterior figura ilustra el flujo de los paquetes de datos por las diferentes tablas y cadenas de Iptables

Veremos ahora el flujo detallado de los paquetes según las tres posibilidades que existen (enviar un paquete, recibirlo y enrutarlo):

- Recibir un paquete
  1 MANGLE [PREROUTING]
    - Esta cadena se utiliza generalmente para cambiar datos de los paquetes recibidos antes de que lleguen al sistema, por ejemplo cambiar el valor TOS

2 NAT [PREROUTING]
   - Aquí suele hacerse DNAT

3 MANGLE [INPUT]
   - Esta cadena se utiliza para modificar los paquetes después que han sido ruteados, pero antes de que se lo envíe al proceso de la máquina

4 FILTER [INPUT]
   - Aquí es donde se filtran todos los paquetes que tienen como destino final nuestra máquina.

- Enviar un paquete

1 MANGLE [OUTPUT]
   - Aquí se pueden modificar los paquetes que salen de nuestra máquina y modificar valores de sus cabeceras antes de que se tome la decisión de ruteo.

2 NAT [OUTPUT]
   - En esta cadena se pueden hacer traducciones de direcciones de red.

3 FILTER [OUTPUT]
   - Aquí se filtran todos los paquetes que salen de nuestra máquina

4 MANGLE [POSTROUTING]
   - Esta cadena se utiliza cuando queremos cambiar un paquete antes de que salga del equipo, y después que se haya tomado la decisión de ruteo.

5 NAT [POSTROUTING]
   - Esta cadena se utiliza para cambiar las direcciones origen de los paquetes (SNAT)

- Enrutar un paquete

1 MANGLE [PREROUTING]
   - Esta cadena se utiliza, generalmente, para cambiar los paquetes, por ejemplo cambiar el TOS

2 NAT [PREROUTING]
   - Esta cadena se utiliza más que nada para hacer DNAT

3 MANGLE [FORWARD]
   - Aquí es donde se filtran todos los paquetes que atraviesan nuestra máquina y que son enrutados hacia otras máquinas. Los paquetes que vienen o van desde Internet con destino algún equipo de la LAN pasan por esta cadena.

4 MANGLE [POSTROUTING]
   - Esta cadena se utiliza cuando nosotros queremos cambiar un paquete antes de que salga del equipo, y después que se haya tomado la decisión de ruteo.

5 NAT [POSTROUTING]

- Esta cadena se utiliza para cambiar las direcciones origen de los paquetes (SNAT)

La tabla RAW se utiliza principalmente para configurar excepciones del registro de conexiones TCP activas, permite modificar los paquetes a la entrada y la salida de la interfaz antes de que estos sean procesados por la pila TCP/IP.

La verdadera potencia de Iptables radica en su diseño modular, que permite añadir nuevos módulos para identificar los paquetes. Por ejemplo existe un modulo capaz de clasificar los paquetes por el origen geográfico del mismo, es decir, es capaz de identificar el país de origen del paquete basándose en una base de datos que relaciona rangos IP con países[79].

3.10.1.3.1. ipp2p

ipp2p es un parche[80] que se aplica a Iptables y que añade la funcionalidad de poder identificar el tráfico P2P. Para este propósito, ipp2p examina el tráfico utilizando patrones pertenecientes a múltiples protocolos P2P. Una vez que el tráfico se ha identificado como P2P se puede tratar de diversas formas: bloqueándolo, o aumentando o reduciendo su prioridad.

El uso principal de ipp2p suele ser el de mejorar el rendimiento global de la red limitando el tráfico P2P ya que este tipo de tráfico tiende a ser muy intensivo y colapsa en ocasiones las redes.

3.10.1.3.2. layer7

Layer7[81] es otro parche que se aplica a iptables, su objetivo es más amplio que el de ipp2p, ya que se puede utilizar para clasificar todo tipo de protocolos de red de la capa 7 (de aplicación) como podrían ser por ejemplo http, ftp, bittorrent, emule, sip…

[79] http://people.netfilter.org/~peejix/geoip/howto/geoip-HOWTO.html
[80] Un parche (match) es un pequeño pedazo de código software diseñado para arreglar problemas, mejorar el rendimiento o añadir características a otro software.
[81] http://l7-filter.sourceforge.net

El funcionamiento de layer7 se basa en el uso de patrones para identificar las conexiones en la capa de enlace y determinar que protocolo están usando. Por supuesto, no tendría mucho sentido examinar cada uno de los paquetes de datos que pasan a través del sistema ya que eso consumiría una gran cantidad de ciclos de reloj de la CPU. En lugar de eso layer7 examina solo los primeros paquetes de cada conexión buscando mensajes del tipo *"220 ftp server ready"* o *"HTTP/1.1 200"* que le permitan identificar los protocolos. Gracias a que solo unos pocos paquetes son examinados por cada conexión, layer7 es razonablemente eficiente y consigue clasificar la mayoría de los protocolos con una tasa de acierto elevada.

El uso de layer7 es muy interesante para dar prioridad a diferentes protocolos como http o SIP (VozIP) o restar prioridad a otros como los protocolos P2P

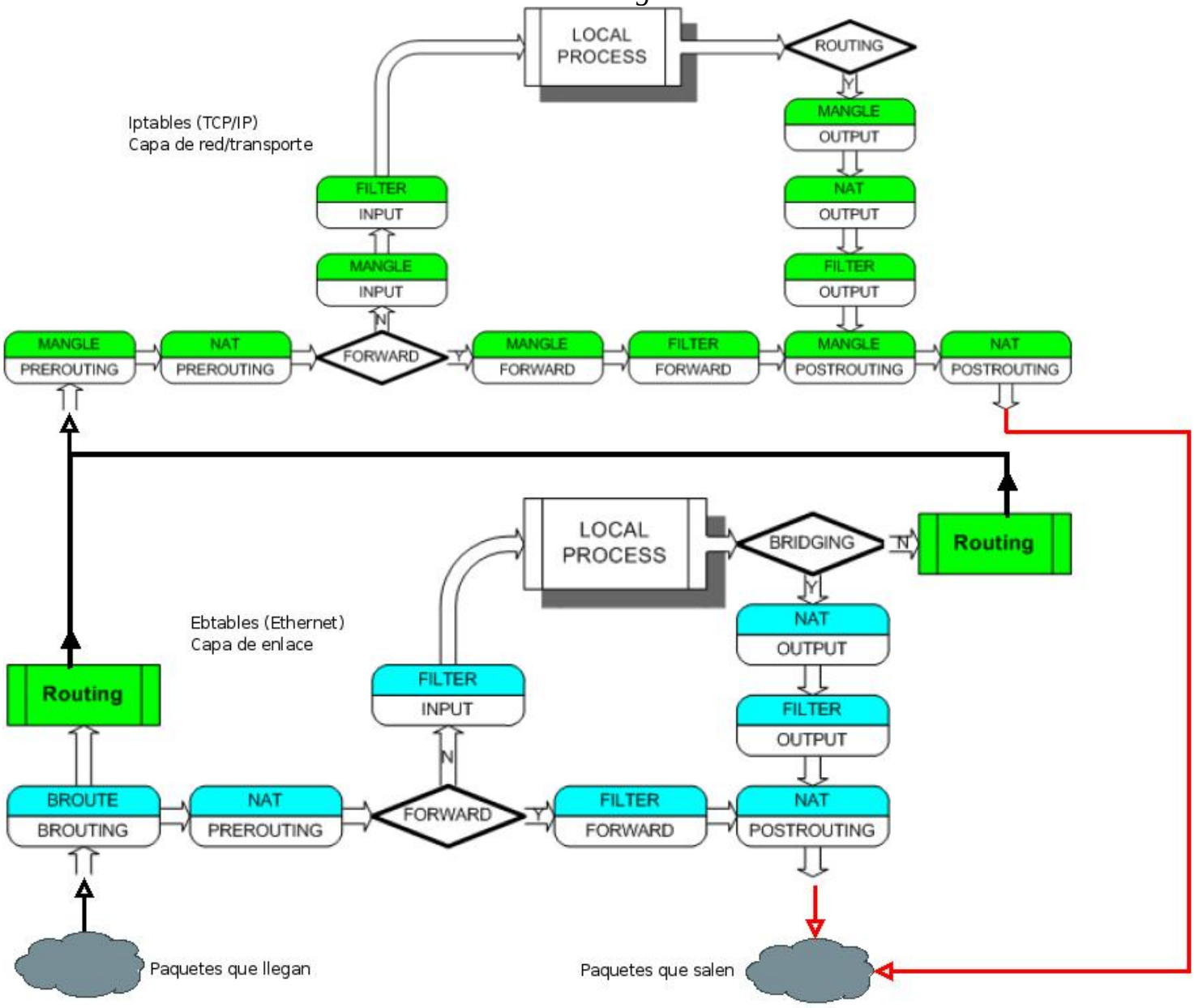
3.10.1.4. Ebtables

Ebtables es muy similar a Iptables, se diferencia de este en que trabaja sobre el protocolo Ethernet, es decir, permite filtrar y modificar los paquetes de datos en la capa 2 (de enlace) del modelo TCP/IP. Con ebtables se pueden filtrar los paquetes en la capa de enlace y alterar sus direcciones MAC.

Su funcionamiento es muy similar a Iptables, las ordenes de filtrado se agrupan de la misma forma (reglas, cadenas y tablas). En ebtables tenemos las siguientes tablas:

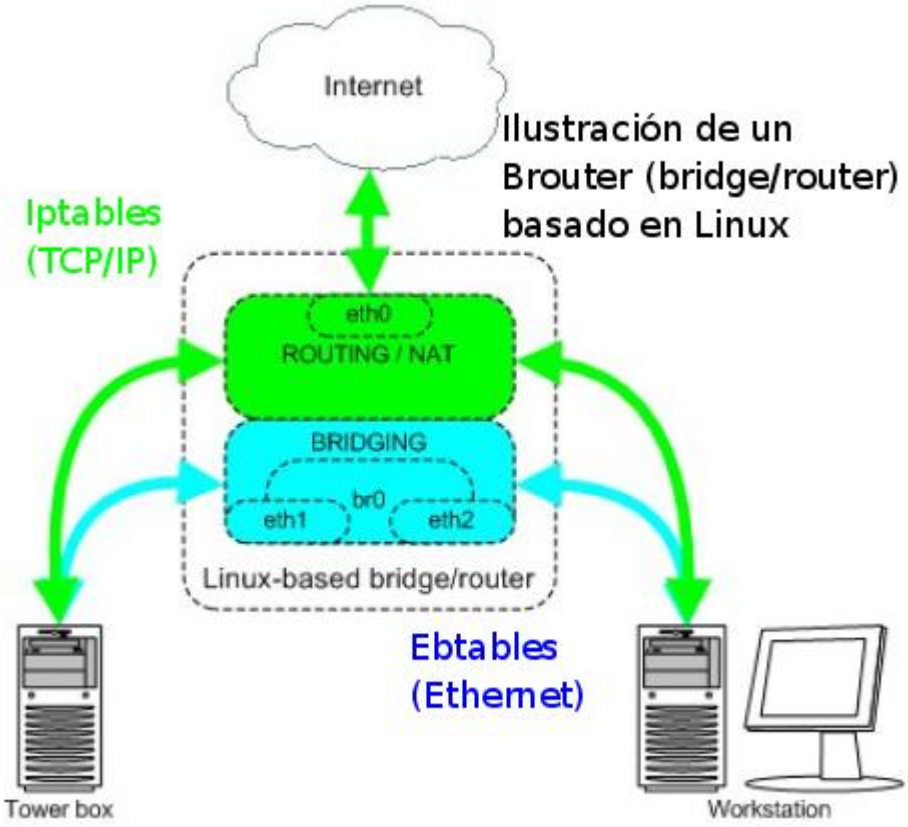
- FILTER: Contiene tres cadenas
  - INPUT: Para paquetes destinados a la maquina
  - FORWARD: Para paquetes que son redirigidos a través del *bridge*
  - OUTPUT: Para paquetes enviados desde la maquina
- NAT: Se utiliza para manipular las direcciones MAC
  - PREROUTING: Para alterar los paquetes tan pronto llegan.
  - OUTPUT: Para alterar paquetes generados en la propia maquina
  - POSTROUTING: Para alterar paquetes que van a ser enviados.

- BROUTE: Se utiliza para hacer un brouter[82]
  - BROUTING: Permite definir que paquetes pasaran a la pila TCP/IP para que se tome la decisión de ruteo adecuada y cuales son directamente "*bridgeados*" a la red.



Flujo de paquetes a través de Ebtables e Iptables

[82] Un brouter (bridge/router) es un dispositivo que trabaja como router para los protocolos encaminables (TCP/IP) y como bridge para los que no lo son (Ethernet)



3.10.1.5. El dispositivo intermedio de encolado (IMQ)

Linux permite controlar el tránsito de datos tanto a la entrada de como a la salida, pero se hacen distinciones. Pues, efectivamente moldear o planificar el tráfico que ya ha llegado no es tan eficaz como hacerlo sobre el tráfico saliente, se tomó la decisión de diseño de no permitir usar disciplinas de encolado con clases sobre el tráfico entrante. Esto implica que no se puede aplicar HTB o CBQ directamente a los datos entrantes (ingress).

Efectivamente, una interfaz de red no tiene control sobre el tráfico que recibe del exterior, no depende de el, por lo que tratar de moldear o planificar el tráfico de entrada en un dispositivo significa descartarlo con la esperanza de que la ventana de congestión del protocolo TCP del transmisor se llene y este se vea forzado a transmitir más lento. Por hacer una analogía con el mundo real es como si intentas dejar de recibir menos cartas en tu buzón de correo tirando a la basura el exceso con la esperanza de que quien te escribe se de cuenta de que te está mandando demasiadas y reduzca su tasa de envío. Probablemente esto no funcione en el mundo real, pero en una red TCP/IP si funciona gracias a que el protocolo TCP esta diseñado para ello.

El dispositivo intermedio de encolado IMQ (*Intermediate queueing device*) no es una cola de encolado de paquetes pero su uso está muy unido a ellas.

Dentro de Linux, las qdisc se asocian a dispositivos de red y todo lo que se encola en el dispositivo se encola antes en la qdisc. Partiendo de este concepto, surgen dos limitaciones:

1. Sólo se pueden hacer ajustes de salida (existe una qdisc de entrada, pero sus posibilidades son muy limitadas comparadas a las qdisc con clases).

2. Una qdisc sólo puede ver el tráfico de una interfaz, de manera que no se pueden imponer limitaciones que engloben varias interfaces de red o a todo el sistema.

IMQ está aquí para resolver ambas limitaciones. En breve, se puede poner todo lo que se quiera en una qdisc. Su funcionamiento es el siguiente:

- Con Iptables se marcan los paquetes que se quieren "enviar" al dispositivo IMQ. Dicho dispositivo es una interfaz de red virtual que se utiliza para hacer ajustes del ancho de banda

- Dichos paquetes marcados de forma especial los interceptan las ranuras NF_IP_PRE_ROUTING y NF_IP_POST_ROUTING de netfilter y pasan por una qdisc asociada a un dispositivo imq virtual.

- Dentro de la interfaz virtual imq se realizan los ajustes de ancho de banda requeridos mediante el uso de qdiscs

- Los paquetes vuelven a su interfaz de red original después del ajuste de ancho de banda.

Esto permite que haga ajustes de entrada ya que puede marcar los paquetes que vienen de cualquier sitio o tratar las interfaces como clases para imponer límites globales. También se puede hacer muchas otras cosas como por ejemplo poner el tráfico http en una qdisc y poner las peticiones de conexiones nuevas (TCP SYN) en otra qdisc.

El dispositivo IMQ no forma parte del núcleo (kernel) de Linux oficial y es necesario aplicar un parche al kernel para darle soporte para IMQ. La razón de que no esté incluido en el núcleo oficial de Linux es que IMQ es que este basa su funcionamiento en la intercepción de los paquetes que entran o salen de la pila IP. Por lo cual constituye una solución teóricamente incorrecta, pero que funciona, hecho que confirma el propio equipo de Linux IMQ.

"IMQ es sólo una chapuza para resolver problemas existentes y es teóricamente incorrecto, sin embargo en la práctica funciona correctamente (...) Esta funcionalidad no debería ser implementada al nivel de la pila de red donde IMQ está ahora" [83]

3.11. Lenguajes de programación

En este apartado se va a proceder a justificar el uso de los diferentes lenguajes de programación que se han utilizado para implementar el sistema final.

3.11.1. PHP

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas Web dinámicas. Es usado principalmente en interpretación del lado del servidor (*server-side scripting*) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

PHP es un acrónimo recursivo que significa *PHP Hypertext Pre-processor*. Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal.

[83] http://www.linuximq.net/faq.html

En el siguiente proyecto se ha utilizado PHP para la programación de las interfaces Web, tanto para la página de administración como para la página de bienvenida al sistema. Su uso está justificado en parte a:

- Es un lenguaje multiplataforma y completamente orientado a la Web.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, en concreto es esencial la capacidad de conexión con la base de datos LDAP del sistema
- Biblioteca nativa de funciones sumamente amplia
- A pesar de no ser un lenguaje compilado se ejecuta a gran velocidad comparado con otros lenguajes interpretados.

3.11.2. Perl

Perl (*Practical Extraction and Report Languaje*) es un lenguaje de programación diseñado por Larry Wall en 1987. Perl es un lenguaje imperativo[84], con variables, expresiones, asignaciones, bloques de código delimitados por llaves, estructuras de control y subrutinas. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

En el presente proyecto se ha utilizado Perl para construir parte del núcleo del sistema, su elección se debe principalmente a la existencia de una gran cantidad de librerías de alto nivel[85] que facilitan enormente la interacción con el servidor RADIUS (para autenticar a los usuarios) y con el servidor de base de datos LDAP.

[84] En la programación imperativa se describe paso a paso un conjunto de instrucciones que deben ejecutarse para variar el estado del programa y hallar la solución, es decir, un algoritmo en el que se describen los pasos necesarios para solucionar el problema. Por contraposición en la programación declarativa las sentencias que se utilizan lo que hacen es describir el problema que se quiere solucionar, pero no las instrucciones necesarias para solucionarlo. Esto último se realizará mediante mecanismos internos de inferencia de información a partir de la descripción realizada.

[85] http://www.cpan.org/

3.11.3. Bash scripting

Bash es un shell de Unix (intérprete de órdenes de Unix) escrito para el proyecto GNU. Su nombre es un acrónimo de *bourne-again shell*, haciendo un juego de palabras (born-again significa renacimiento) sobre el Bourne shell (sh), que fue uno de los primeros intérpretes importantes de Unix.

Hacia 1978 el intérprete Bourne era el intérprete distribuido con el Unix Version 7. Stephen Bourne, por entonces investigador de los Laboratorios Bell, escribió el intérprete Bourne original. Brian Fox escribió el intérprete bash en 1987. En 1990, Chet Ramey se convirtió en su principal desarrollador. Bash es el intérprete predeterminado en la mayoría de sistemas GNU/Linux, además de MacOS X, y puede ejecutarse en la mayoría de los sistemas operativos tipo Unix.

Un script para bash es un archivo tipo texto, cuyas líneas tienen comandos que son ejecutados (interpretados) por bash. Para lograr que el intérprete de comandos intérprete las líneas el archivo ha de comenzar con la línea *#!/bin/bash*

En el proyecto actual se han implementado multitud de scripts para bash que realizan diversas funciones como la de conectar/desconectar a un determinado usuario del sistema.

La elección de scripts en bash para la realización de estas tareas viene justificada por la facilidad, sencillez y rapidez con la que se implementan las soluciones.

3.11.4. C

C es un lenguaje de programación creado en 1972 por Kenneth L. Thompson, Brian Kernighan y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL.

Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

C proporciona al desarrollador un núcleo del lenguaje simple, con una gran cantidad de funcionalidades añadidas importantes, como funciones matemáticas, de manejo de archivos y de interacción con el sistema operativo, proporcionadas por diferentes librerías.

En el presente proyecto se ha utilizado el lenguaje C, de forma marginal, para programar algunas utilidades donde los *scripts para bash* no ofrecían la funcionalidad requerida como la ejecución de servicios en segundo plano.

Capitulo 4. Metodología

En este capitulo realizaremos una descripción del proceso de desarrollo utilizado en este proyecto para luego analizar detenidamente el funcionamiento del mismo con el lenguaje de modelado UML.

4.1. Proceso de desarrollo unificado

El proyecto se inicia con una gran especificación de requisitos con el objetivo de que tanto el desarrollador como el empresario adquieran una idea global de la aplicación. Posteriormente se ha seguido un proceso de desarrollo incremental estableciéndose reuniones periódicas con la empresa, como mínimo al final y comienzo de cada fase, lo cual ha permitido asegurar la correcta evolución del proyecto.

Una vez identificados los requisitos de la aplicación se pasa a la segunda fase del desarrollo, la implementación de la aplicación mediante la metodología de Proceso Unificado de Desarrollo, la cual está guiada por un ciclo de vida iterativo e incremental. Ésta no es una metodología estrictamente dicha, sino que es un proceso marco donde se definen unas directrices. El Proceso Unificado de Desarrollo es flexible y extensible, lo cual permite que sea susceptible de adaptarse a diferentes tipos de proyectos.

Algunas de las características más importantes de esta metodología son:

- Está dirigido por casos de uso: En el Proceso Unificado de Desarrollo los casos de uso se utilizan para obtener los requisitos funcionales de la aplicación y para definir los contenidos de las iteraciones. La idea fundamental es que cada iteración tome un conjunto de casos de uso o escenarios y desarrolle todo el camino a través de las distintas disciplinas.

- Es iterativo e incremental: Este tipo de desarrollo se caracteriza porque se realiza en diversas fases, en las cuales se va añadiendo

- Está centrado en la arquitectura: El Proceso Unificado asume que no existe un modelo único que cubra todos los aspectos del sistema. Por este motivo existen múltiples modelos y vistas que definen la arquitectura del sistema

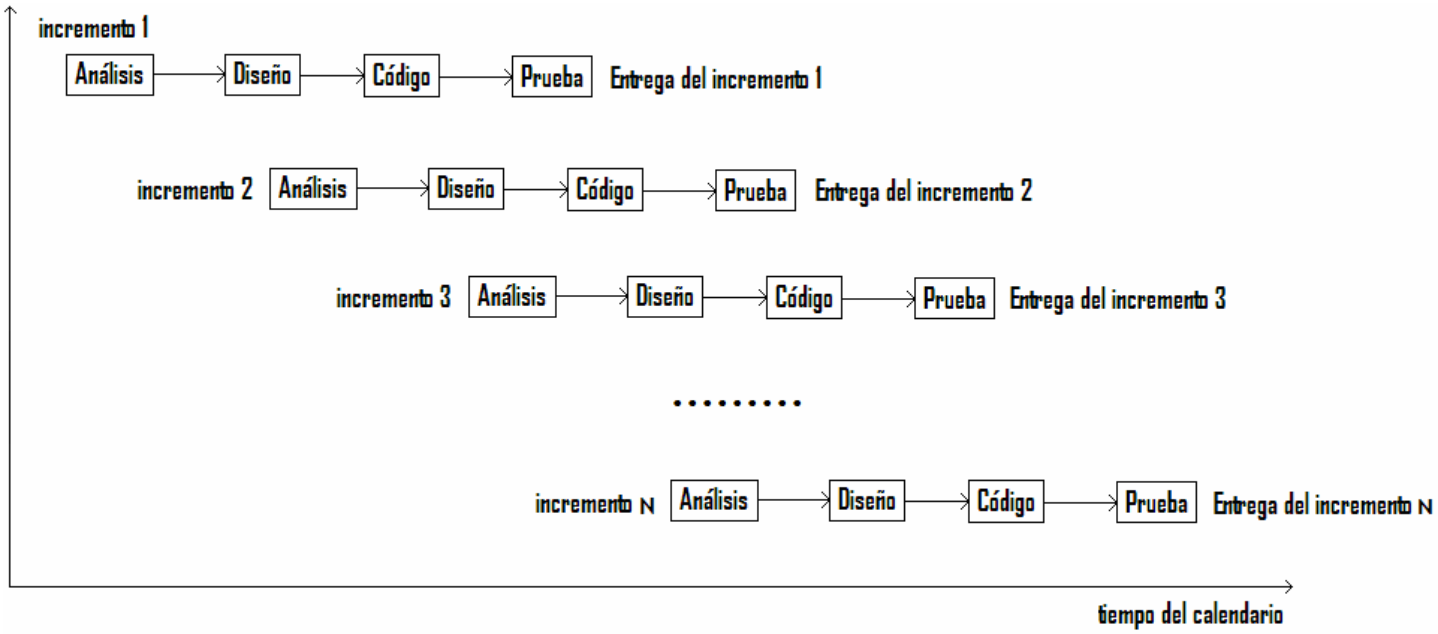
- Utiliza modelado visual: Son utilizados diversos tipos de diagramas para representar la funcionalidad del sistema e ilustrar los casos de uso y las interacciones de la aplicación.

- Verificación continúa de la calidad: Como el Proceso Unificado se caracteriza por el desarrollo de software en diversas iteraciones que van entregando pequeños productos utilizables, se puede ir comprobando la calidad, tanto por parte del desarrollador de la aplicación como por parte del cliente.

- Gestión de requisitos cambiantes: Se van entregando al cliente pequeños productos software que el cliente podrá utilizar y podrá cambiar ciertos requisitos referentes al desarrollo de la aplicación si así lo cree necesario.

Además de estos conceptos también existen otros aspectos de diseño que son convenientes que sean aplicados durante el desarrollo de la aplicación. Éstos son el encapsulamiento, la modularidad y la facilidad de mantenimiento.

Un desarrollo iterativo permite realizar entregas de pequeños productos de software al cliente, lo que en este caso se traduce como poder ver el funcionamiento de la aplicación antes de que esté finalizada. Con esta característica el desarrollador del proyecto podrá ver cuáles son los puntos débiles del diseño implementado antes de que el producto esté terminado y el cliente podrá pedir requisitos a mayores o cambiar alguno. Este modelo incremental refleja la filosofía de proceso de construir una implementación parcial del sistema global y posteriormente ir incrementando la funcionalidad del sistema.



Basándose en el anterior diagrama, en el desarrollo de esta aplicación se han seguido unos determinados pasos. Primeramente se han estudiado todos los casos de uso y los requisitos globales de la aplicación para más adelante establecer un ciclo iterativo e incremental de desarrollo en el que cada iteración está compuesta por los siguientes pasos:

1. Análisis de los objetivos de cada iteración y estimación del tiempo de desarrollo de cada ciclo.

2. Diseño de una arquitectura y un modelado acorde a los objetivos y los casos de uso.

3. Implementación del punto anterior.

4. Pruebas para comprobar el correcto funcionamiento de la implementación y para la integración con el resto del sistema.

La definición y planificación de cada uno de los incrementos se abordará en los siguientes apartados.

Esta metodología es la aplicación reiterada de varias secuencias basadas en el modelo en cascada. Cada aplicación del ciclo constituye un incremento del software y cada incremento puede ser el perfeccionamiento de un incremento anterior o el desarrollo de una nueva funcionalidad no incorporada en los anteriores incrementos.

En el primer incremento de un sistema se debe abordar el núcleo esencial del sistema, los requisitos fundamentales y en posteriores incrementos se abordarán las demás funciones especificadas.

El Proceso Unificado utiliza el Lenguaje Unificado de Modelado (UML) para preparar y reflejar todos los modelos de la aplicación en desarrollo. UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

Una táctica utilizada por la metodología del Proceso Unificado de Desarrollo consiste en dividir cada proyecto en un número de diagramas que lo definen y este estándar nos dice que no es necesario realizar todos los tipos de diagrama, sino únicamente los que se consideren necesarios para cada proyecto en particular.

La idea principal detrás del proceso de desarrollo incremental o mejoramiento iterativo es desarrollar un sistema de programas de manera incremental, permitiéndole al desarrollador sacar ventaja de lo que se ha aprendido a lo largo del desarrollo anterior, incrementando, versiones entregables del sistema. El aprendizaje viene de dos vertientes: el desarrollo del sistema, y su uso (mientras sea posible).

4.2. Análisis de requisitos

La obtención de requisitos es la parte fundamental en la elaboración de un modelo del sistema que se pretende desarrollar. Un requisito es una declaración sobre un producto deseado que especifica qué debería hacer o cómo debería hacerlo. Los requisitos de la aplicación tienen como objetivo definir de manera clara y precisa todas las funcionalidades y restricciones del sistema que se desea construir.

Los requisitos funcionales hacen referencia a las características o servicios que un sistema debe proporcionar desde el punto de vista del usuario. Nos dicen la manera en que el sistema debe reaccionar ante determinadas entradas y cómo debe comportarse en situaciones particulares. Es preferible que estos requisitos sean completos (que recojan la descripción de todos los servicios esperados por el usuario) y consistentes (que no haya contradicciones entre estas especificaciones).

Los requisitos no funcionales son restricciones de los servicios ofrecidos por el sistema (restricciones del entorno o de la implementación, rendimiento, extensibilidad, escalabilidad, fiabilidad, tiempo de respuesta, capacidad de almacenamiento, etc.).

4.2.1. Actores del sistema

Los actores representan a todo lo que necesita intercambiar información con el sistema, es decir, que interactúa con el mismo. Las instancias de los actores son los usuarios; los cuales llevan a cabo un número de operaciones y desarrollan una secuencia de transacciones en comunicación con el sistema. A esta secuencia de acciones se le llama caso de uso. En la aplicación existen dos tipos de actores:

- Usuario normal: Es todo usuario que se conecta a Internet a través del sistema. Las acciones que puede realizar un usuario son conectarse o desconectarse, así como cambiar su contraseña o consultar sus datos de acceso. Un usuario también puede adquirir (comprar) una cuenta si no dispone de ella.

- Administrador: Es todo aquel usuario que puede realizar todas las funcionalidades de los anteriores usuarios y además podrá realizar la gestión de los otros usuarios, añadiendo, modificando, desconectado o borrando usuarios en el caso del administrador de usuarios o puede también administrar el sistema cambiando diferentes opciones como las direcciones IP o los parámetros de calidad de servicio o filtrado en el caso del administrador del sistema

4.3. Ciclo Iterativo

El Desarrollo Iterativo e Incremental consiste en dividir el ciclo de vida del desarrollo de software en varias iteraciones. Cada iteración está compuesta por una serie de pasos, es decir, gestión de requisitos, análisis, diseño, implementación y pruebas. El desarrollo se va efectuando de manera incremental de manera que ya en etapas tempranas del ciclo de vida del desarrollo, existen paquetes entregables de software que son un subconjunto de la futura aplicación.

El ciclo iterativo incremental se ha dividido en varios incrementos o iteraciones, en los cuales cada uno de los nuevos incrementos añade nuevas funcionalidades a la aplicación en desarrollo. Las ventajas que presenta esta división del trabajo en iteraciones son las siguientes:

- El trabajo a desarrollar está claramente dividido y diferenciado en partes más pequeñas y manejables que facilitan su control y organización dentro del proyecto.

- Cada incremento es revisado y utilizado por el cliente, el cual puede estar en desacuerdo con ciertos aspectos de la implementación, siendo una gran ventaja el poder corregir estos aspectos según se va desarrollando el proyecto.

- De otra manera si hubiese que corregir cambios, sería mucho más costoso corregirlos cuando la aplicación estuviera terminada.

- El cliente podría usar una parte de la aplicación en desarrollo mientras que se van desarrollando las siguientes iteraciones, si es que la aplicación fuese muy urgente.

- Cada una de las iteraciones creadas proporciona una experiencia al programador susceptible de ser aplicada a las siguientes iteraciones.

- Al desarrollar una de las sucesivas iteraciones el programador adquiere conocimientos que puede aplicar a las siguientes iteraciones, aprendiendo así de los problemas encontrados y los errores cometidos en iteraciones pasadas.

4.3.1. Incremento cero.

En esta etapa preliminar al análisis de requisitos se procedió a estudiar y analizar el sistema CANSAS de Igalia en el que se basaría el presente proyecto. Como fruto de dicho estudio de características de CANSAS surgirán una serie de requisitos para el inicio de la Iteración primera. Comprende las primeras actividades de análisis general de los requisitos de la aplicación y la formación. A la finalización de esta iteración se tendrá una idea general de la aplicación y los conocimientos técnicos necesarios para la realización del resto del proyecto. Comprende las subactividades de:

- **Análisis Previo:** Estudio de CANSAS, comprensión de su funcionamiento y análisis de sus características que serán los requisitos de la primera iteración.

- **Formación:** Formación en las principales tecnologías a utilizar, desconocidas hasta el momento: RADIUS, LDAP y control avanzando del tráfico en Linux (iptables, tc)

4.3.2. Primer incremento

El objetivo del primer incremento fue construir un sistema partiendo de cero que ofreciese la misma funcionalidad que CANSAS del que ya se ha hablado anteriormente. El principal objetivo fue actualizar todos los componentes software que integran CANSAS a sus últimas versiones y cambiar la distribución Linux base y el kernel por unos más modernos que soportasen el hardware actual.

Se tuvo que parchear y compilar una buena parte del software que hace funcionar al sistema ya que para el presente proyecto se utilizan configuraciones del mismo atípicas que no vienen activadas en los paquetes precompilados que existen en la distribución Ubuntu.

Un caso que merece especial mención es el núcleo de Linux, que necesita ser parcheado para activar el soporte para IMQ[86] (*dispositivo de encolado intermedio)*, ya que esta característica no se incluye en el núcleo oficial de Linux.

4.3.2.1. Requisitos iniciales

Estos son los requisitos que se contemplaron en un principio:

- Se creará un sistema como CANSAS partiendo de cero. Se instalará una distribución moderna de Linux (Ubuntu) y se instalarán y configuraran todos el software necesario

[86] http://www.linuximq.net/

para hacer funcionar el sistema imitando el funcionamiento de CANSAS:

- El sistema dispondrá de dos tarjetas de red. Una conectada a Internet y la otra a un *switch* que ofrecerá conectividad a los usuarios.

- Cuando un usuario intente acceder a Internet el sistema debe redirigirlo a una página de bienvenida donde podrá introducir sus credenciales si dispone de ellas.

- El sistema debe proporcional acceso seguro (encriptación de datos) a Internet a través de una VPN.

- Deben poderse definir políticas de acceso diferentes para cada usuario en cuanto al ancho de banda del que cada uno puede hacer uso.

- El administrador podrá dar de alta usuarios, borrarlos, modificarlos y desconectarlos.

.

- Los datos de los usuarios se almacenaran de forma permanente en una base de datos en el disco duro.

- Se implementara una Interfaz Web para la administración del sistema que permitirá gestionar el alta de usuarios.

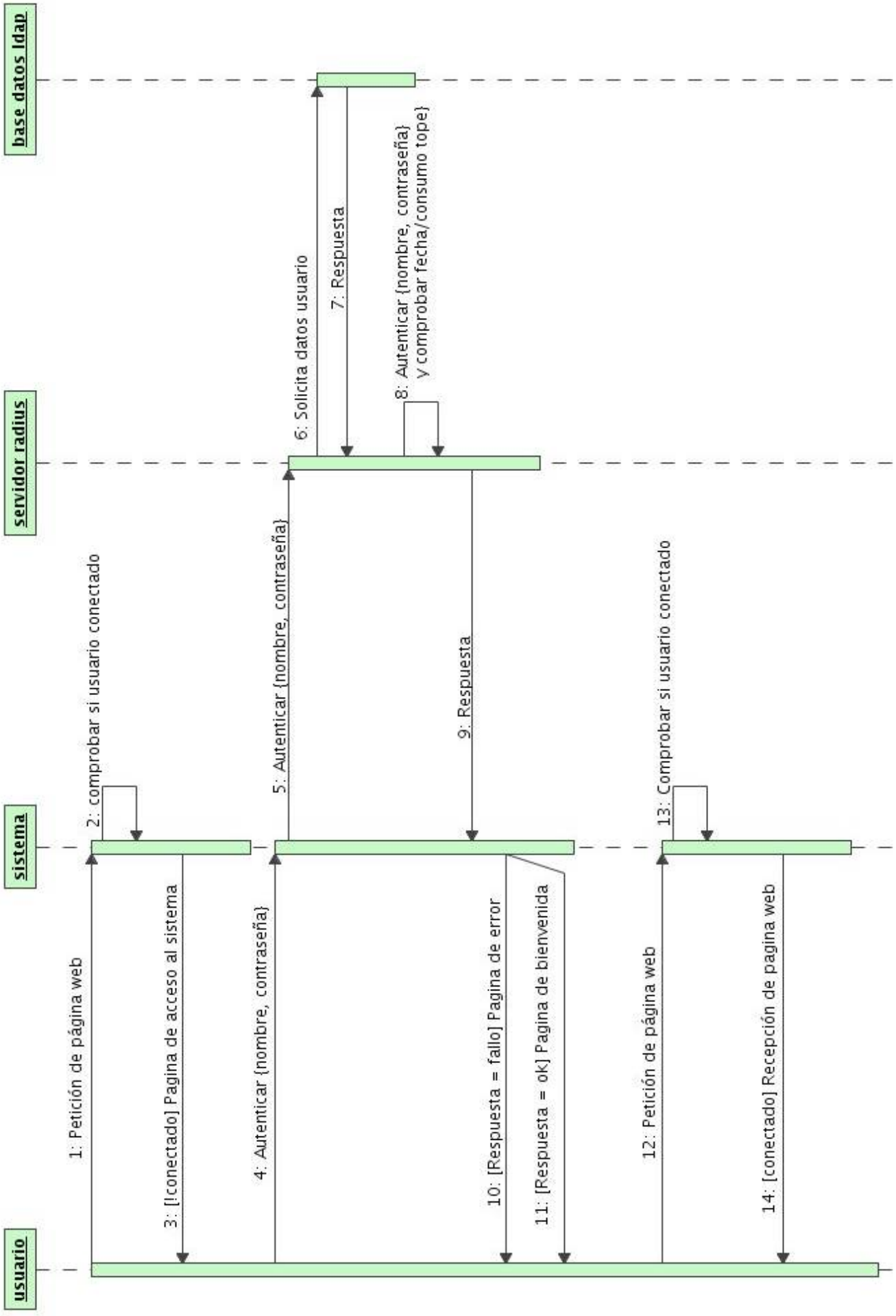
4.3.2.2. Análisis y diseño

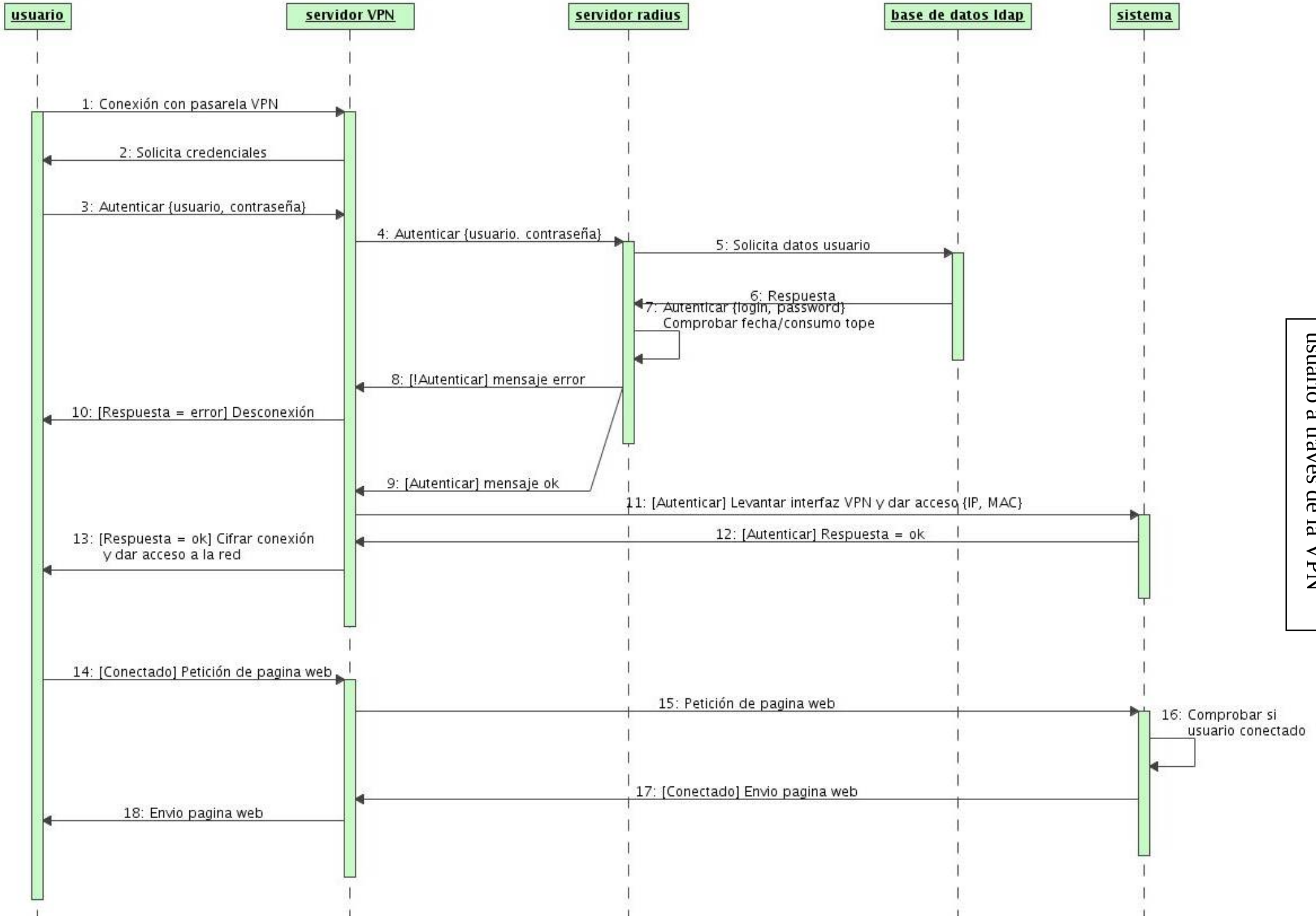
En los siguientes diagramas se muestran los casos de uso del sistema contemplados, así como la secuencia de sucesos cuando un usuario se conecta

4.3.2.2.1. Casos de uso

4.3.2.2.2. Diagramas

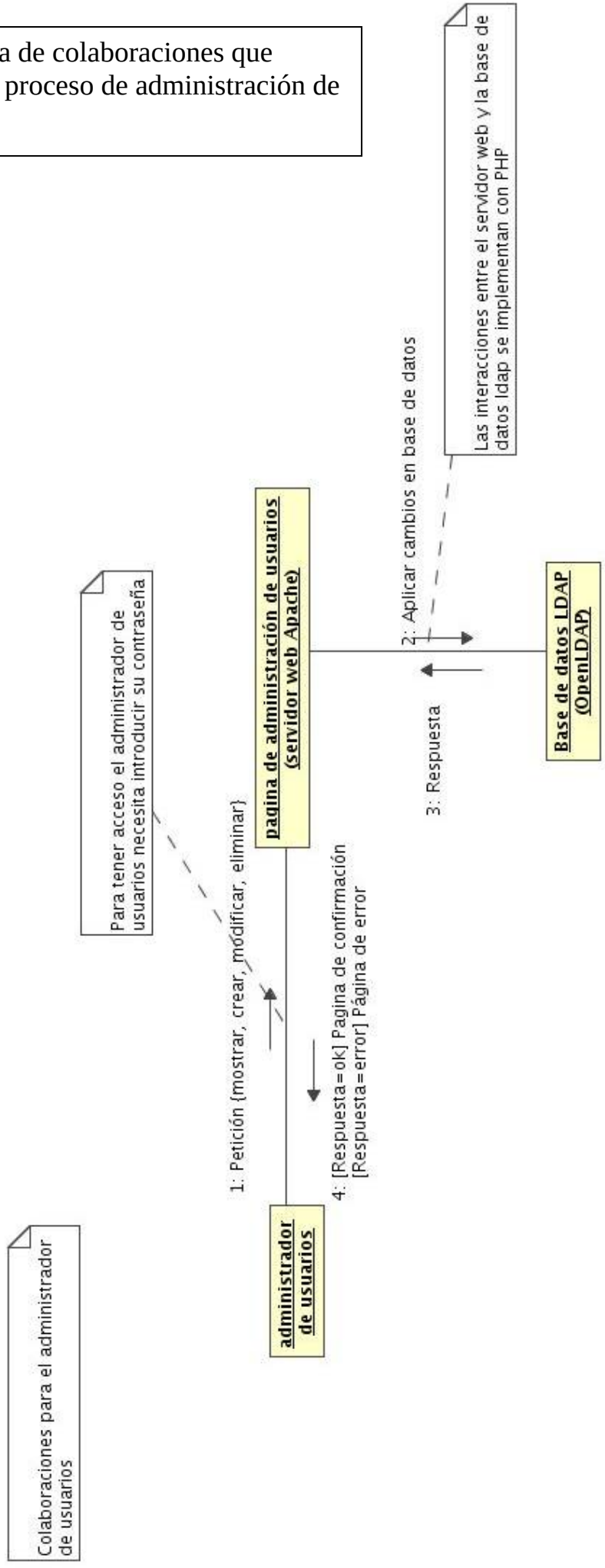
Secuencia de conexión de un usuario a través de la pagina de bienvenida





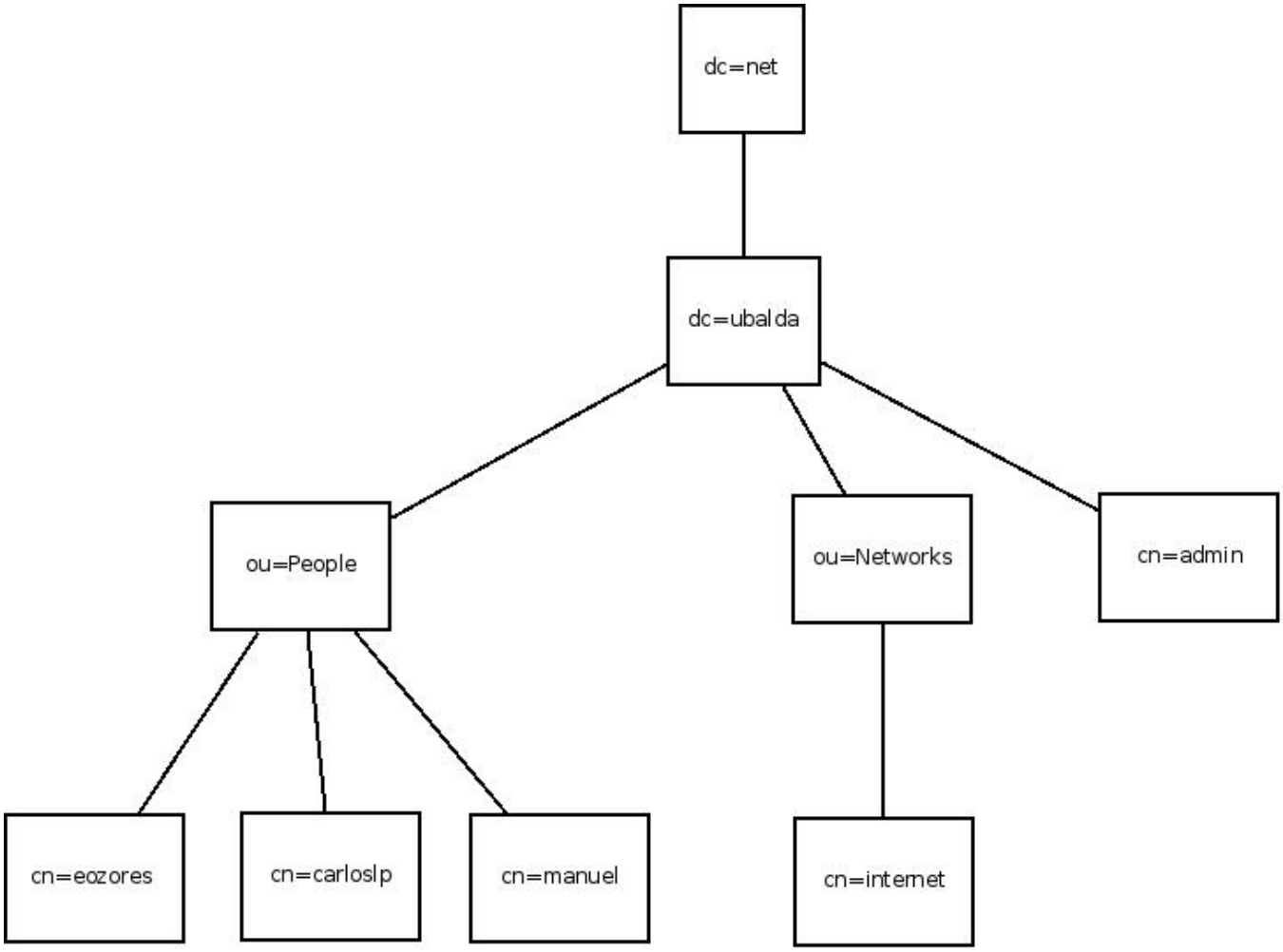
Secuencia de conexión de un usuario a través de la VPN

Diagrama de colaboraciones que ilustra el proceso de administración de usuarios



4.3.2.2.3. Diseño de la base de datos

Para el presente proyecto se ha utilizado una base de datos LDAP, la cual ya hemos estudiado. Se trata de una base de datos no relacional, con estructura jerárquica en forma de árbol, por lo que no se puede realizar un diagrama entidad-relación de la misma. No obstante si podemos mostrar su diseño jerárquico

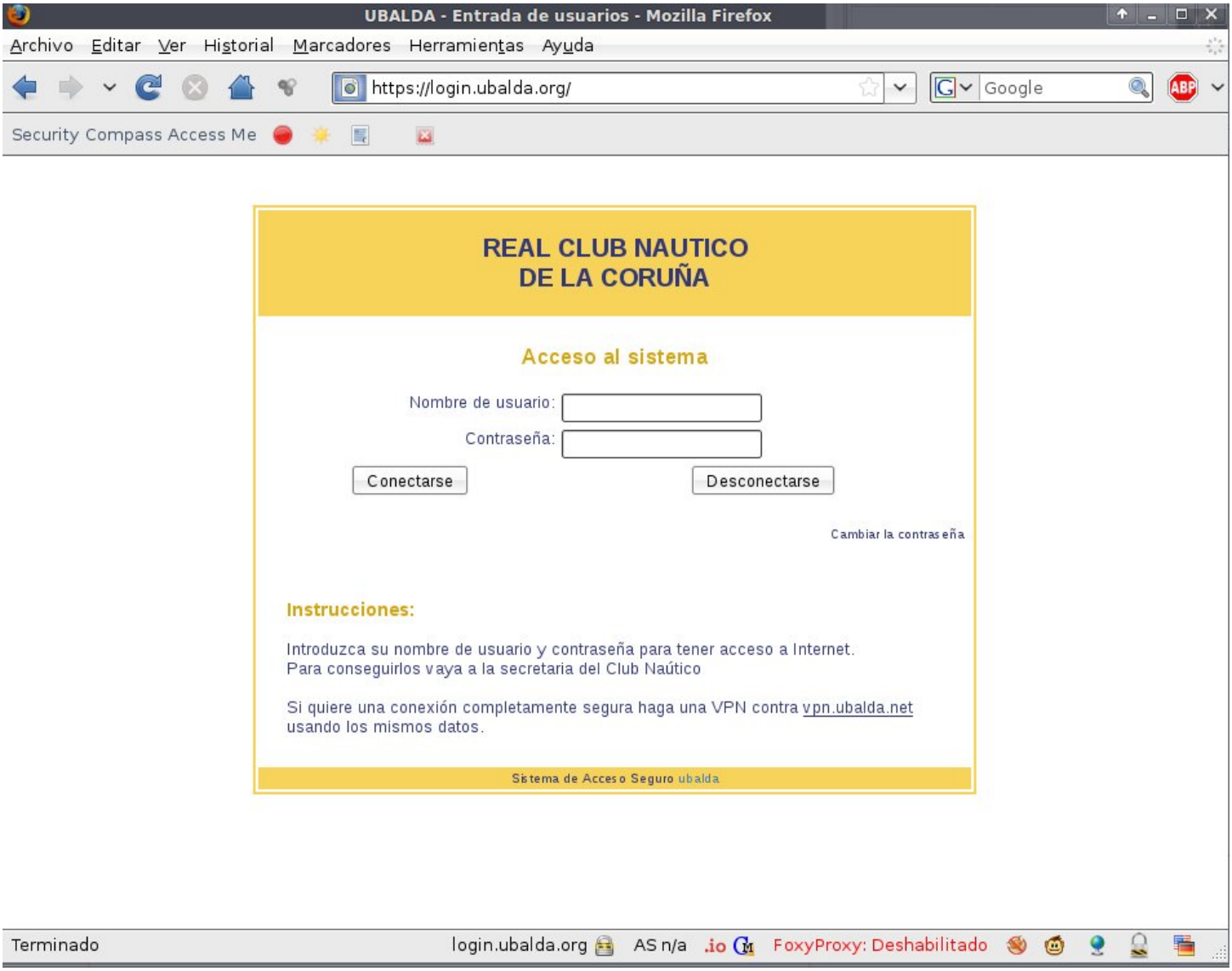


Estructura del directorio LDAP implementado en el presente proyecto

4.3.2.3. Interfaz

Se van a mostrar diferentes capturas de pantalla de la interfaz inicial que se diseño para el sistema, tanto la de usuario como la de administrador

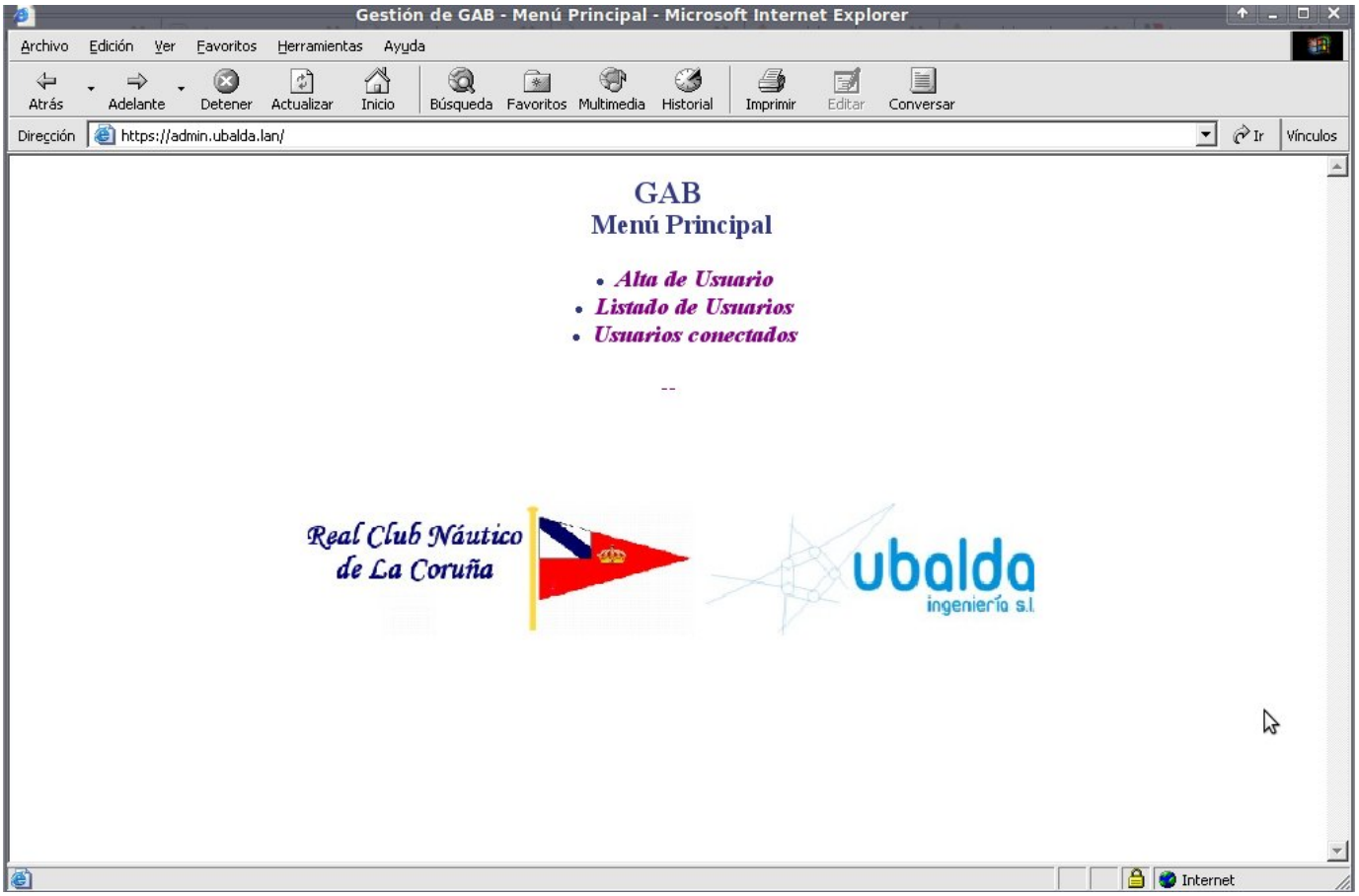
4.3.2.3.1. Interfaz de usuario del sistema

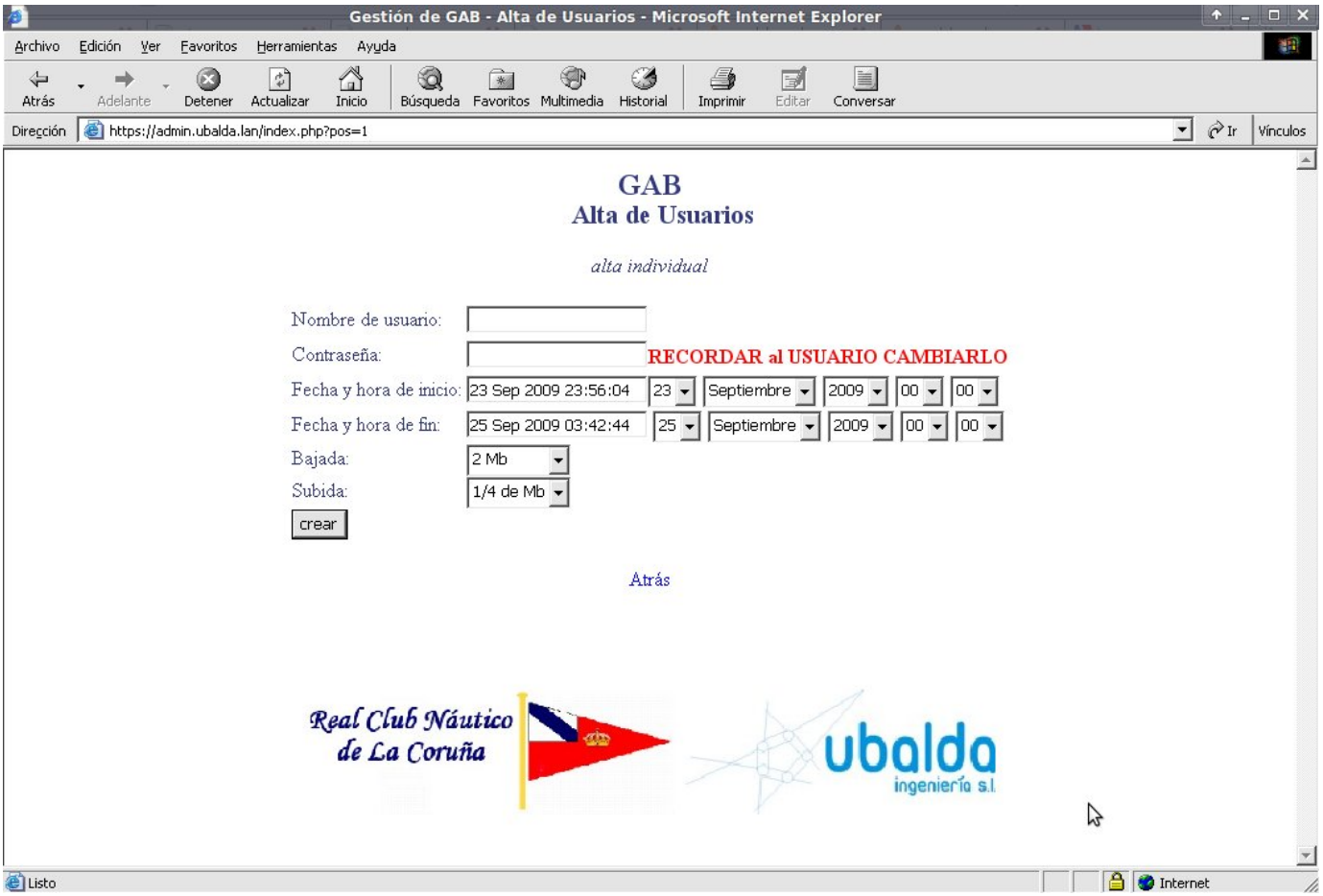


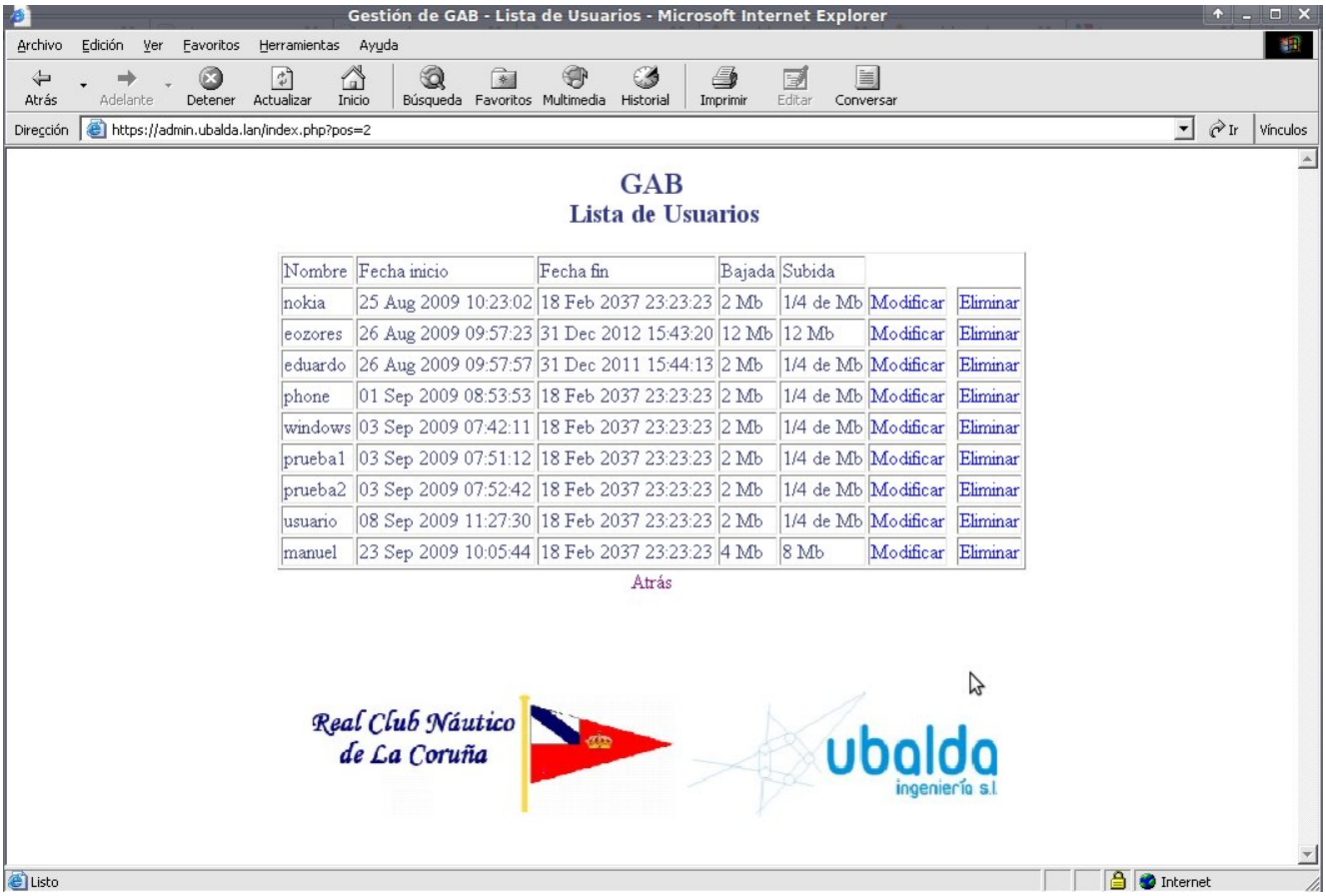
Arriba podemos apreciar la página de bienvenida para la conexión del usuario y abajo la página para cambiar la contraseña.

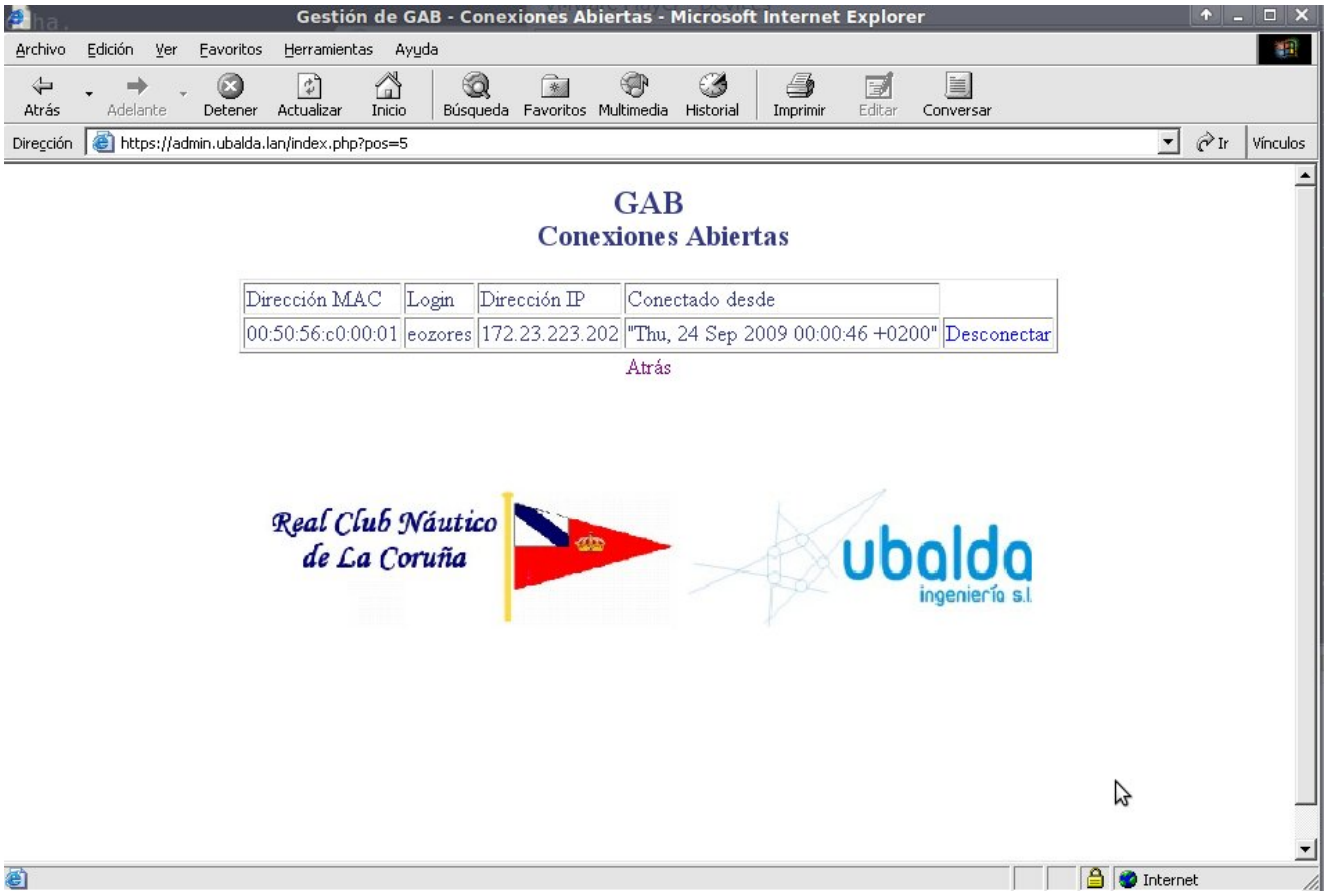
4.3.2.3.2. Interfaz de administrador

Para el acceso a la página de administración es necesaria una contraseña que solo el administrador conoce


Gestión de GAB - Menú Principal - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Adelante Detener Actualizar Inicio Búsqueda Favoritos Multimedia Historial Imprimir Editar Conversar
Dirección https://admin.ubalda.lan/
Ir Vínculos
GAB
Menú Principal
Alta de Usuario
Listado de Usuarios
Usuarios conectados
--
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
Internet


Gestión de GAB - Alta de Usuarios - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Adelante Detener Actualizar Inicio Búsqueda Favoritos Multimedia Historial Imprimir Editar Conversar
Dirección https://admin.ubalda.lan/index.php?pos=1
Ir Vínculos
GAB
Alta de Usuarios
alta individual
Nombre de usuario:
Contraseña:
RECORDAR al USUARIO CAMBIARLO
Fecha y hora de inicio: 23 Sep 2009 23:56:04 23 Septiembre 2009 00 00
Fecha y hora de fin: 25 Sep 2009 03:42:44 25 Septiembre 2009 00 00
Bajada: 2 Mb
Subida: 1/4 de Mb
crear
Atrás
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
Listo
Internet


Gestión de GAB - Lista de Usuarios - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Adelante Detener Actualizar Inicio Búsqueda Favoritos Multimedia Historial Imprimir Editar Conversar
Dirección https://admin.ubalda.lan/index.php?pos=2
Ir Vínculos
GAB
Lista de Usuarios
Nombre | Fecha inicio | Fecha fin | Bajada | Subida
nokia | 25 Aug 2009 10:23:02 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
eozores | 26 Aug 2009 09:57:23 | 31 Dec 2012 15:43:20 | 12 Mb | 12 Mb | Modificar | Eliminar
eduardo | 26 Aug 2009 09:57:57 | 31 Dec 2011 15:44:13 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
phone | 01 Sep 2009 08:53:53 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
windows | 03 Sep 2009 07:42:11 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
prueba1 | 03 Sep 2009 07:51:12 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
prueba2 | 03 Sep 2009 07:52:42 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
usuario | 08 Sep 2009 11:27:30 | 18 Feb 2037 23:23:23 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
manuel | 23 Sep 2009 10:05:44 | 18 Feb 2037 23:23:23 | 4 Mb | 8 Mb | Modificar | Eliminar
Atrás
Real Club Náutico de La Coruña
ubalda ingeniería s.l
Listo
Internet


Gestión de GAB - Conexiones Abiertas - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Adelante Detener Actualizar Inicio Búsqueda Favoritos Multimedia Historial Imprimir Editar Conversar
Dirección https://admin.ubalda.lan/index.php?pos=5
Ir Vínculos
GAB
Conexiones Abiertas
Dirección MAC | Login | Dirección IP | Conectado desde
00:50:56:c0:00:01 | eozores | 172.23.223.202 | "Thu, 24 Sep 2009 00:00:46 +0200" | Desconectar
Atrás
Real Club Náutico de La Coruña
ubalda ingeniería s.l
Internet

4.3.2.4. Pruebas

La principales pruebas en esta etapa del desarrollo fueron principalmente la de estudiar y comprobar la interacción entre los diversos componentes del sistema.

La primera prueba fue la de comprobar el funcionamiento del caso de uso "conexión a través de la página de bienvenida". Se detectó que el sistema no ajustaba bien el ancho de banda debido a un fallo en el *bash script* que controlaba el acceso de un usuario. Tras diversas pruebas se detecto el fallo y fue corregido.

Después se procedió a comprobar el caso de uso "conexión a través de una VPN" y se detecto un fallo crítico que comprometía la estabilidad entera del sistema. Cuando un usuario se conectaba a través de una VPN en pocos minutos el sistema se quedaba completamente *"colgado"* y era necesario reiniciarlo. Se procedió a hacer un estudio detenido de la causa de tal comportamiento anómalo y se concluyo que era un fallo documentado en la implementación del driver IMQ para Linux[87]

También se detectaron fallos en la configuración de la política del firewall del sistema que permitían a un usuario que no estaba conectado acceder a Internet a través de un proxy o usar servicios como el FTP por ejemplo. El sistema solo bloqueaba el acceso al puerto 80 (http) de los usuarios no *logueados*.

4.3.3. Segundo incremento

El segundo incremento se centró en corregir los fallos detectados, sobretodo en lo que atañe a la estabilidad del sistema. Además se incorporaron al sistema nuevas funcionalidades como la posibilidad de dar de alta usuarios fijando como límite el consumo del mismo, ya que antes solo se podía hacer en base a una fecha limite.

[87] http://wiki.nix.hu/cgi-bin/twiki/view/IMQ/ImqFaq#How_stable_is_IMQ

Lo primero que se hizo en este segundo incremento fue estabilizar el sistema corrigiendo los fallos detectados. Una vez hecho esto se implantó el sistema en el **Real Club Náutico de la Coruña** donde lleva funcionando sin dar problemas unos cuantos meses, confirmando que efectivamente se ha conseguido estabilizar el sistema por completo.

Tras comprobar que el sistema era estable y se habían corregido los fallos del *firewall* se procedió con el resto de requisitos.

4.3.3.1. Requisitos

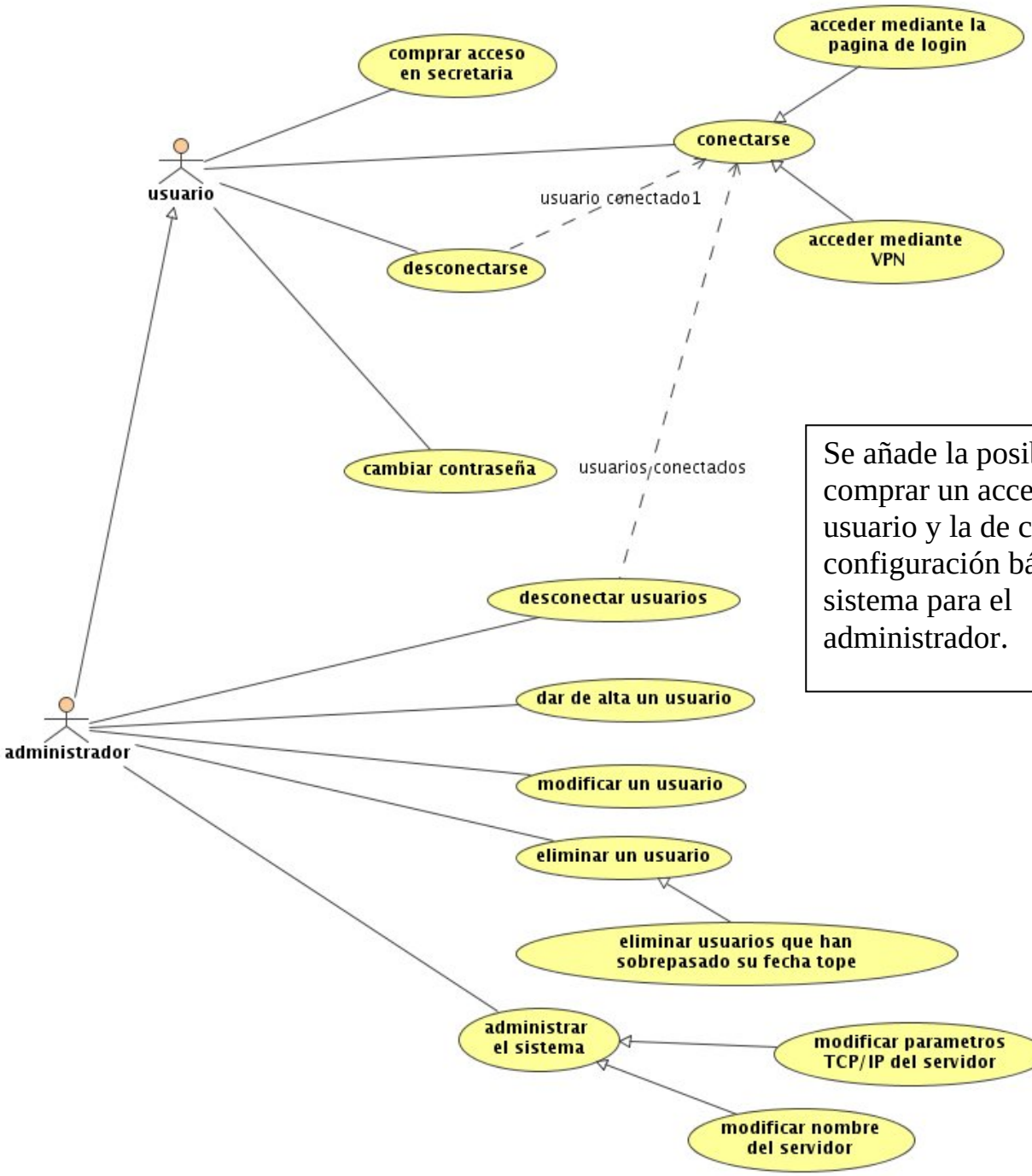
Estos son los requisitos que se contemplaron en un principio:

- Asegurar la completa estabilidad del sistema, corrigiendo los fallos detectados en la etapa anterior.
- Añadir la posibilidad de que se puedan dar de alta usuarios por consumo, dichos usuarios tendrán X Gigabits como límite de uso de su cuenta en vez de la fecha limite anterior
- Rediseñar la política del firewall a fin de garantizar que los usuarios no *logueados* no tienen ningún tipo de acceso a Internet.
- Comprobar que el sistema responde y escala bien cuando se conectan múltiples usuarios y hacen un uso intensivo de la conexión.
- Añadir la posibilidad de que el administrador pueda cambiar la configuración de TCP/IP desde la propia pagina de administración, esto es, la dirección IP del servidor, puerta de enlace y máscara de red.
- Añadir la posibilidad de modificar el nombre del servidor (que aparece en la URL del navegador) desde la página de administración

4.3.3.2. Análisis y diseño

En los siguientes diagramas se muestran los casos de uso del segundo incremento que añade una funcionalidad al administrador.

4.3.3.2.1. Casos de uso



Se añade la posibilidad de comprar un acceso para el usuario y la de cambiar la configuración básica del sistema para el administrador.

4.3.3.2.2. Diagramas

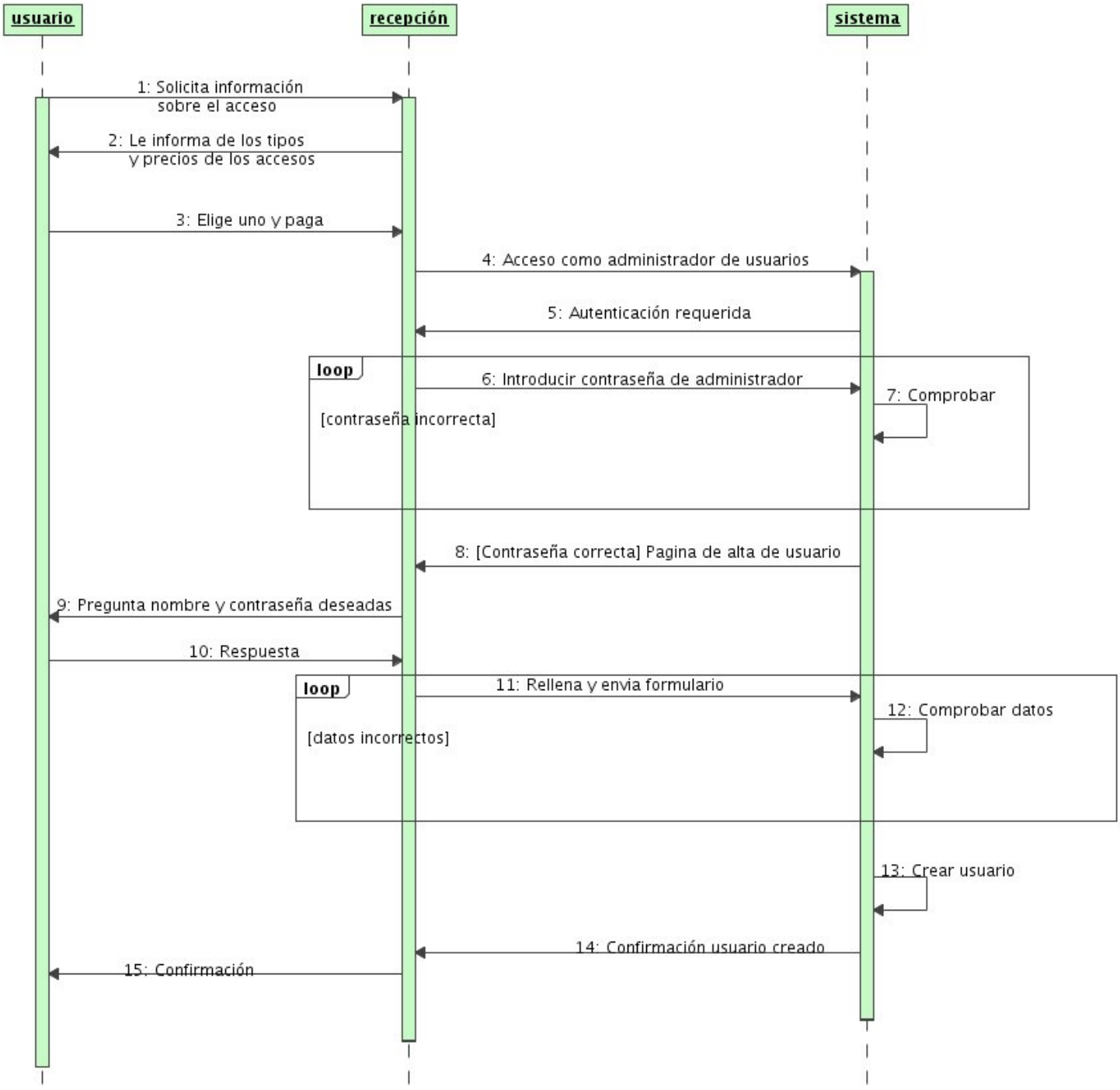


Diagrama de secuencias que ilustra el proceso de comprar una cuenta de usuario en recepción

4.3.3.3. Pruebas

Se realizaron numerosas pruebas de conectividad y tests de velocidad para comprobar que el funcionamiento del sistema era el esperado.

En todo momento el desarrollador se conectaba a Internet a través del gestor de ancho de banda como si fuera un usuario del mismo, de esta forma se contribuía de forma efectiva a probar el funcionamiento del mismo.

De igual forma, como se ha comentado, el sistema se instaló en el Real Club Náutico de Coruña donde esta dando servicio a los usuarios a través de un punto de acceso inalámbrico conectado al mismo. Esta fue tal vez la prueba de fuego para asegurar la estabilidad del sistema.

4.3.4. Tercer Incremento

Tras haber conseguido un sistema completamente estable y funcional en el segundo incremento en el tercero se definen una gran cantidad de requisitos nuevos para el sistema.

También se decide que se rediseñara completamente la interfaz, tanto la de administración como la página de bienvenida. Se añade un nuevo actor al sistema: El administrador del sistema. Ahora existen dos administradores, uno que solo puede encargarse de la configuración de los usuarios y otro que, además, puede configurar los parámetros del sistema.

Debido a la gran cantidad de nuevos requisitos tuvieron que compilarse nuevamente numerosas partes del sistema como el núcleo o las utilidades iptables con los parches que permitían el uso de estas características.

4.3.4.1. Requisitos

Estos son los requisitos que se contemplan en el tercer incremento:

- Rediseño de la página de administración y de la página de bienvenida de usuarios.

- Añadir un “administrador del sistema” con más privilegios que el “administrador de usuarios”

- Posibilidad de dar de alta múltiples usuarios de forma sencilla. Se añade la posibilidad de añadir múltiples usuarios a través de un fichero de Microsoft Excel.

- Añadir soporte para UPnP y calidad de servicio priorizando diferentes tipos de tráfico.

- Añadir protección contra un uso malintencionado del sistema. Protección contra “*flood*”

- Añadir la posibilidad de filtrar el tráfico P2P así como la de bloquear el acceso a la red Interna de la empresa por parte de los usuarios.

- Posibilidad de modificar la pagina de bienvenida desde la propia página de administración de una forma sencilla y amigable.

- Posibilidad de definir redirecciones de puertos (NAT) estáticas para poder controlar los puntos de acceso conectados al sistema desde el exterior.

- Mostrar un aviso al usuario cuando intenta visitar una dirección de Internet que no existe o esta haciendo un uso indebido del sistema.

- Posibilidad de que el usuario consulte sus datos de acceso.

- Posibilidad de que el usuario compre una cuenta para acceder al sistema a través de PayPal de forma automática.

- Posibilidad de conectar el sistema a una máquina de monedas con una impresora de tickets para automatizar el proceso de venta de cuentas.

- Añadir soporte para que los usuarios se puedan conectar de forma inalámbrica utilizando el protocolo WPA Empresarial, de forma que con su nombre de usuario y contraseña se conecten al sistema de forma automática e inalámbrica.

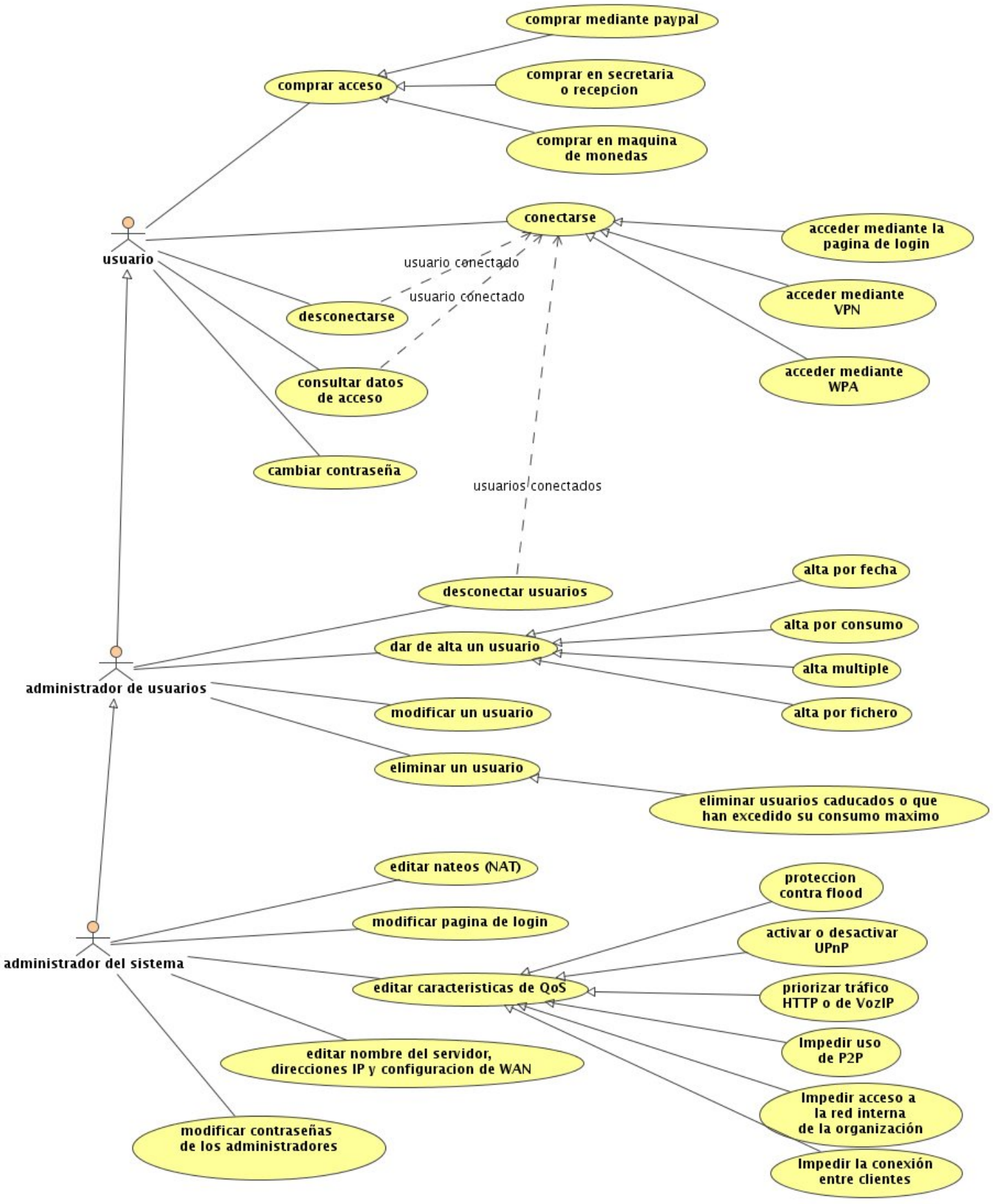
- Añadir soporte para que el sistema funcione con tarjetas WiFi integradas sin la necesidad de un punto de acceso inalámbrico.

- Conseguir configurar doble SSID para la interfaz inalámbrica, uno abierto y otro encriptado con WPA Empresarial (WPA-PEAP).

4.3.4.2. Análisis y diseño

En los siguientes diagramas se muestran los casos de uso del tercer incremento que añade una funcionalidad al administrador.

4.3.4.2.1. Casos de uso



Casos de uso del tercer incremento

4.3.4.2.2. Diagramas

Secuencia que ilustra el proceso de conexión de un usuario a través de la red inalámbrica configurada con encriptación WPA-PEAP (WPA Empresarial)

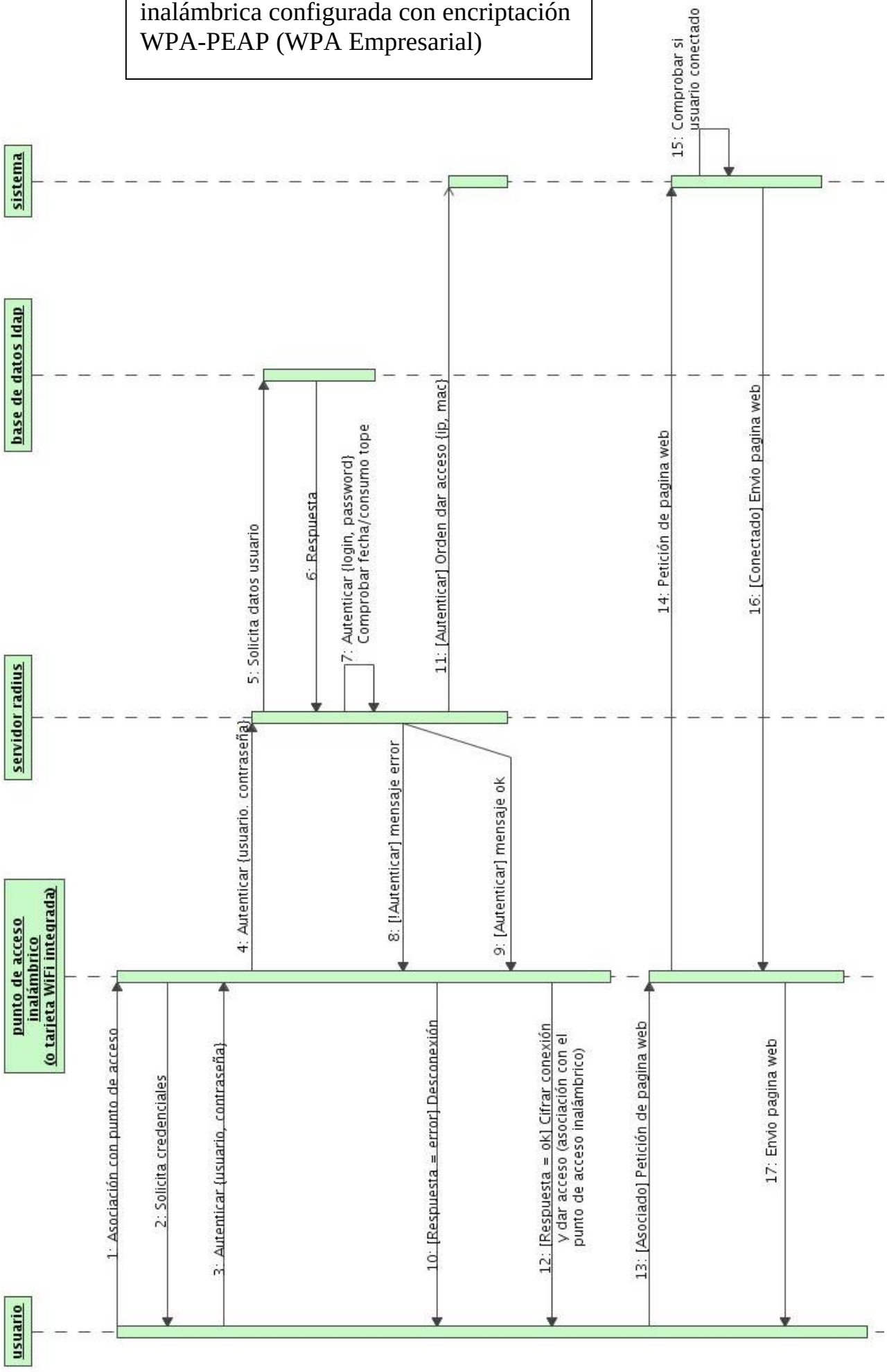


Diagrama de actividades que ilustra el proceso de conexión de un usuario

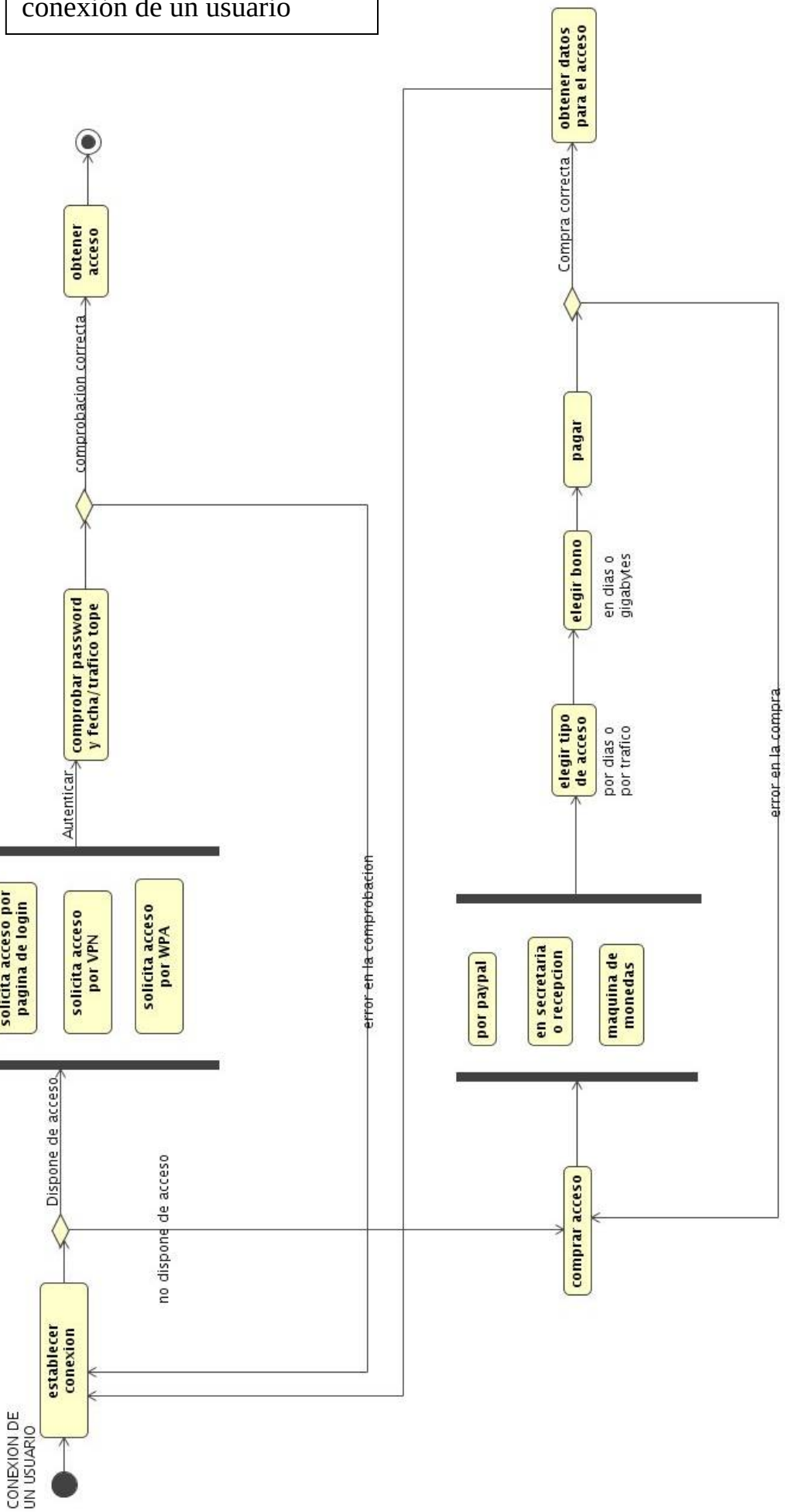


Diagrama de secuencia para el pago por PayPal

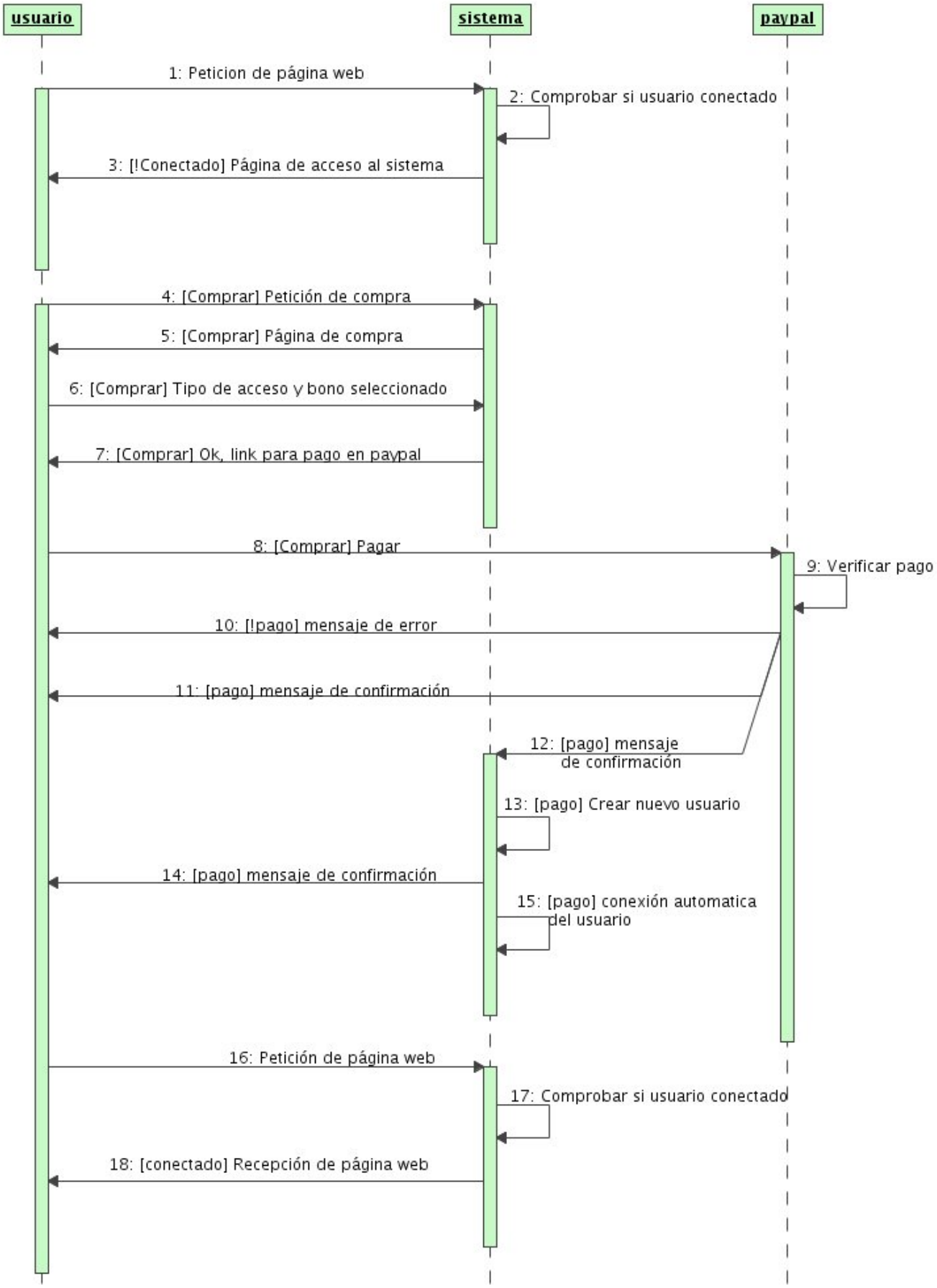
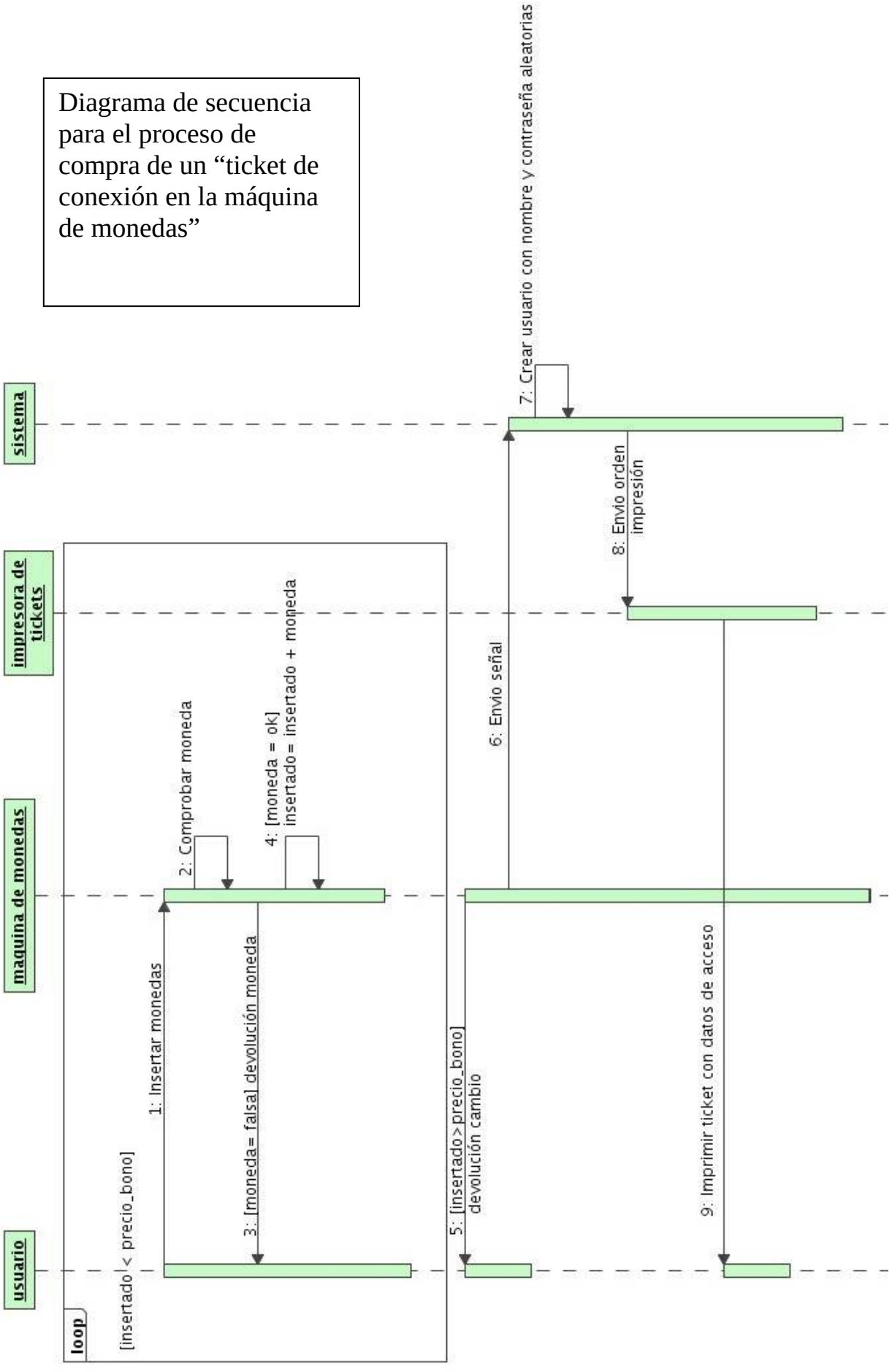
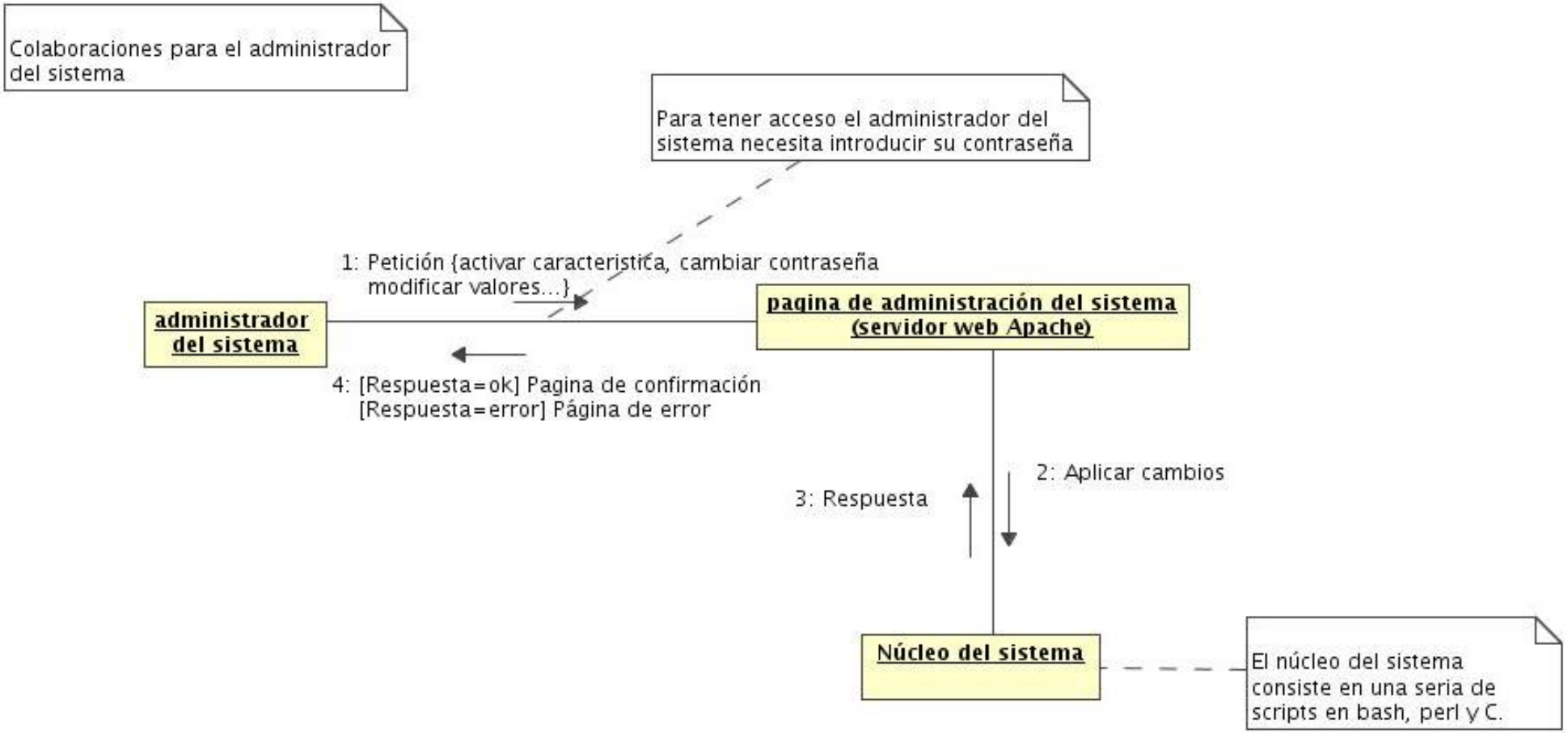
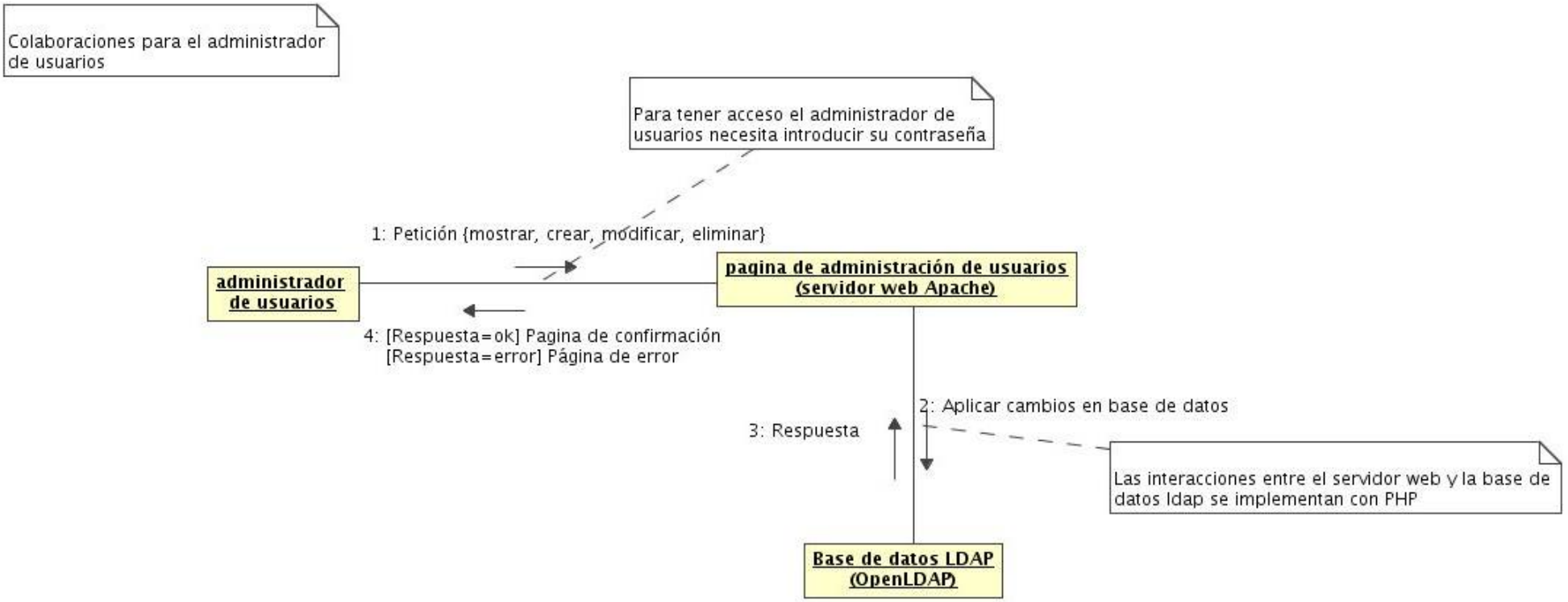
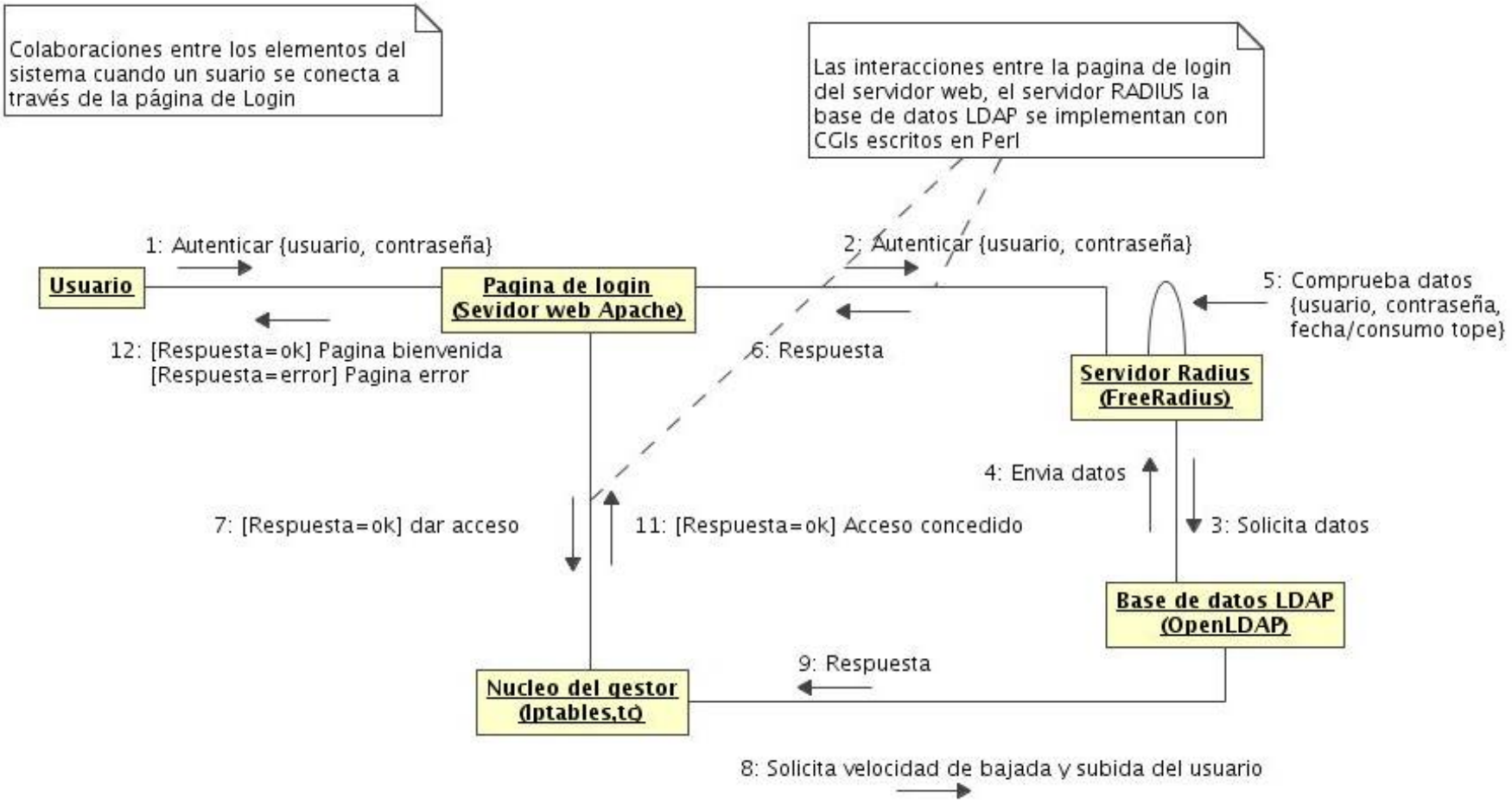


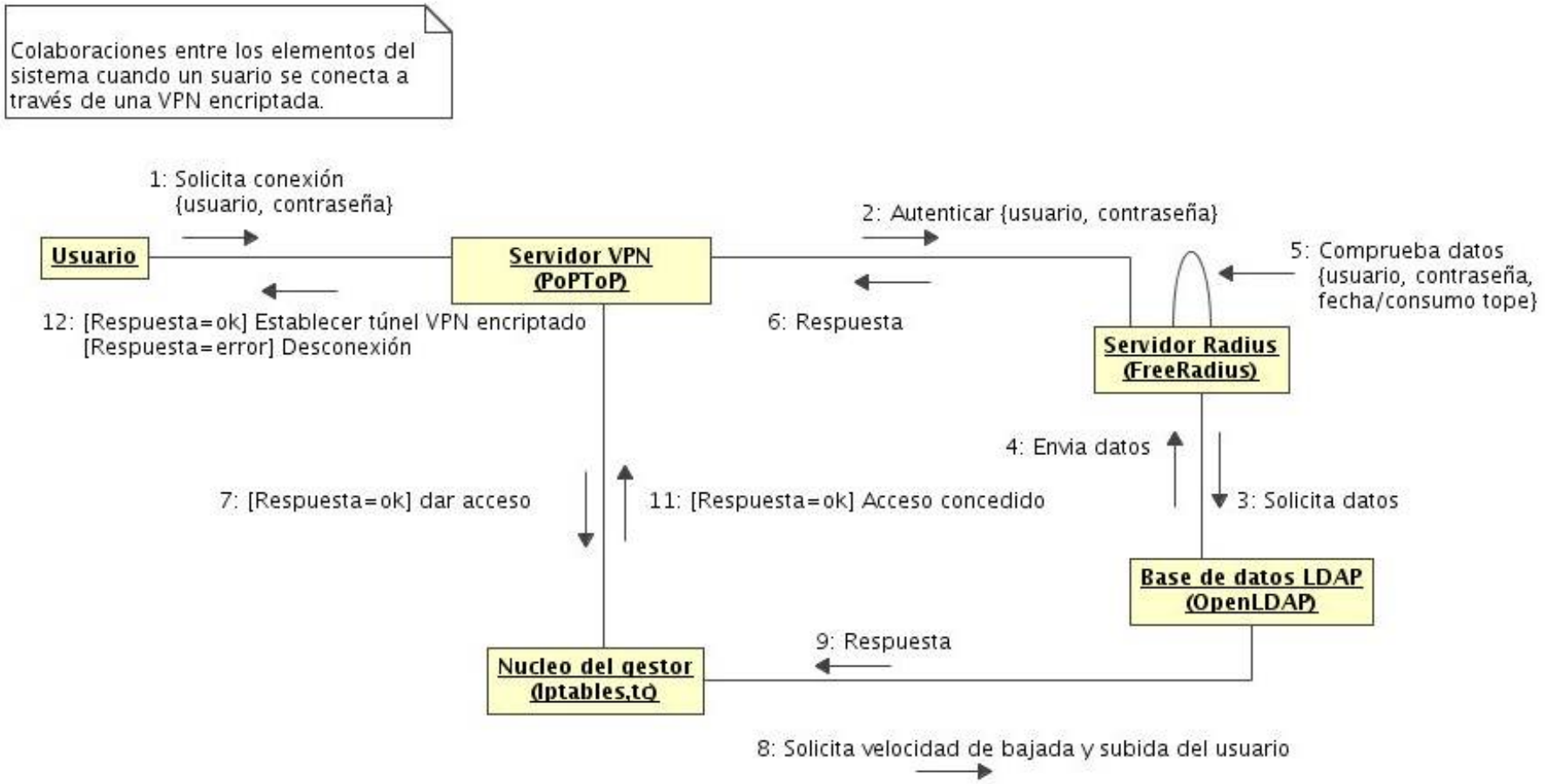
Diagrama de secuencia para el proceso de compra de un “ticket de conexión en la máquina de monedas”

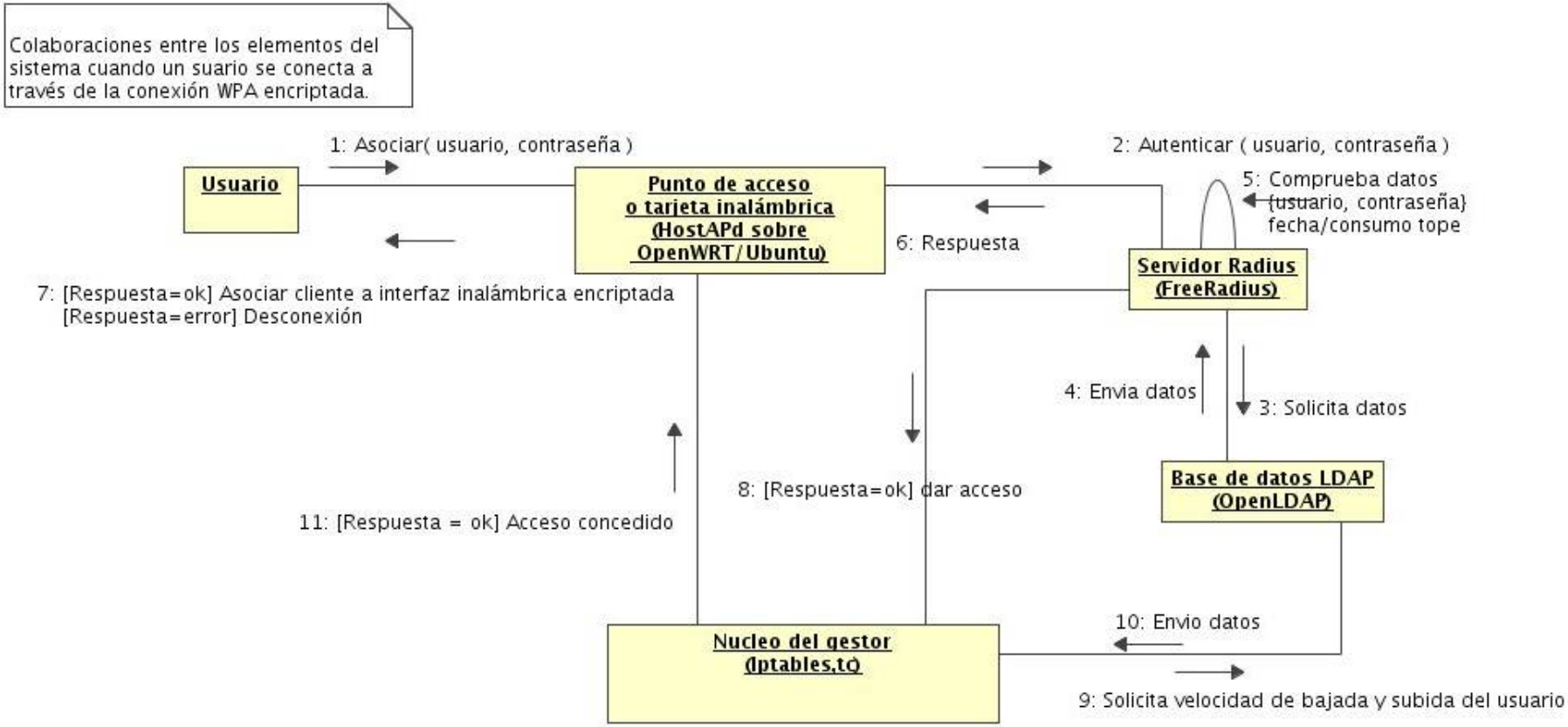


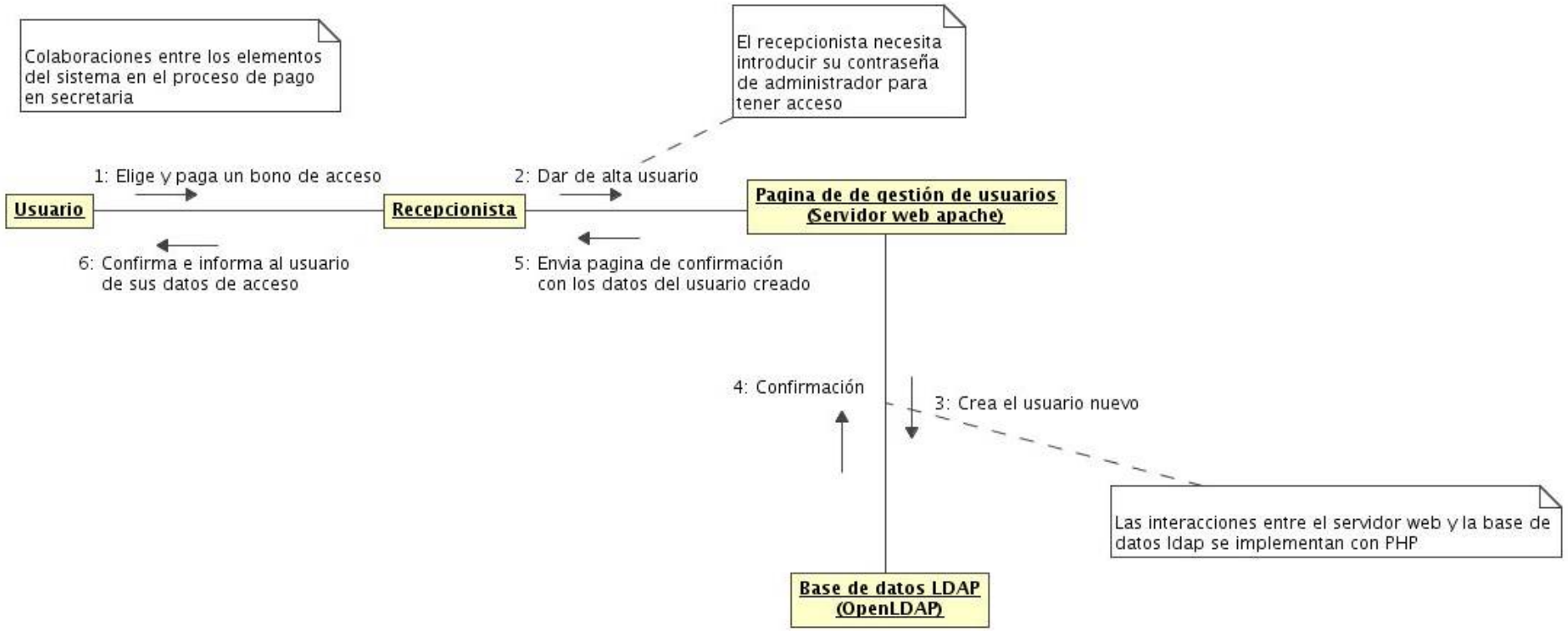

Colaboraciones para el administrador del sistema
Para tener acceso el administrador del sistema necesita introducir su contraseña
1: Petición {activar caracteristica, cambiar contraseña modificar valores...}
administrador del sistema
pagina de administración del sistema (servidor web Apache)
4: [Respuesta=ok] Pagina de confirmación
[Respuesta=error] Página de error
2: Aplicar cambios
3: Respuesta
Núcleo del sistema
El núcleo del sistema consiste en una seria de scripts en bash, perl y C.

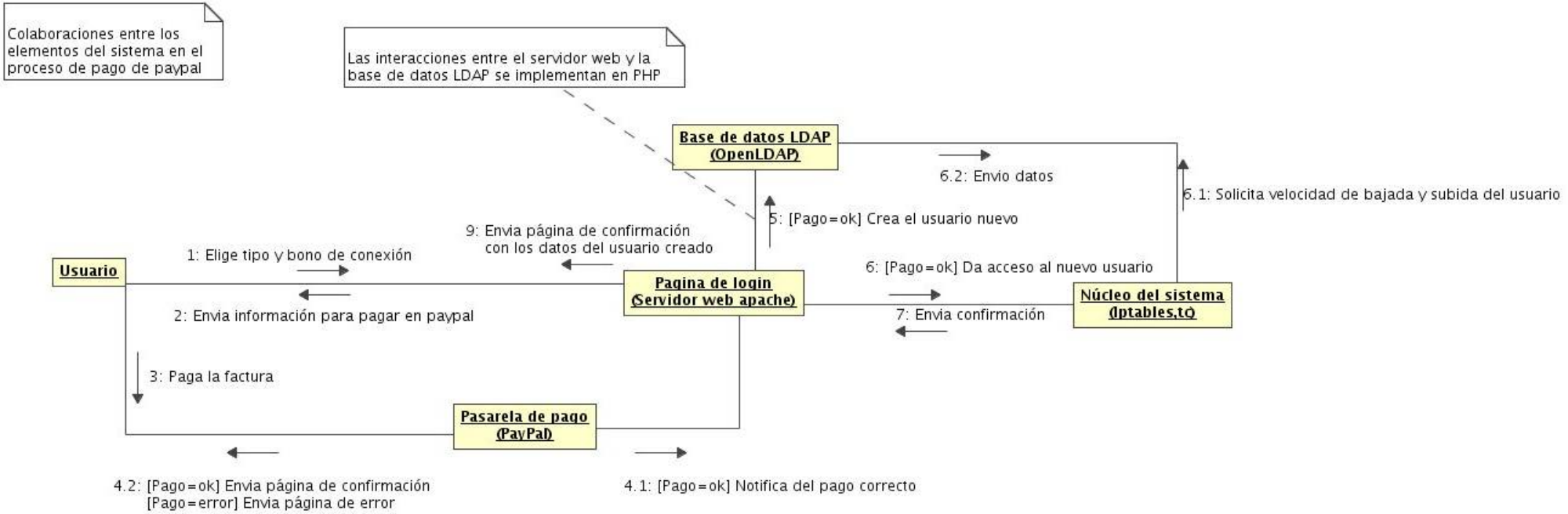

Colaboraciones para el administrador de usuarios
Para tener acceso el administrador de usuarios necesita introducir su contraseña
1: Petición {mostrar, crear, modificar, eliminar}
administrador de usuarios
pagina de administración de usuarios (servidor web Apache)
4: [Respuesta=ok] Pagina de confirmación
[Respuesta=error] Página de error
2: Aplicar cambios en base de datos
3: Respuesta
Las interacciones entre el servidor web y la base de datos ldap se implementan con PHP
Base de datos LDAP (OpenLDAP)

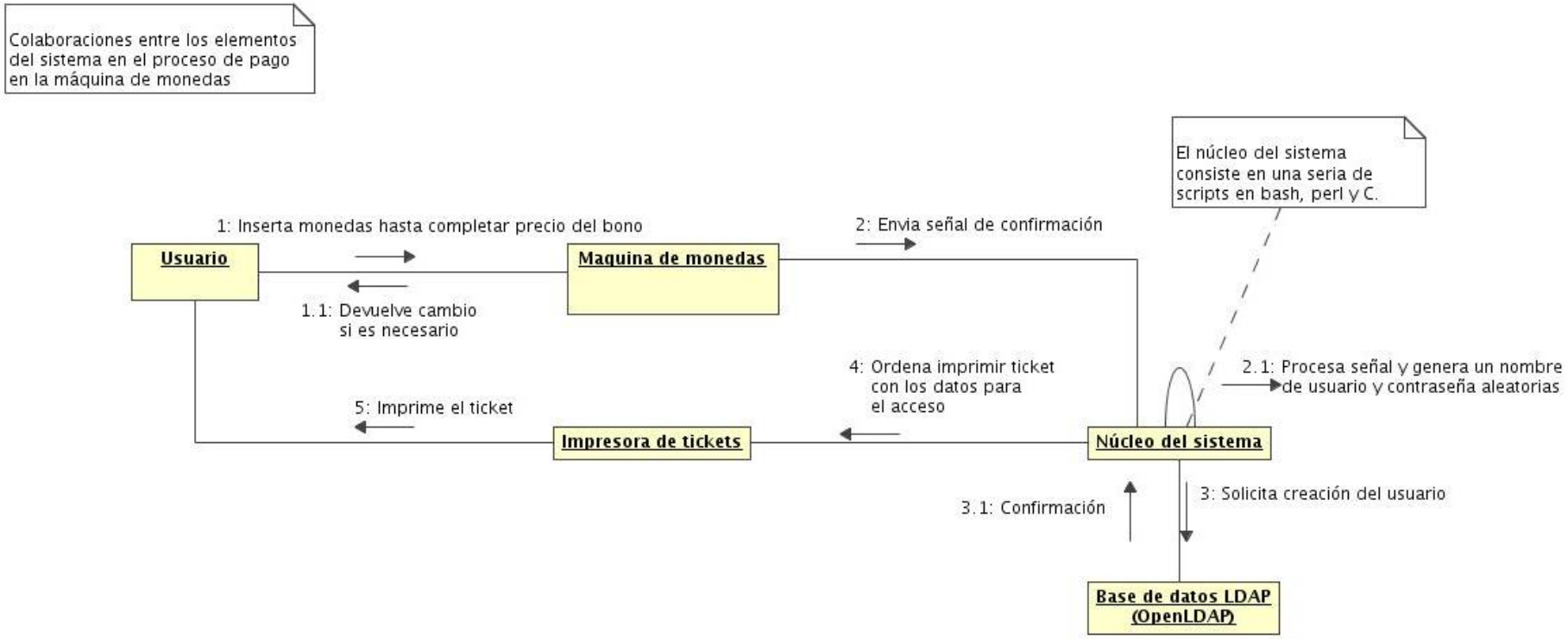

Colaboraciones entre los elementos del sistema cuando un suario se conecta a través de la página de Login
Las interacciones entre la pagina de login del servidor web, el servidor RADIUS la base de datos LDAP se implementan con CGIs escritos en Perl
1: Autenticar {usuario, contraseña}
2: Autenticar {usuario, contraseña}
Usuario
Pagina de login (Sevidor web Apache)
5: Comprueba datos {usuario, contraseña, fecha/consumo tope}
12: [Respuesta=ok] Pagina bienvenida
[Respuesta=error] Pagina error
6: Respuesta
Servidor Radius (FreeRadius)
4: Envia datos
7: [Respuesta=ok] dar acceso
11: [Respuesta=ok] Acceso concedido
3: Solicita datos
Base de datos LDAP (OpenLDAP)
9: Respuesta
Nucleo del gestor (Iptables,tc)
8: Solicita velocidad de bajada y subida del usuario

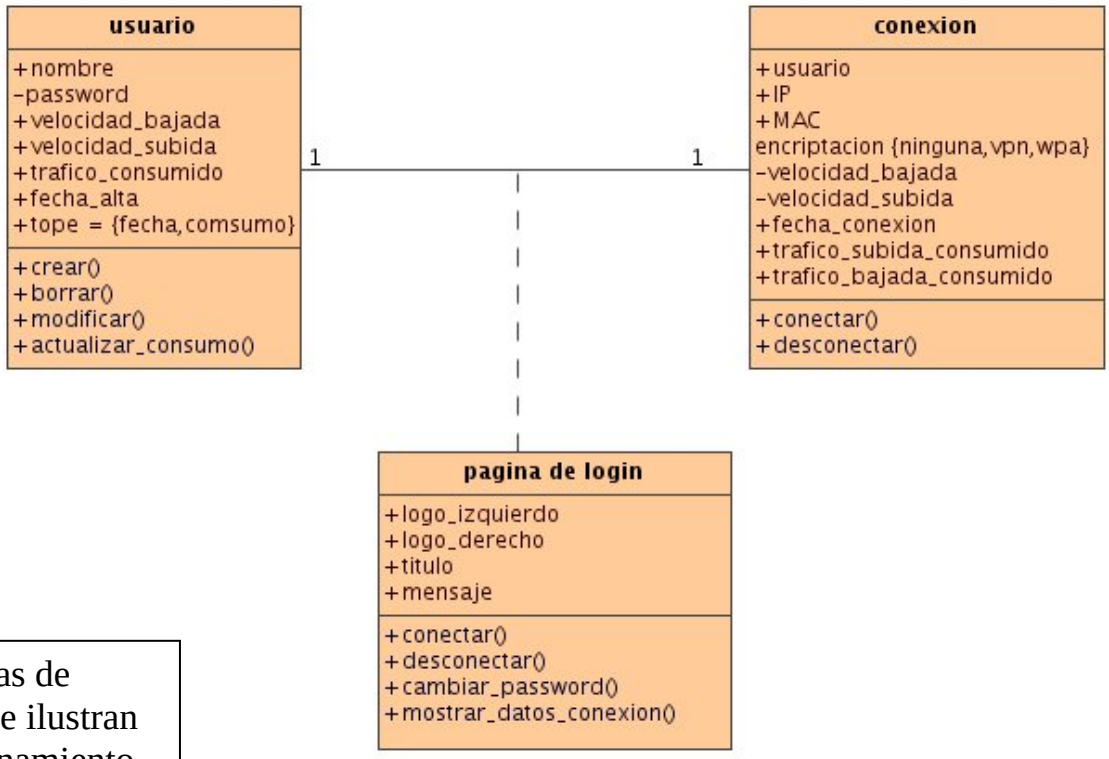

Colaboraciones entre los elementos del sistema cuando un suario se conecta a través de una VPN encriptada.
1: Solicita conexión {usuario, contraseña}
Usuario
Servidor VPN (PoPToP)
2: Autenticar {usuario, contraseña}
5: Comprueba datos {usuario, contraseña, fecha/consumo tope}
12: [Respuesta=ok] Establecer túnel VPN encriptado
[Respuesta=error] Desconexión
6: Respuesta
Servidor Radius (FreeRadius)
4: Envia datos
7: [Respuesta=ok] dar acceso
11: [Respuesta=ok] Acceso concedido
3: Solicita datos
Base de datos LDAP (OpenLDAP)
9: Respuesta
Nucleo del gestor (Iptables,tc)
8: Solicita velocidad de bajada y subida del usuario


Colaboraciones entre los elementos del sistema cuando un suario se conecta a través de la conexión WPA encriptada.
1: Asociar(usuario, contraseña)
Usuario
Punto de acceso o tarjeta inalámbrica (HostAPd sobre OpenWRT/Ubuntu)
2: Autenticar (usuario, contraseña)
5: Comprueba datos (usuario, contraseña) fecha/consumo tope
6: Respuesta
Servidor Radius (FreeRadius)
7: [Respuesta=ok] Asociar cliente a interfaz inalámbrica encriptada
[Respuesta=error] Desconexión
4: Envia datos
3: Solicita datos
8: [Respuesta=ok] dar acceso
Base de datos LDAP (OpenLDAP)
11: [Respuesta = ok] Acceso concedido
10: Envio datos
Nucleo del gestor (Iptables,tc)
9: Solicita velocidad de bajada y subida del usuario

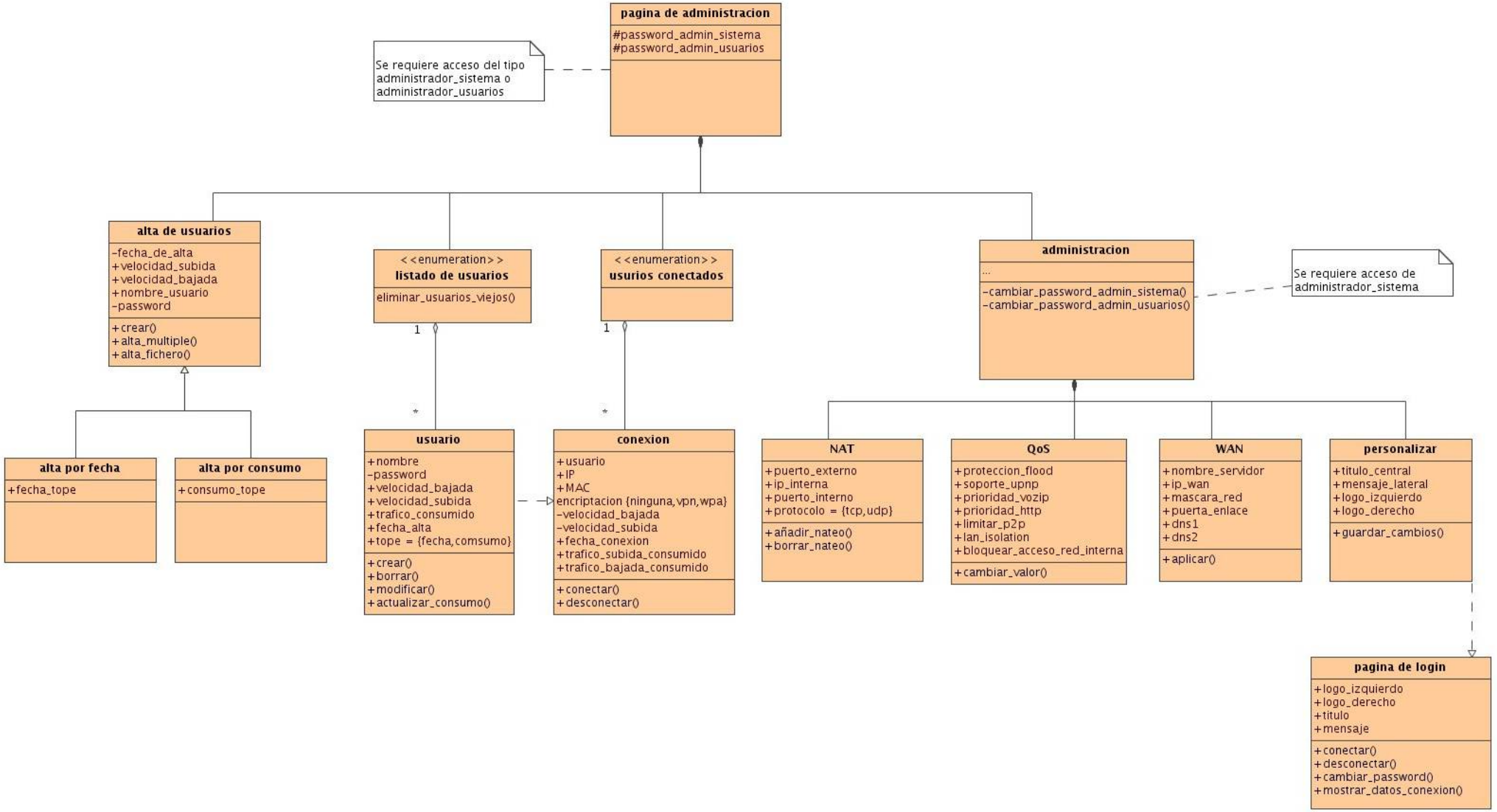

Colaboraciones entre los elementos del sistema en el proceso de pago en secretaria
El recepcionista necesita introducir su contraseña de administrador para tener acceso
1: Elige y paga un bono de acceso
2: Dar de alta usuario
Usuario
Recepcionista
Pagina de de gestión de usuarios (Servidor web apache)
6: Confirma e informa al usuario de sus datos de acceso
5: Envia pagina de confirmación con los datos del usuario creado
4: Confirmación
3: Crea el usuario nuevo
Las interacciones entre el servidor web y la base de datos ldap se implementan con PHP
Base de datos LDAP (OpenLDAP)


Colaboraciones entre los elementos del sistema en el proceso de pago de paypal
Las interacciones entre el servidor web y la base de datos LDAP se implementan en PHP
Base de datos LDAP (OpenLDAP)
6.2: Envio datos
6.1: Solicita velocidad de bajada y subida del usuario
5: [Pago=ok] Crea el usuario nuevo
9: Envia página de confirmación con los datos del usuario creado
1: Elige tipo y bono de conexión
Usuario
6: [Pago=ok] Da acceso al nuevo usuario
Pagina de login (Servidor web apache)
Núcleo del sistema (Iptables,tc)
2: Envia información para pagar en paypal
7: Envia confirmación
3: Paga la factura
Pasarela de pago (PayPal)
4.2: [Pago=ok] Envia página de confirmación
[Pago=error] Envia página de error
4.1: [Pago=ok] Notifica del pago correcto


Colaboraciones entre los elementos del sistema en el proceso de pago en la máquina de monedas
El núcleo del sistema consiste en una seria de scripts en bash, perl y C.
1: Inserta monedas hasta completar precio del bono
2: Envia señal de confirmación
Usuario
Maquina de monedas
1.1: Devuelve cambio si es necesario
2.1: Procesa señal y genera un nombre de usuario y contraseña aleatorias
4: Ordena imprimir ticket con los datos para el acceso
5: Imprime el ticket
Impresora de tickets
Núcleo del sistema
3.1: Confirmación
3: Solicita creación del usuario
Base de datos LDAP (OpenLDAP)



Diagramas de clases que ilustran el funcionamiento interno del sistema


pagina de administracion
#password_admin_sistema
#password_admin_usuarios
Se requiere acceso del tipo administrador_sistema o administrador_usuarios
alta de usuarios
-fecha_de_alta
+velocidad_subida
+velocidad_bajada
+nombre_usuario
-password
+crear()
+alta_multiple()
+alta_fichero()
<<enumeration>>
listado de usuarios
eliminar_usuarios_viejos()
<<enumeration>>
usurios conectados
administracion
...
-cambiar_password_admin_sistema()
-cambiar_password_admin_usuarios()
Se requiere acceso de administrador_sistema
alta por fecha
+fecha_tope
alta por consumo
+consumo_tope
usuario
+nombre
-password
+velocidad_bajada
+velocidad_subida
+trafico_consumido
+fecha_alta
+tope = {fecha, comsumo}
+crear()
+borrar()
+modificar()
+actualizar_consumo()
conexion
+usuario
+IP
+MAC
encriptacion {ninguna, vpn, wpa}
-velocidad_bajada
-velocidad_subida
+fecha_conexion
+trafico_subida_consumido
+trafico_bajada_consumido
+conectar()
+desconectar()
NAT
+puerto_externo
+ip_interna
+puerto_interno
+protocolo = {tcp, udp}
+añadir_nateo()
+borrar_nateo()
QoS
+proteccion_flood
+soporte_upnp
+prioridad_vozip
+prioridad_http
+limitar_p2p
+lan_isolation
+bloquear_acceso_red_interna
+cambiar_valor()
WAN
+nombre_servidor
+ip_wan
+mascara_red
+puerta_enlace
+dns1
+dns2
+aplicar()
personalizar
+titulo_central
+mensaje_lateral
+logo_izquierdo
+logo_derecho
+guardar_cambios()
pagina de login
+logo_izquierdo
+logo_derecho
+titulo
+mensaje
+conectar()
+desconectar()
+cambiar_password()
+mostrar_datos_conexion()
1
*
1
*

4.3.4.3. Diseño de la Interfaz.

4.3.4.3.1. Modelo MVC

La interfaz del sistema, tanto la página Web de administración como la de usuario han sido completamente rediseñadas en el tercer incremento. Para su diseño se ha utilizado el patrón **MVC** (Modelo, Vista, Controlador), que es el patrón arquitectural utilizado para la implementación de esta aplicación.

La Interfaz de esta aplicación está diseñada según el patrón arquitectónico **MVC** que la separa en tres capas:

- **Modelo**: Tiene relación con los objetos de la aplicación. Es la capa encargada de la persistencia de los objetos de la aplicación. Proporciona una serie de fachadas con las que la capa superior (el controlador) puede ejecutar las acciones del modelo.

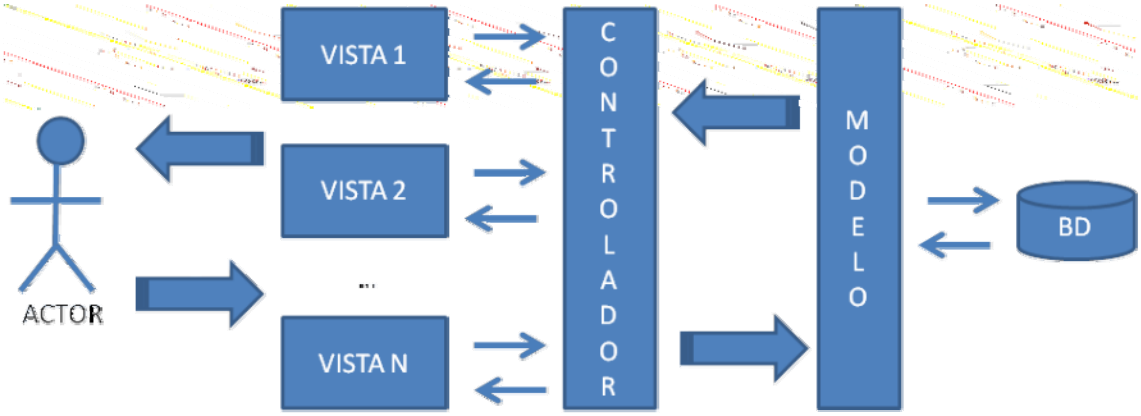
- **Controlador**: Es un mediador entre la capa superior de la aplicación (Vista) y la capa inferior (Modelo). Recibe las peticiones de la capa vista, las interpreta y les aplica una lógica y comprobaciones para después realizar la correspondiente llamada al modelo que obtendrá los resultados. Una vez obtenidos los resultados de la capa modelo, los trasforma y los devuelve a la capa vista para que esta los muestre.

- **Vista**: Es la capa encargada de tratar con los usuarios. Permite a los usuarios realizar peticiones y mostrar los resultados obtenidos. Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado (ya que existe una clara separación entre ellas) y luego unirlas en tiempo de ejecución. Si uno de los componentes, posteriormente, se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas.

El patrón MVC tiene las siguientes ventajas:

- Existe un API muy bien definido. El Modelo, la Vista o el Controlador podrán ser reemplazados por cualquier persona que use el API sin dificultad.
- La conexión existente entre el modelo y las Vistas es dinámica, se produce en tiempo de ejecución, no en tiempo de compilación.
- El modelo y el controlador pueden ser usados con distintas vistas.

Una ilustración esquemática del MVC es la que se expone a continuación:

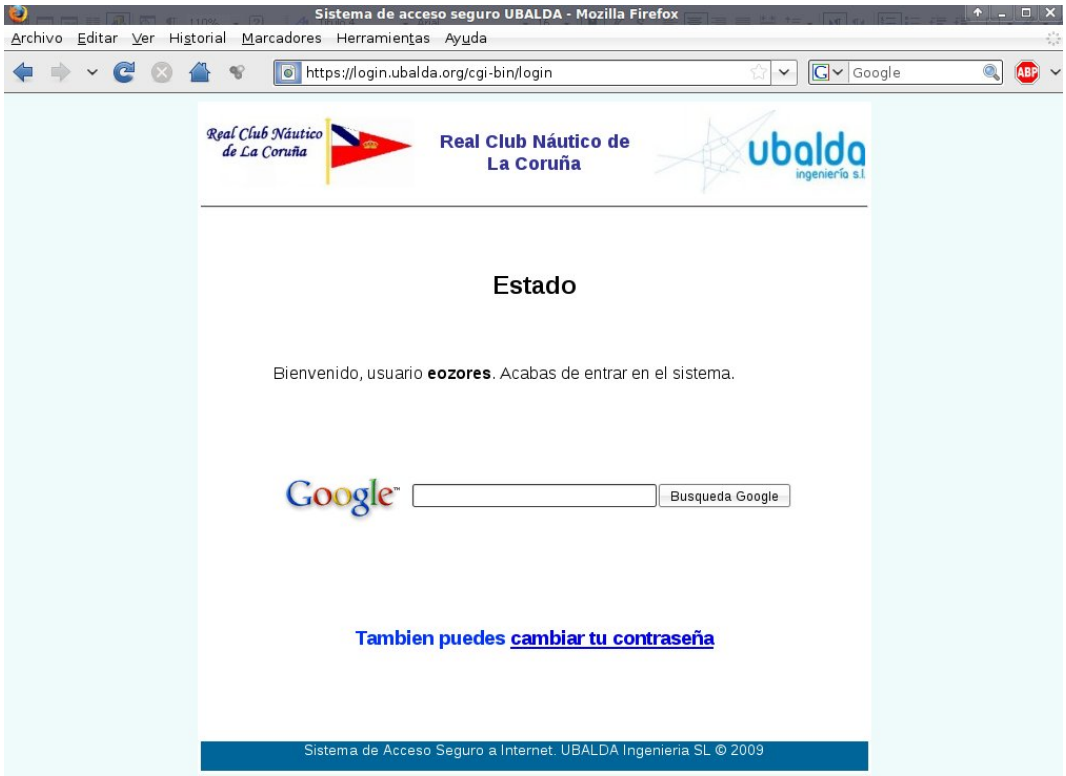


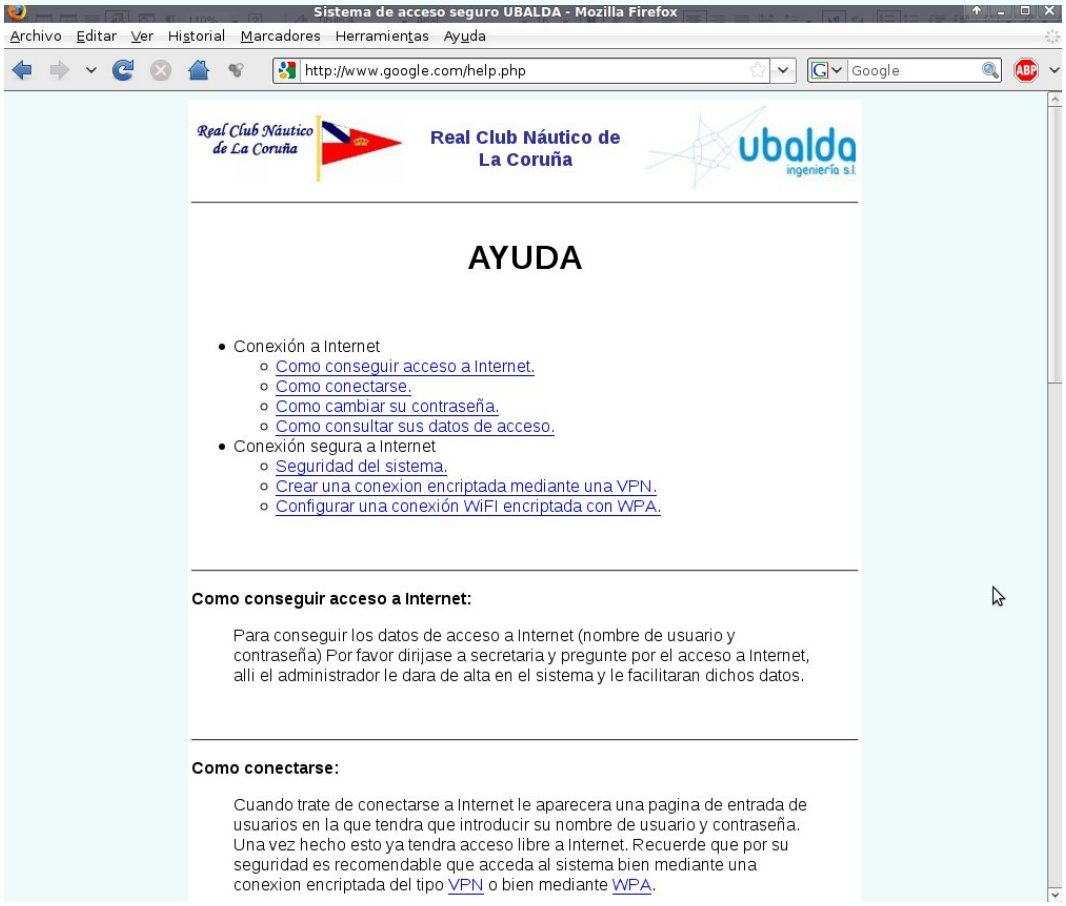
Gracias al uso del patrón MVC se pudo añadir la característica de que el administrador pueda modificar la página de bienvenida desde la propia página de administración, ya que la modificación de la presentación (la vista) no afecta al resto del sistema. Los componentes del modelo MVC implementados para las interfaces Web del gestor de ancho de banda son:

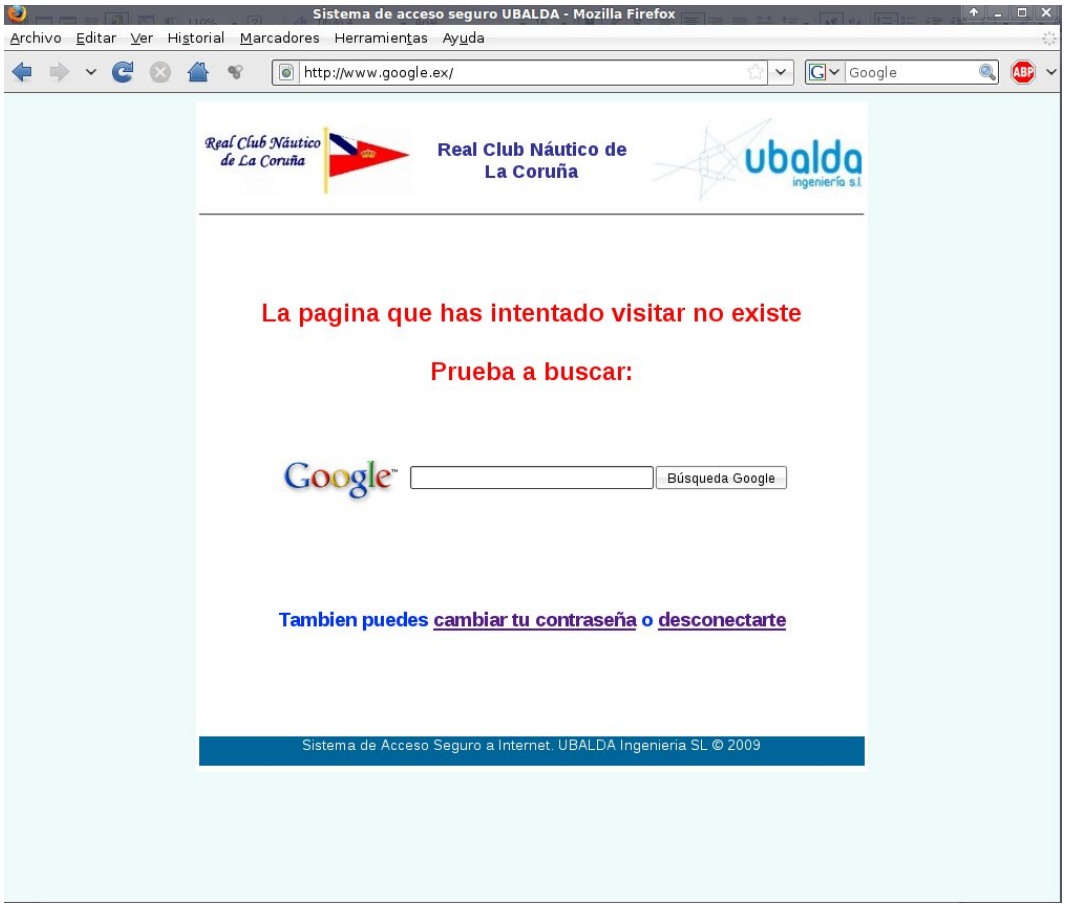
- **Modelo**: Son una serie de scripts en Perl y para bash que interactúan con el sistema, el servidor RADIUS y la base de datos LDAP y permiten la gestión a bajo nivel de todas las actividades del sistema.
- **Controlador**: Se ha programado utilizando PHP, se encarga de realizar la lógica de alto nivel a partir de los datos que suministra el modelo y entregarlos procesados para que puedan ser mostrados al usuario.
- **Vista**: Son una serie de plantillas en HTML, CSS y JavaScript que le dan el aspecto final a la Interfaz y pueden ser modificadas sin que ello influya en la funcionamiento de la Interfaz.

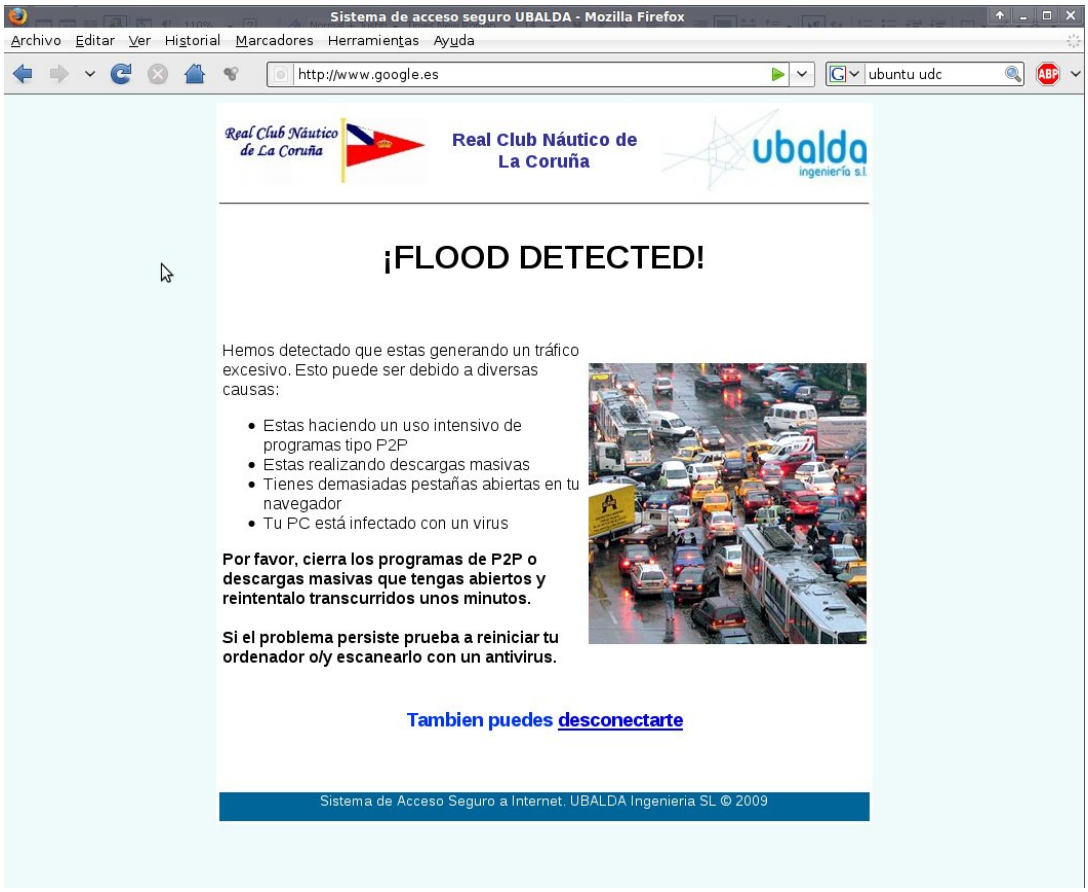
A continuación vamos a proceder a mostrar una serie de capturas de pantalla de dichas Interfaces.

4.3.4.3.2. Interfaz de la página de bienvenida

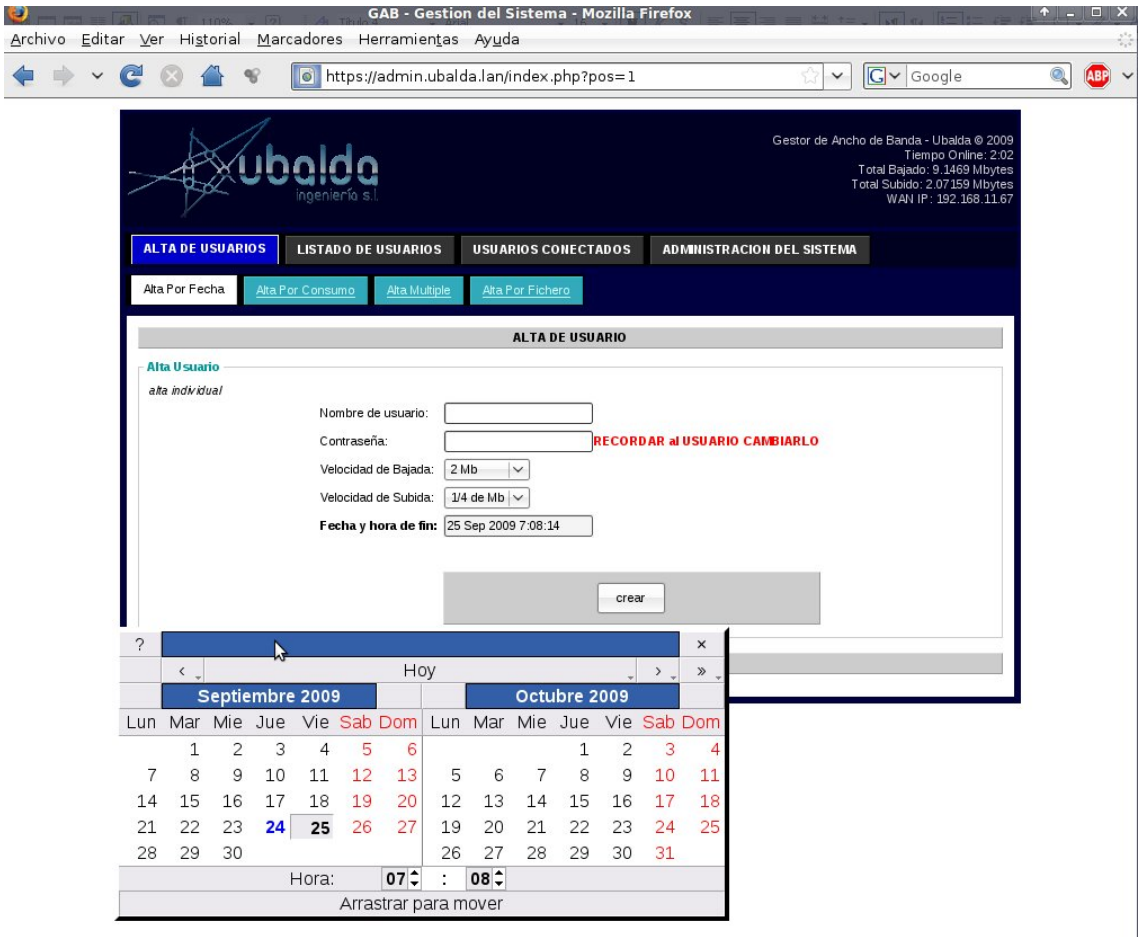

Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://login.ubalda.org/cgi-bin/login
Google
Real Club Náutico de La Coruña
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
Estado
Bienvenido, usuario eozores. Acabas de entrar en el sistema.
Google
Busqueda Google
Tambien puedes cambiar tu contraseña
Sistema de Acceso Seguro a Internet. UBALDA Ingenieria SL © 2009

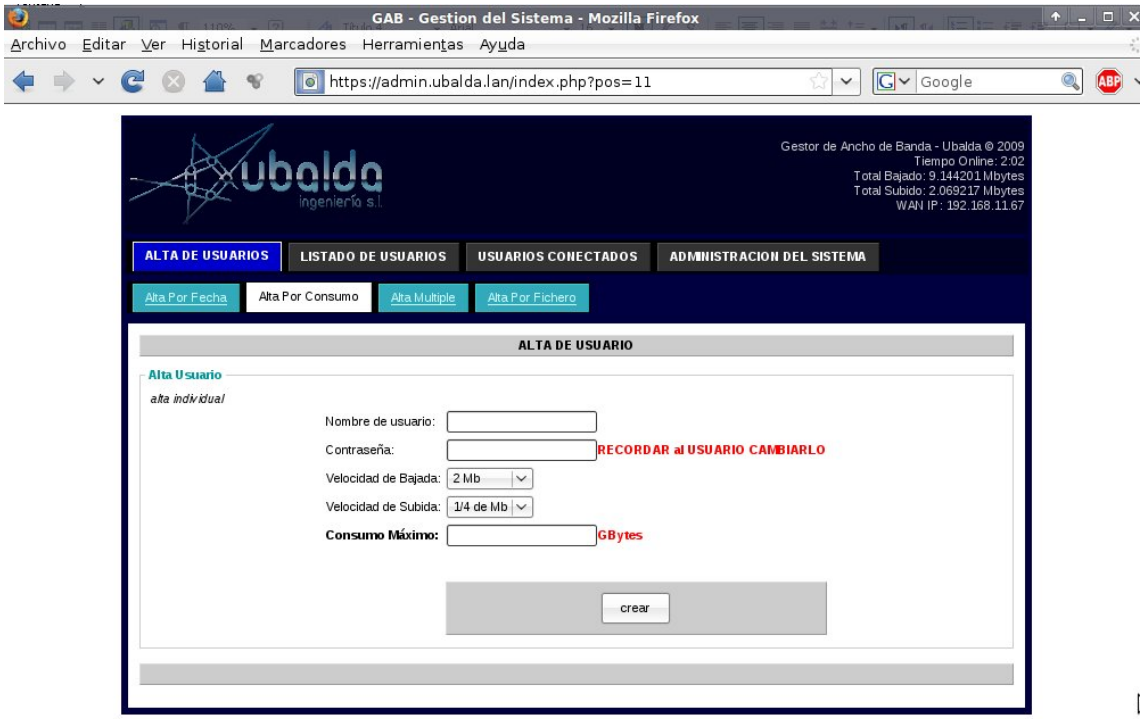

Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
http://www.google.com/help.php
Google
Real Club Náutico de La Coruña
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
AYUDA
Conexión a Internet
Como conseguir acceso a Internet.
Como conectarse.
Como cambiar su contraseña.
Como consultar sus datos de acceso.
Conexión segura a Internet
Seguridad del sistema.
Crear una conexion encriptada mediante una VPN.
Configurar una conexión WiFI encriptada con WPA.
Como conseguir acceso a Internet:
Para conseguir los datos de acceso a Internet (nombre de usuario y contraseña) Por favor dirijase a secretaria y pregunte por el acceso a Internet, alli el administrador le dara de alta en el sistema y le facilitaran dichos datos.
Como conectarse:
Cuando trate de conectarse a Internet le aparecera una pagina de entrada de usuarios en la que tendra que introducir su nombre de usuario y contraseña. Una vez hecho esto ya tendra acceso libre a Internet. Recuerde que por su seguridad es recomendable que acceda al sistema bien mediante una conexion encriptada del tipo VPN o bien mediante WPA.

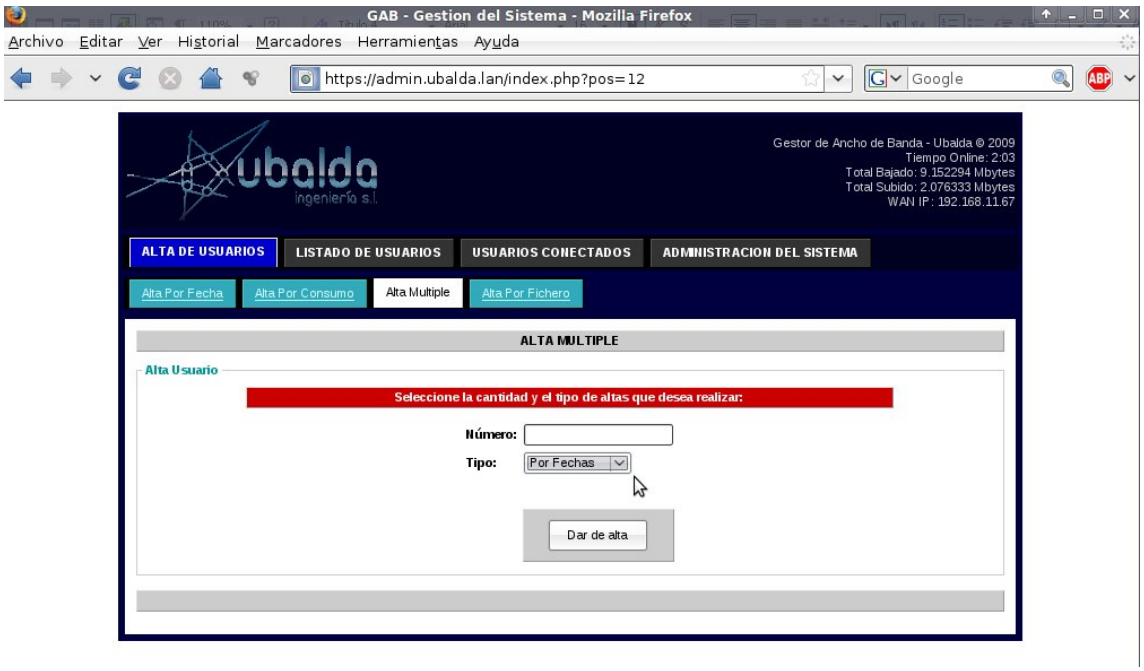

Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
http://www.google.ex/
Google
Real Club Náutico de La Coruña
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
La pagina que has intentado visitar no existe
Prueba a buscar:
Google
Búsqueda Google
Tambien puedes cambiar tu contraseña o desconectarte
Sistema de Acceso Seguro a Internet. UBALDA Ingenieria SL © 2009

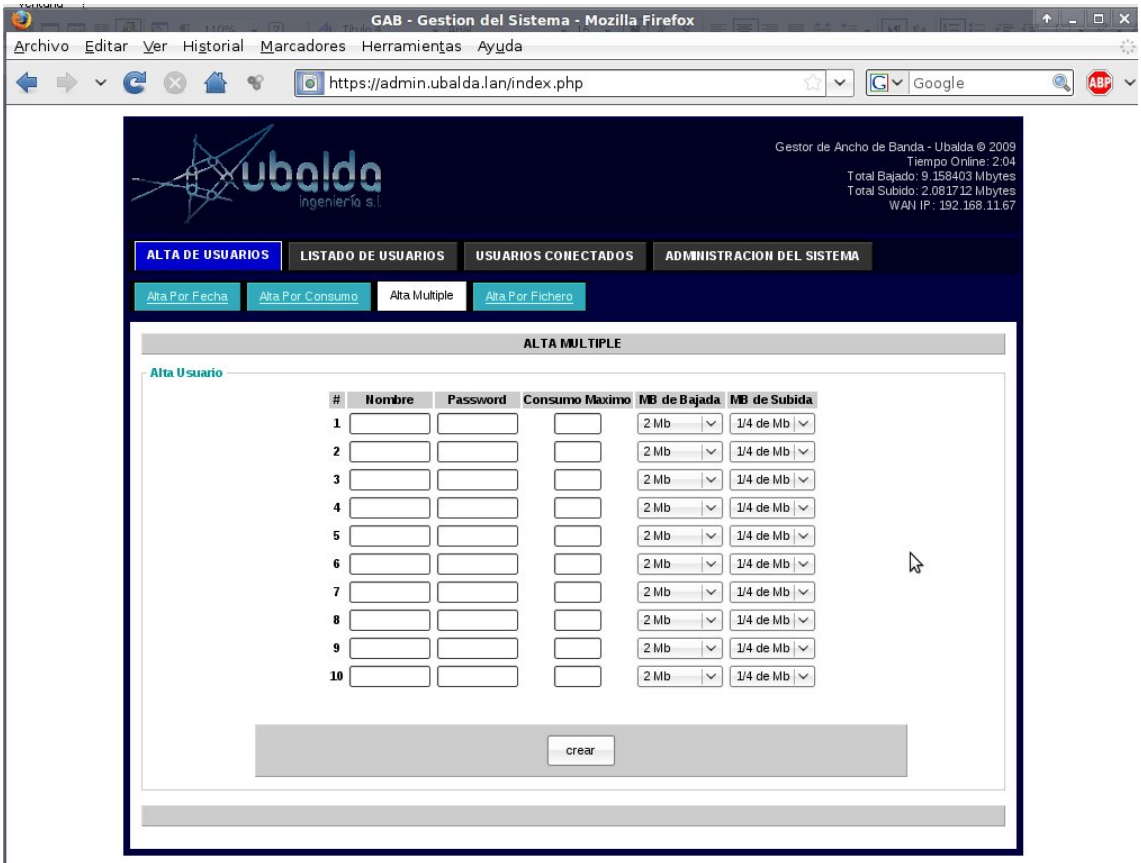

Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
http://www.google.es
ubuntu udc
Real Club Náutico de La Coruña
Real Club Náutico de La Coruña
ubalda ingeniería s.l.
¡FLOOD DETECTED!
Hemos detectado que estas generando un tráfico excesivo. Esto puede ser debido a diversas causas:
Estas haciendo un uso intensivo de programas tipo P2P
Estas realizando descargas masivas
Tienes demasiadas pestañas abiertas en tu navegador
Tu PC está infectado con un virus
Por favor, cierra los programas de P2P o descargas masivas que tengas abiertos y reintentalo transcurridos unos minutos.
Si el problema persiste prueba a reiniciar tu ordenador o/y escanearlo con un antivirus.
Tambien puedes desconectarte
Sistema de Acceso Seguro a Internet. UBALDA Ingenieria SL © 2009

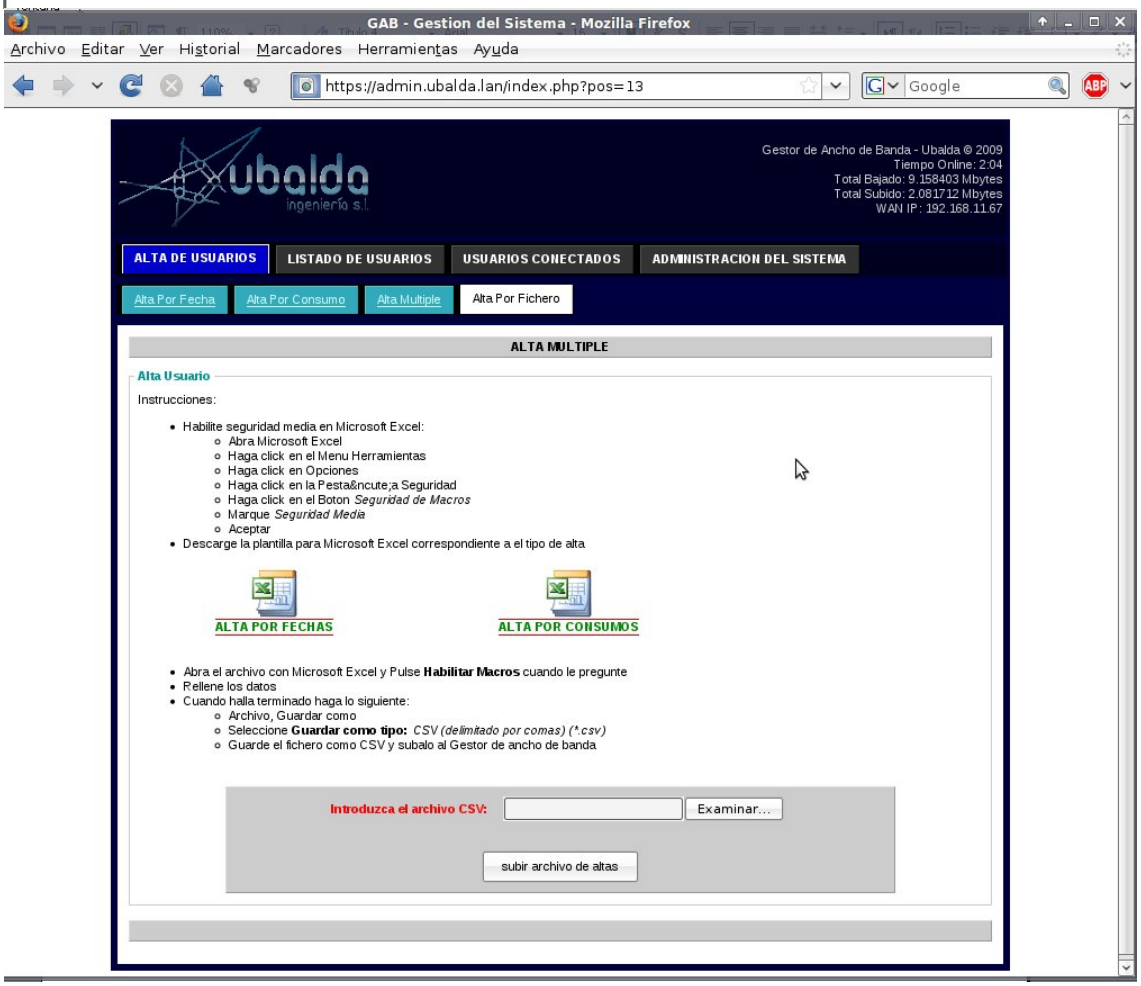
4.3.4.3.3. Interfaz de la página de administración de usuarios

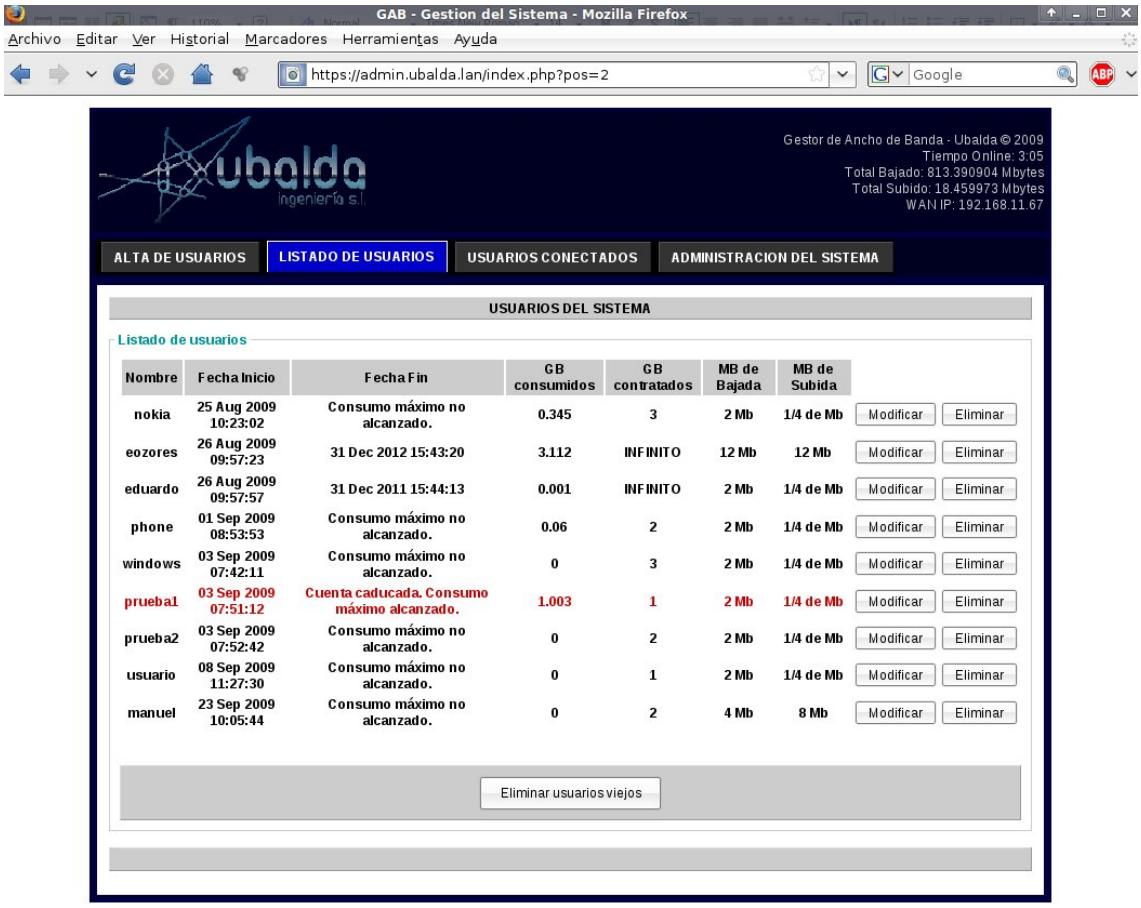


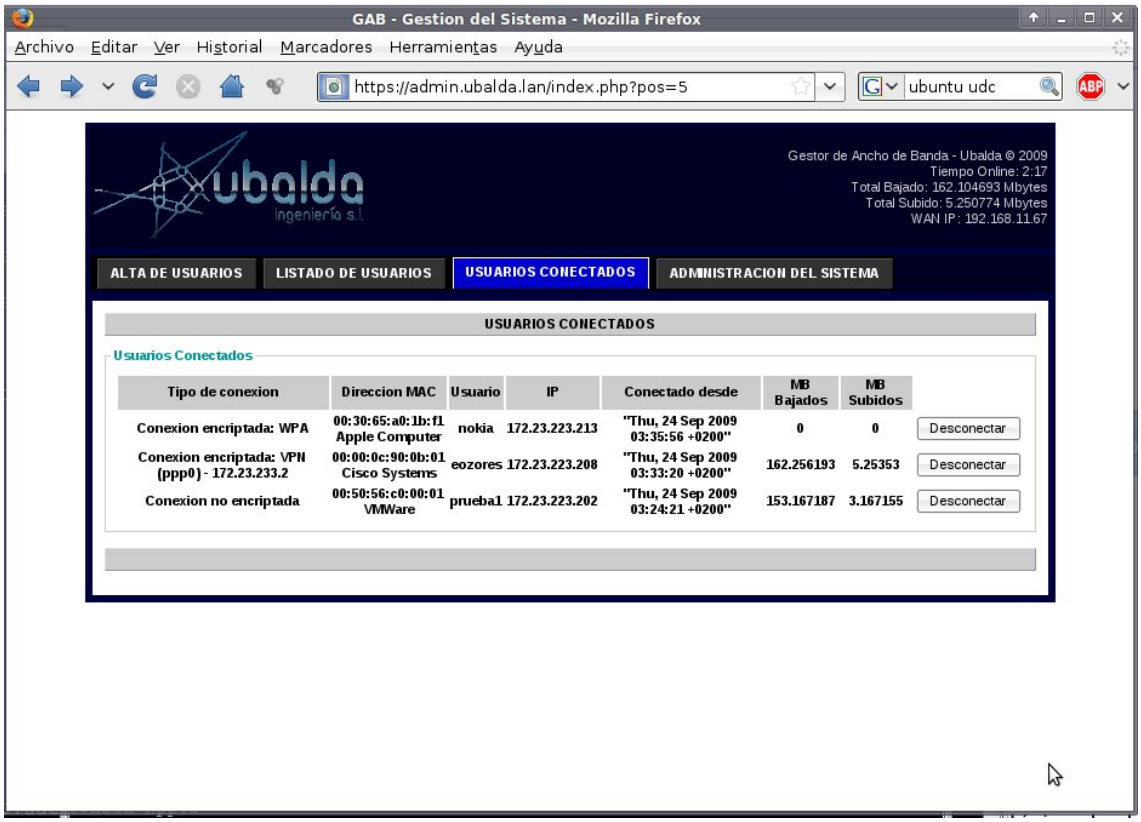

GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php?pos=11
Google
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:02
Total Bajado: 9.144201 Mbytes
Total Subido: 2.069217 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
Alta Por Fecha
Alta Por Consumo
Alta Multiple
Alta Por Fichero
ALTA DE USUARIO
Alta Usuario
alta individual
Nombre de usuario:
Contraseña:
RECORDAR al USUARIO CAMBIARLO
Velocidad de Bajada: 2 Mb
Velocidad de Subida: 1/4 de Mb
Consumo Máximo:
GBytes
crear


GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php?pos=12
Google
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:03
Total Bajado: 9.152294 Mbytes
Total Subido: 2.076333 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
Alta Por Fecha
Alta Por Consumo
Alta Multiple
Alta Por Fichero
ALTA MULTIPLE
Alta Usuario
Seleccione la cantidad y el tipo de altas que desea realizar:
Número:
Tipo:
Por Fechas
Dar de alta

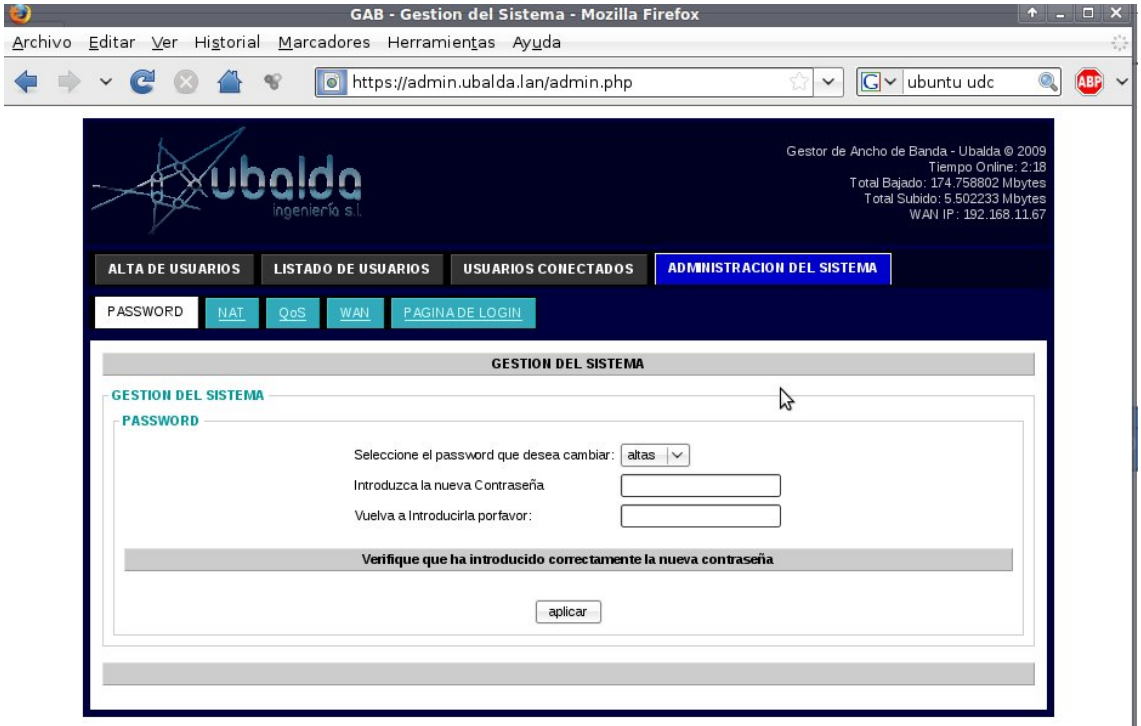

GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php
Google
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:04
Total Bajado: 9.158403 Mbytes
Total Subido: 2.081712 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
Alta Por Fecha
Alta Por Consumo
Alta Multiple
Alta Por Fichero
ALTA MULTIPLE
Alta Usuario
Nombre Password Consumo Maximo MB de Bajada MB de Subida
1 2 Mb 1/4 de Mb
2 2 Mb 1/4 de Mb
3 2 Mb 1/4 de Mb
4 2 Mb 1/4 de Mb
5 2 Mb 1/4 de Mb
6 2 Mb 1/4 de Mb
7 2 Mb 1/4 de Mb
8 2 Mb 1/4 de Mb
9 2 Mb 1/4 de Mb
10 2 Mb 1/4 de Mb
crear

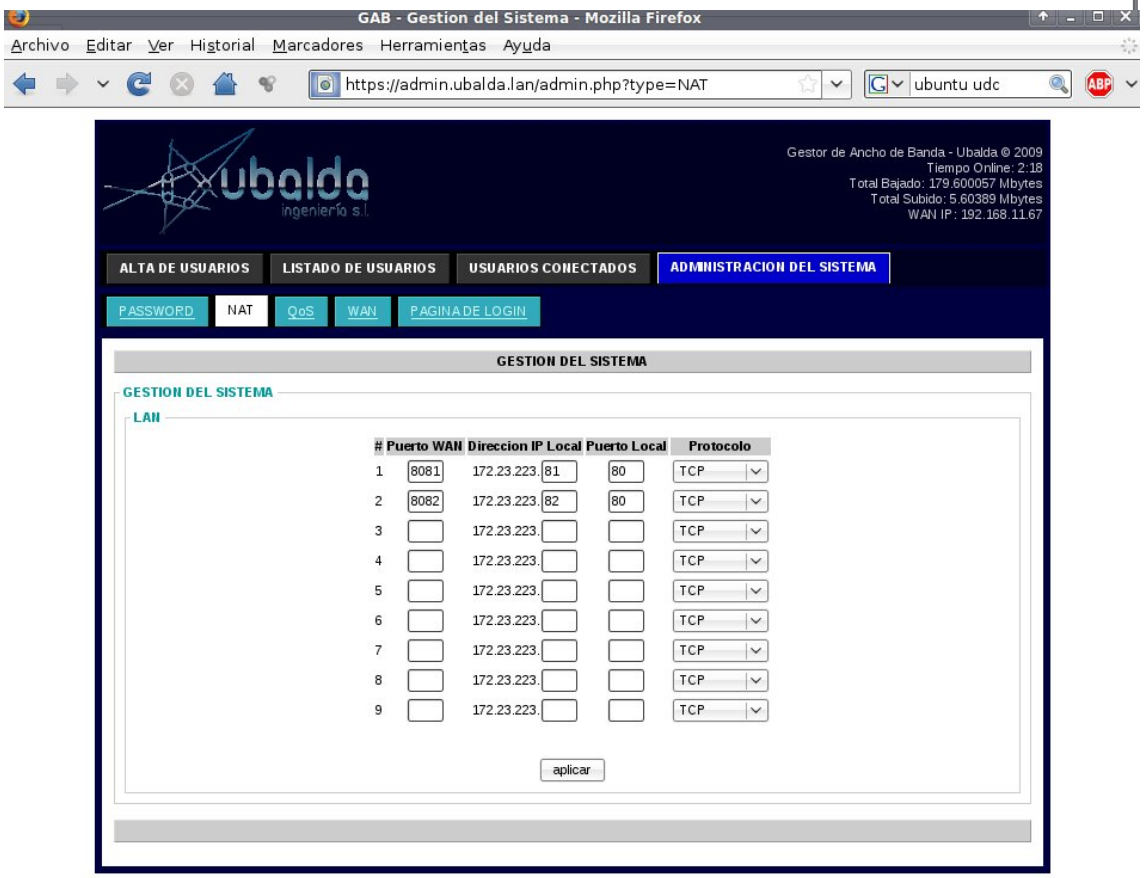

GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php?pos=13
Google
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:04
Total Bajado: 9.158403 Mbytes
Total Subido: 2.081712 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
Alta Por Fecha
Alta Por Consumo
Alta Multiple
Alta Por Fichero
ALTA MULTIPLE
Alta Usuario
Instrucciones:
Habilite seguridad media en Microsoft Excel:
Abra Microsoft Excel
Haga click en el Menu Herramientas
Haga click en Opciones
Haga click en la Pesta&ncute;a Seguridad
Haga click en el Boton Seguridad de Macros
Marque Seguridad Media
Aceptar
Descarge la plantilla para Microsoft Excel correspondiente a el tipo de alta
ALTA POR FECHAS
ALTA POR CONSUMOS
Abra el archivo con Microsoft Excel y Pulse Habilitar Macros cuando le pregunte
Rellene los datos
Cuando halla terminado haga lo siguiente:
Archivo, Guardar como
Seleccione Guardar como tipo: CSV (delimitado por comas) (*.csv)
Guarde el fichero como CSV y subalo al Gestor de ancho de banda
Introduzca el archivo CSV:
Examinar...
subir archivo de altas

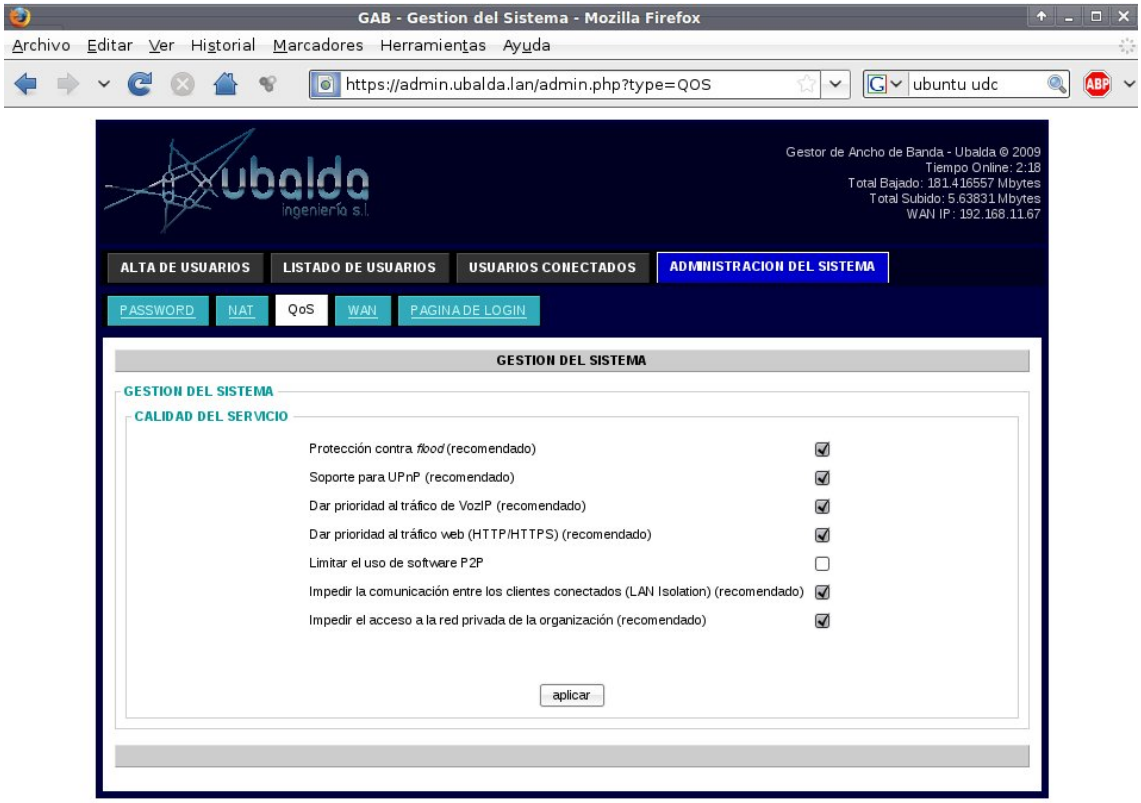

GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php?pos=2
Google
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 3:05
Total Bajado: 813.390904 Mbytes
Total Subido: 18.459973 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
USUARIOS DEL SISTEMA
Listado de usuarios
Nombre | Fecha Inicio | Fecha Fin | GB consumidos | GB contratados | MB de Bajada | MB de Subida
nokia | 25 Aug 2009 10:23:02 | Consumo máximo no alcanzado. | 0.345 | 3 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
eozores | 26 Aug 2009 09:57:23 | 31 Dec 2012 15:43:20 | 3.112 | INFINITO | 12 Mb | 12 Mb | Modificar | Eliminar
eduardo | 26 Aug 2009 09:57:57 | 31 Dec 2011 15:44:13 | 0.001 | INFINITO | 2 Mb | 1/4 de Mb | Modificar | Eliminar
phone | 01 Sep 2009 08:53:53 | Consumo máximo no alcanzado. | 0.06 | 2 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
windows | 03 Sep 2009 07:42:11 | Consumo máximo no alcanzado. | 0 | 3 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
prueba1 | 03 Sep 2009 07:51:12 | Cuenta caducada. Consumo máximo alcanzado. | 1.003 | 1 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
prueba2 | 03 Sep 2009 07:52:42 | Consumo máximo no alcanzado. | 0 | 2 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
usuario | 08 Sep 2009 11:27:30 | Consumo máximo no alcanzado. | 0 | 1 | 2 Mb | 1/4 de Mb | Modificar | Eliminar
manuel | 23 Sep 2009 10:05:44 | Consumo máximo no alcanzado. | 0 | 2 | 4 Mb | 8 Mb | Modificar | Eliminar
Eliminar usuarios viejos

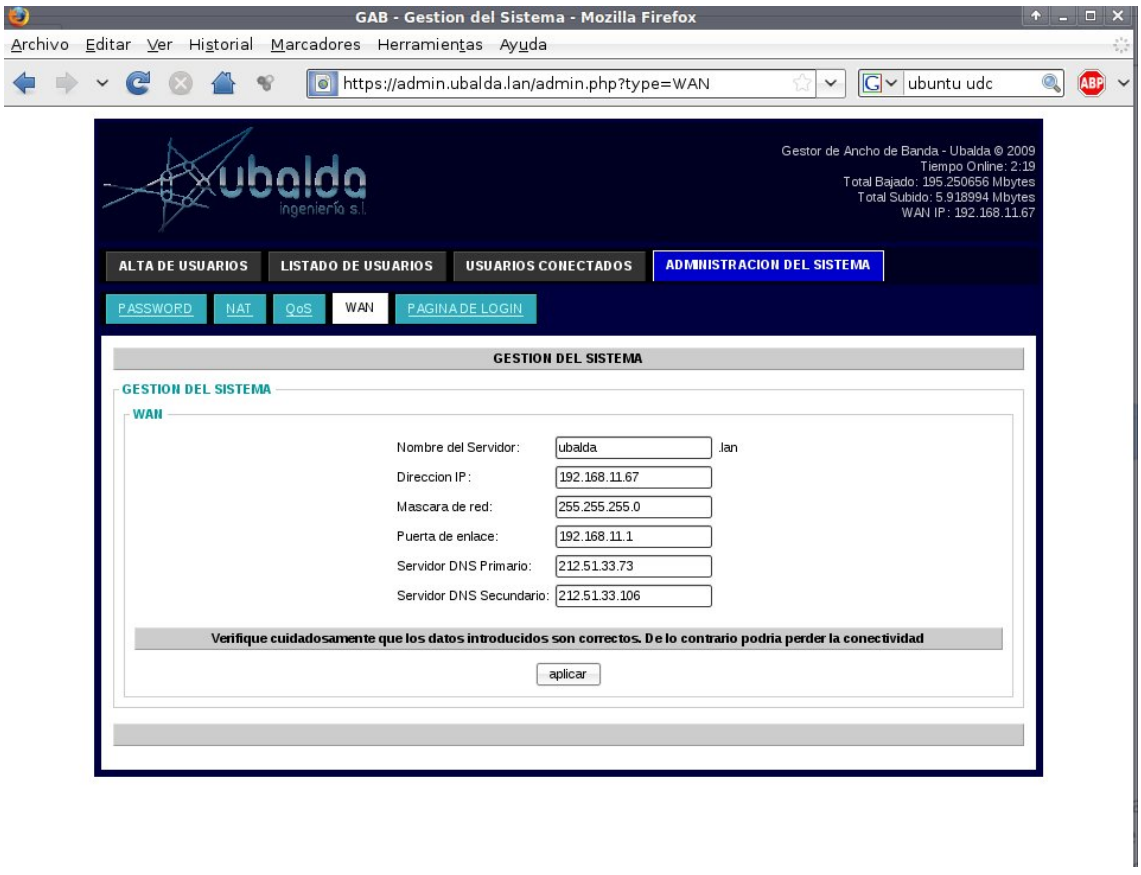

GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/index.php?pos=5
ubuntu udc
ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:17
Total Bajado: 162.104693 Mbytes
Total Subido: 5.250774 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
USUARIOS CONECTADOS
Usuarios Conectados
Tipo de conexion | Direccion MAC | Usuario | IP | Conectado desde | MB Bajados | MB Subidos
Conexion encriptada: WPA | 00:30:65:a0:1b:f1 Apple Computer | nokia | 172.23.223.213 | "Thu, 24 Sep 2009 03:35:56 +0200" | 0 | 0 | Desconectar
Conexion encriptada: VPN (ppp0) - 172.23.233.2 | 00:00:0c:90:0b:01 Cisco Systems | eozores | 172.23.223.208 | "Thu, 24 Sep 2009 03:33:20 +0200" | 162.256193 | 5.25353 | Desconectar
Conexion no encriptada | 00:50:56:c0:00:01 VMWare | prueba1 | 172.23.223.202 | "Thu, 24 Sep 2009 03:24:21 +0200" | 153.167187 | 3.167155 | Desconectar

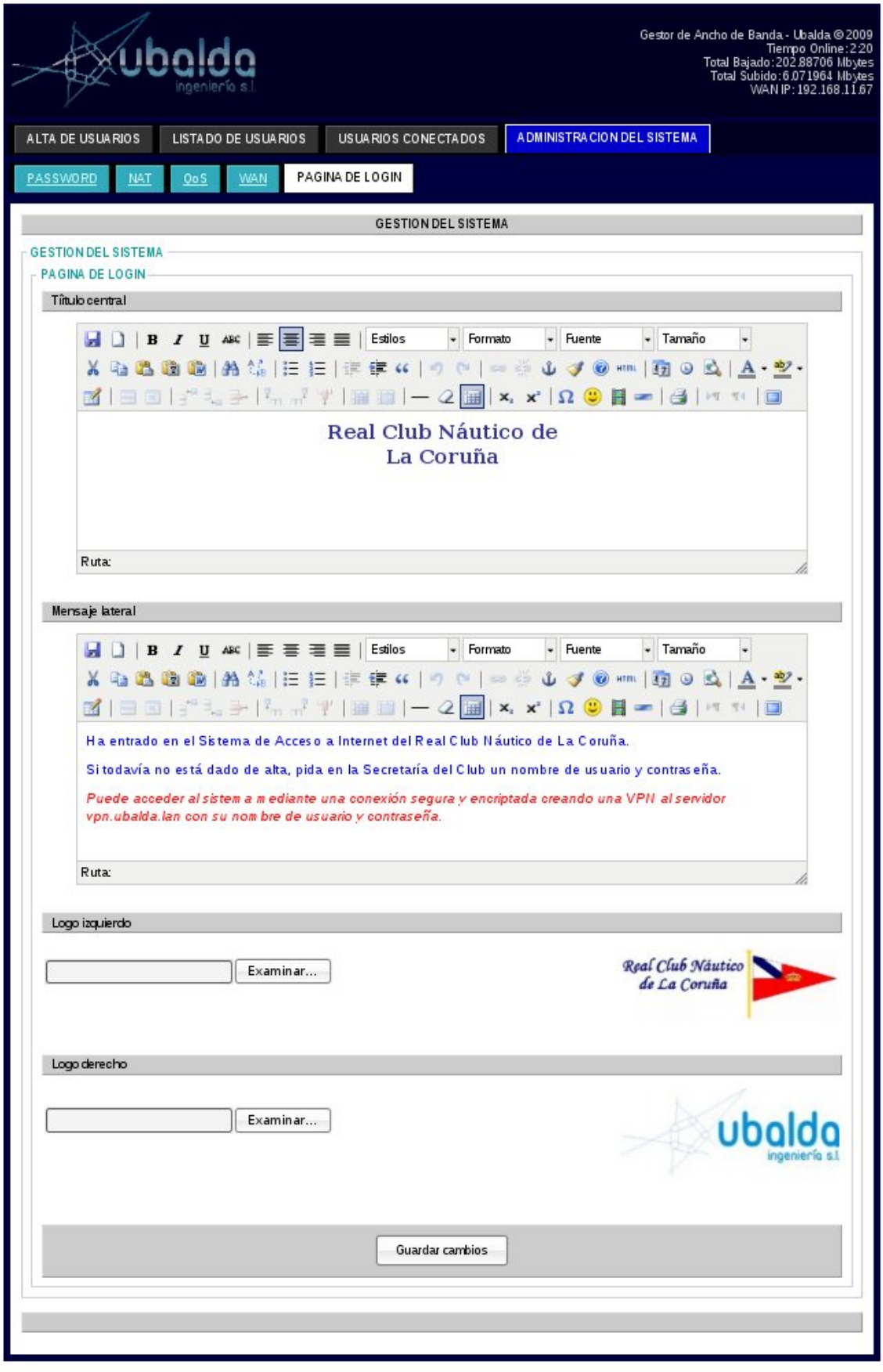
4.3.4.3.4. Interfaz de la página de administración del sistema






GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/admin.php?type=QOS
ubuntu udc
ubalda ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:18
Total Bajado: 181.416557 Mbytes
Total Subido: 5.63831 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
PASSWORD
NAT
QoS
WAN
PAGINA DE LOGIN
GESTION DEL SISTEMA
GESTION DEL SISTEMA
CALIDAD DEL SERVICIO
Protección contra flood (recomendado)
Soporte para UPnP (recomendado)
Dar prioridad al tráfico de VozIP (recomendado)
Dar prioridad al tráfico web (HTTP/HTTPS) (recomendado)
Limitar el uso de software P2P
Impedir la comunicación entre los clientes conectados (LAN Isolation) (recomendado)
Impedir el acceso a la red privada de la organización (recomendado)
aplicar


GAB - Gestion del Sistema - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://admin.ubalda.lan/admin.php?type=WAN
ubuntu udc
ubalda ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:19
Total Bajado: 195.250656 Mbytes
Total Subido: 5.918994 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
PASSWORD
NAT
QoS
WAN
PAGINA DE LOGIN
GESTION DEL SISTEMA
GESTION DEL SISTEMA
WAN
Nombre del Servidor: ubalda .lan
Direccion IP: 192.168.11.67
Mascara de red: 255.255.255.0
Puerta de enlace: 192.168.11.1
Servidor DNS Primario: 212.51.33.73
Servidor DNS Secundario: 212.51.33.106
Verifique cuidadosamente que los datos introducidos son correctos. De lo contrario podria perder la conectividad
aplicar


ubalda
ingeniería s.l.
Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:20
Total Bajado: 202.88706 Mbytes
Total Subido: 6.071964 Mbytes
WAN IP: 192.168.11.67
ALTA DE USUARIOS
LISTADO DE USUARIOS
USUARIOS CONECTADOS
ADMINISTRACION DEL SISTEMA
PASSWORD
NAT
QoS
WAN
PAGINA DE LOGIN
GESTION DEL SISTEMA
GESTION DEL SISTEMA
PAGINA DE LOGIN
Título central
Estilos
Formato
Fuente
Tamaño
Real Club Náutico de
La Coruña
Ruta:
Mensaje lateral
Estilos
Formato
Fuente
Tamaño
Ha entrado en el Sistema de Acceso a Internet del Real Club Náutico de La Coruña.
Si todavía no está dado de alta, pida en la Secretaría del Club un nombre de usuario y contraseña.
Puede acceder al sistema mediante una conexión segura y encriptada creando una VPN al servidor vpn.ubalda.lan con su nombre de usuario y contraseña.
Ruta:
Logo izquierdo
Examinar...
Real Club Náutico de La Coruña
Logo derecho
Examinar...
ubalda
ingeniería s.l.
Guardar cambios

4.3.5. Pruebas

Debido al cambio de componentes internos que se tuvo que realizar para soportar las características nuevas fue necesario comprobar nuevamente la estabilidad del sistema resultando todas las pruebas satisfactorias.

El soporte para tarjetas inalámbricas integradas en el propio gestor (tarjetas PCI WiFi con chip Atheros) ocasiono una serie de problemas. En concreto con la tarjeta inalámbrica integrada en el Asus eee que se uso en el presente proyecto. El driver que se utilizó en un primer momento fue MadWifi[88] que no viene incluido en el kernel oficial de Linux y tuvo que ser compilado aparte.

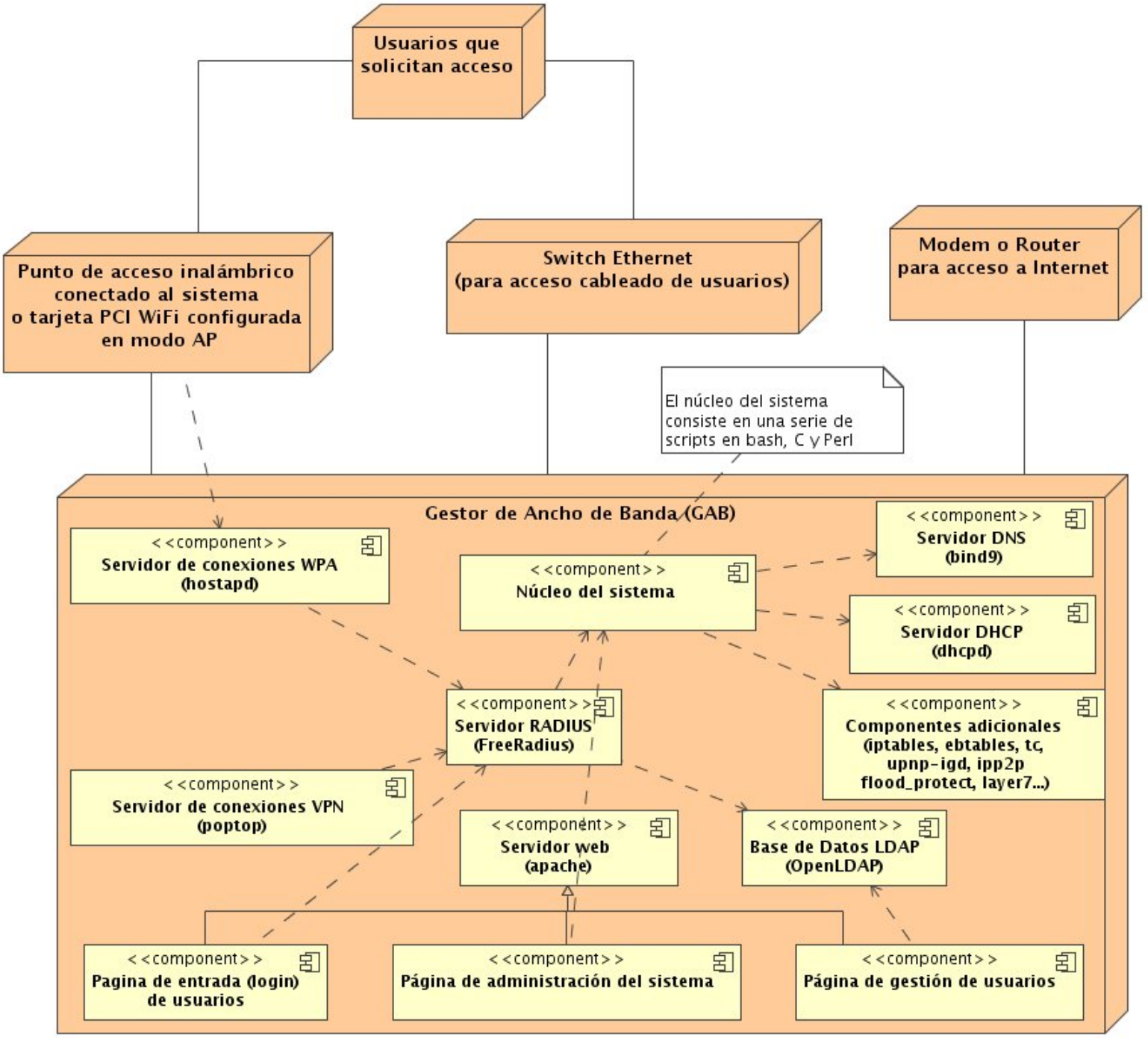
MadWifi es uno de los más avanzados drivers WLAN disponibles hoy en día para Linux. Es estable y ha sido probado por miles de usuarios, el driver es opensource sin embargo depende de código binario propietario para la capa de abstracción de hardware HAL (Hardware Abstraction Layer). Esta característica implica que los desarrolladores del driver no pueden modificar la capa HAL del mismo ya que solo disponen del código binario, lo cual impone limitaciones severas a su desarrollo. El driver funciona correctamente en modo "cliente" pero al poner la tarjeta en modo AP aparece un *bug* documentado conocido como "Stuck Beacon"[89] que hace que la tarjeta se resetee aleatoriamente desconectando a los usuarios. Dicho *bug* no tiene solución ya que la parte de bajo nivel del controlador (HAL) está en formato binario y no se puede modificar. La solución fue utilizar el driver Ath5k donde el código HAL fue escrito por la comunidad opensource desde cero aplicando ingeniería inversa sobre el HAL original de MadWifi en formato binario. Hasta hace poco el driver Ath5k no funcionaba en modo AP, pero recientemente se ha incorporado esa característica y las pruebas realizadas con el han sido satisfactorias por el momento.

Se probaron todas las características nuevas añadidas al sistema con resultados satisfactorios

[88] http://madwifi-project.org/
[89] http://madwifi-project.org/wiki/StuckBeacon

4.4. Visión global del sistema

En el siguiente diagrama podemos apreciar todos los componentes software que componen el sistema y como se interrelacionan entre si.



Capitulo 5. Planificación y evaluación de costes

5.1. Planificación

La jornada laboral considerada es de 5 horas, con un horario de lunes a viernes de 9:00 a 14:00. Los días de descanso serán los sábados, domingos y festivos.

Como se ha comentado, la metodología a seguir se basa en la definida por el Proceso Unificado de Desarrollo de Software. Esta filosofía de trabajo propone el desarrollo por mini-proyectos dentro de un esquema iterativo e incremental, de forma que cada iteración incorpora más funcionalidad sobre la anterior, hasta que en la última se termina con un software que implementa toda la funcionalidad.

A continuación se enumeran las iteraciones en las que se ha dividido el desarrollo de este proyecto:

- **ITERACIÓN 0**: "Análisis y Formación". Comprende el estudio del sistema CANSAS en el que se basa esta aplicación, del análisis de esta aplicación surgirán una serie de requisitos para el inicio de la Iteración primera. Comprende las primeras actividades de análisis general de los requisitos de la aplicación y la formación. A la finalización de esta iteración se tendrá una idea general de la aplicación y los conocimientos técnicos necesarios para la realización de la mayor parte de sus funcionalidades. Comprende las subactividades de:

  - Análisis Previo: Estudio de CANSAS, comprensión de su funcionamiento y análisis de sus características que serán los requisitos de la primera iteración.

  - Formación: Formación en las principales tecnologías a utilizar, desconocidas hasta el momento: RADIUS, LDAP y control avanzando del tráfico en Linux (iptables, tc)

- **ITERACIÓN 1**: Implementación de un sistema con las mismas características que CANSAS partiendo de cero y utilizando las últimas versiones de los componentes software que componen el sistema.
  - Análisis de requisitos: se tomaron como requisitos las características y funcionalidades obtenidas en el estudio de CANSAS.

  - Diseño e Implementación: Se procedió a instalar Ubuntu 8.04 server y a compilar e instalar todo el software necesario comprobando que funcionaba todo correctamente. Se diseño una Interfaz Web para la administración del sistema.

  - Pruebas: Diversas pruebas de funcionamiento y estabilidad del sistema

- **ITERACIÓN 2**: Estabilización del sistema, corrección de errores y *"bugs"* detectados en las pruebas de la etapa anterior. Se añade la posibilidad de imponer límites al consumo de ancho de banda de los usuarios en vez de la fecha límite de acceso. Se añaden la posibilidad de cambiar la configuración TCP/IP del sistema desde la Interfaz Web de administración.

  - Análisis de requisitos: estabilización del sistema, corrección de errores y *"bugs"* y añadir las funcionalidades descritas.

  - Diseño e Implementación: Un análisis profundo del funcionamiento del sistema revelo la causa de los errores y los problemas de estabilidad que fueron corregidos. Se modifica la base de datos LDAP y se crea un módulo para el servidor RADIUS que permite tener usuarios con un límite de acceso impuesto por su consumo acumulado de ancho de banda. Se programan los Scripts en bash y Perl que permiten implementarlo.

  - Pruebas: Diversas pruebas de funcionamiento y estabilidad del sistema entre las que destaca la implantación del sistema en el Real Club Náutico de Coruña de forma exitosa.

- **ITERACIÓN 3**: Esta etapa se caracteriza por una gran especificación de requisitos nuevos fruto de las necesidades de la empresa Ubalda y el *"feedback"* aportado por los usuarios del sistema instalado en Real Club Náutico de Coruña. Se añaden características avanzadas de control de tráfico y de calidad del servicio (QoS) y se implementa el soporte nativo para redes inalámbricas encriptadas WPA-PEAP. Se añade también la posibilidad de pagar de forma on-line a través de PayPal o utilizando un sistema monedero con una impresora de tickets. Se decide que se rediseñará completamente la Interfaz de Administración Web del sistema y la Interfaz de bienvenida de usuarios siguiendo el patrón MVC. Se añade una nueva figura de administrador para gestionar los parámetros del sistema.

  - Análisis de requisitos: los requisitos provienen de las necesidades de la empresa y las sugerencias de los usuarios del sistema.

  - Diseño e Implementación: Debido a la gran especificación de requisitos nuevos se hizo necesario compilar parte del software que compone el sistema (iptables y el kernel de linux) así como la instalación de nuevo software que permite implementar dichas características (hostapd, ebtables). Se programaron las interfaces Web de administración y de usuarios en base a los nuevos requisitos.

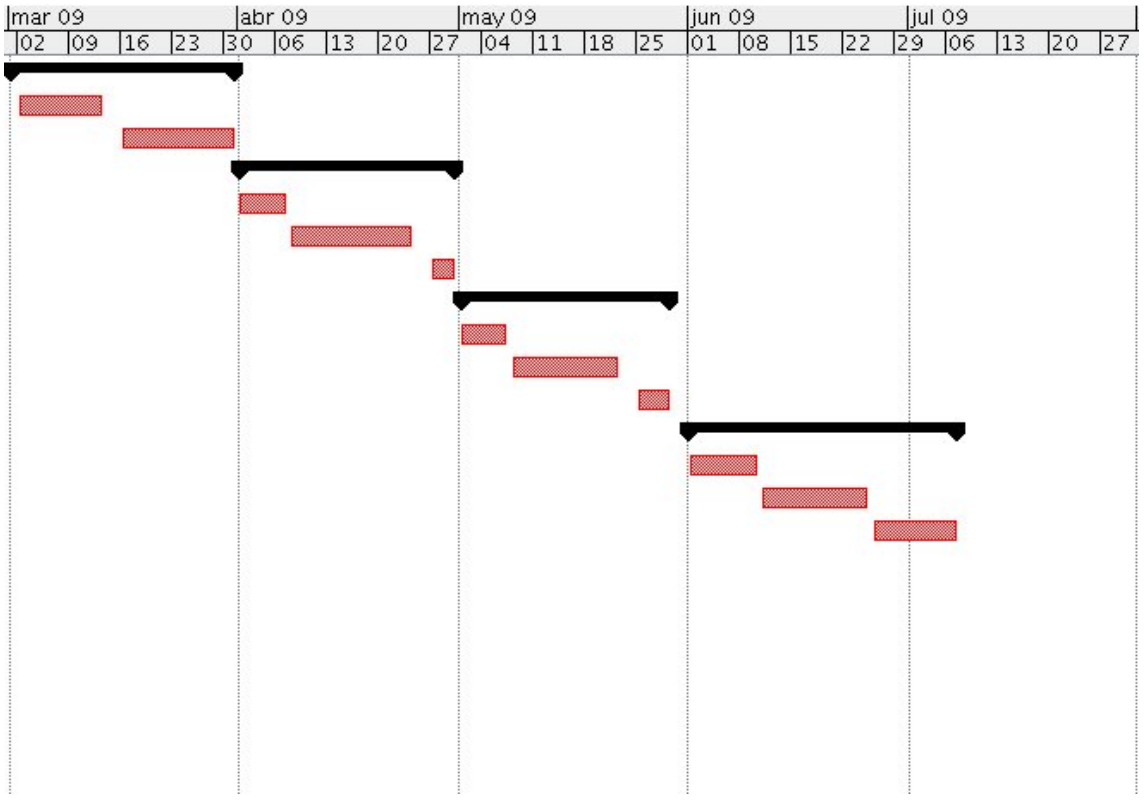
  - Pruebas: Diversas pruebas de estabilidad del sistema debido al cambio de componentes que resultaron satisfactorias a excepción del caso descrito en el apartado anterior con la tarjeta inalámbrica WiFi integrada que ha sido solventado pero necesita de pruebas más intensivas para garantizar su completa estabilidad. Se probo el funcionamiento de las nuevas características de forma correcta

A continuación se muestran una serie de diagramas que ilustran el proceso, se incluye el correspondiente diagrama de Gantt[90]

[90] http://es.wikipedia.org/wiki/Diagrama_de_Gantt

	Nombre	Trabajo	Entorno de Trabajo
1	Trabajo	348 horas	
	Análisis de requisitos	*24 horas*	*Plano*
	Diseño e Implementación	*65 horas*	*Plano*
	Pruebas	*20 horas*	*Plano*
	Análisis de requisitios	*24 horas*	*Plano*
	Diseño e Implementación	*55 horas*	*Plano*
	Pruebas	*25 horas*	*Plano*
	Análisis de requisitos	*40 horas*	*Plano*
	Diseño e Implementación	*55 horas*	*Plano*
	Pruebas	*40 horas*	*Plano*

		Nombre	Duración	Inicio	Terminado
1		**Iteración 0**	22 days?	**1/03/09 9:00**	**31/03/09 14:00**
2		Análisis previo	10 days?	1/03/09 9:00	13/03/09 14:00
3		Formación	12 days?	14/03/09 9:00	31/03/09 14:00
4		**Iteración 1**	22 days?	**1/04/09 8:00**	**30/04/09 14:00**
5		Análisis de requisitos	4,8 days?	1/04/09 8:00	7/04/09 13:00
6		Diseño e Implementación	13 days?	8/04/09 8:00	24/04/09 14:00
7		Pruebas	4 days?	25/04/09 8:00	30/04/09 14:00
8		**Iteración 2**	21 days?	**1/05/09 8:00**	**29/05/09 14:00**
9		Análisis de requisitios	4,8 days?	1/05/09 8:00	7/05/09 13:00
10		Diseño e Implementación	11 days?	8/05/09 8:00	22/05/09 14:00
11		Pruebas	5 days?	24/05/09 8:00	29/05/09 14:00
12		**Iteración 3**	27 days?	**1/06/09 8:00**	**7/07/09 14:00**
13		Análisis de requisitos	8 days?	1/06/09 8:00	10/06/09 14:00
14		Diseño e Implementación	11 days?	11/06/09 8:00	25/06/09 14:00
15		Pruebas	8 days?	26/06/09 8:00	7/07/09 14:00



5.2. Estimación de costes

Todo el software empleado en el desarrollo del sistema es Software Libre de código abierto (Open Source[91]), por lo que todo el coste del proyecto se divide entre horas de trabajo y hardware. No obstante existirá un coste de software derivado de los certificados SSL necesarios para el correcto funcionamiento del sistema.

El desarrollo del proyecto se hace entre los meses de Marzo y Julio de 2009 bajo una beca FEUGA. A continuación se detallan los costes estimados en Hardware y horas de trabajo.

💻 Hardware = 560

Descripción	Nº Unidades	Cote Unitario	Total
IBM Pentium II	1	80	80
PC Pentium IV	1	150	150
Asus eeepc	1	150	150
Tarjeta Ethernet	2	5	10
Tarjeta WiFi	2	10	20

[91] http://es.wikipedia.org/wiki/Código_abierto

Punto de acceso	3	50	150

- Tiempo = 2100

Descripción	Nº Unidades	Coste Unitario	Total
Hora de trabajo	350	6	2100

- Software = 250

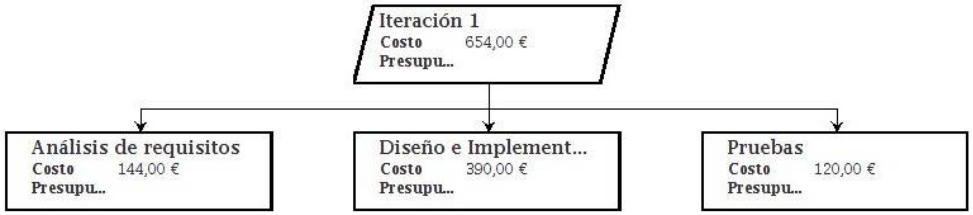
Descripción	Nº Unidades	Coste Unitario	Total
Certificado SSL pagina de *login*	1	10	10
Certificado SSL acceso WPA[92]	1	240	240

- Coste total = 2100 + 560 + 250 = 2910

El mantenimiento de la aplicación así como su implantación definitiva en el mundo real se prevé que será realizado por el desarrollador de la misma por lo que no supondrá un gran coste añadido.

[92] http://www.verisign.com/ssl/buy-ssl-certificates/specialized-ssl-certificates/wireless-lan-security/


Iteración 0
Costo 0,00 €
Presupu...
Análisis previo
Costo 0,00 €
Presupu...
Formación
Costo 0,00 €
Presupu...


Iteración 1
Costo 654,00 €
Presupu...
Análisis de requisitos
Costo 144,00 €
Presupu...
Diseño e Implement...
Costo 390,00 €
Presupu...
Pruebas
Costo 120,00 €
Presupu...


Iteración 2
Costo 624,00 €
Presupu...
Análisis de requisitios
Costo 144,00 €
Presupu...
Diseño e Implement...
Costo 330,00 €
Presupu...
Pruebas
Costo 150,00 €
Presupu...


Iteración 3
Costo 810,00 €
Presupu...
Análisis de requisitos
Costo 240,00 €
Presupu...
Diseño e Implement...
Costo 330,00 €
Presupu...
Pruebas
Costo 240,00 €
Presupu...

Capitulo 6. Conclusiones y futuras líneas de trabajo.

En este apartado se realizarán las conclusiones oportunas y se analizarán los objetivos alcanzados en este proyecto. Además se comentarán futuros desarrollos y líneas de trabajo para mejorar el producto obtenido.

6.1. Conclusiones

En el presente proyecto se ha conseguido implementar con éxito una solución que ofrece características que solo están presentes en los sistemas de altas prestaciones. Además integra características únicas como las de poder definir políticas de ancho de banda independientes para cada uno de los usuarios. El desarrollo es totalmente flexible y se puede adaptar o modificar de forma sencilla una vez que se ha comprendido como funciona todo el conjunto, por ejemplo puede adaptarse el sistema para que use un servidor RADIUS externo para autenticar a los usuarios o incluso para que trabaje contra una base de datos de *Microsoft Windows Active Directory*. Este tipo de bases de datos son las que emplea Windows Server para almacenar los datos de los usuarios y consiste en una base de datos LDAP muy similar a la empleada en este proyecto por lo que si una empresa lo desease el gestor de ancho de banda se integraría con su base de datos de forma que se importaría automáticamente todos los perfiles de los usuarios.

La capacidad de Linux para manejar las interfaces de red y el tráfico TCP/IP es realmente impresionante, con los conocimientos necesarios se pueden hacer cosas que ni siquiera son imaginables en otro tipo de sistemas propietarios como los basados en Windows. Esto es posible, en su mayor parte, gracias a la filosofía *"opensource"*, ya que al estar disponible para el programador todo el código fuente del sistema este puede modificarlo para que haga justo lo que el desee, cosa que no es posible en los sistemas propietarios donde el software que maneja el tráfico TCP/IP depende de un API propietaria e inflexible.

Lo que mas valoro de esta experiencia es el amplio conocimiento que he adquirido en todo lo relativo a gestión de redes, protocolos de red, controles de accesos y seguridad de sistemas. También me ha servido para profundizar y ampliar mis conocimientos del sistema operativo Linux.

Un software especialmente interesante es FreeRADIUS, que permite utilizarlo como plataforma de autenticación segura para infinidad de aplicaciones, por ejemplo la autenticación de usuarios en el acceso a una WPA Empresarial o en el acceso a su cuenta de usuario del sistema operativo, ya sea Windows o Linux. Además el diseño modular de FreeRADIUS permite implementar nuevas características y modelos de autenticación avanzados de forma relativamente sencilla.

6.2. Futuras líneas de trabajo

La siguiente línea de trabajo en el presente desarrollo sería la de establecer una cuarta iteración en el proceso de desarrollo que comenzaría con una gran especificación de nuevos requisitos obtenidos en parte de las necesidades de la empresa y en parte del *"feedback"* de los propios usuarios, es decir, de las opiniones, sugerencias y críticas que puedan aportar los usuarios del sistema durante su uso.

Algunas de las nuevas especificaciones en las que se ha pensado son las siguientes:

- Desarrollar una versión del sistema optimizada para dispositivos empotrados (router inalámbricos) basada en OpenWRT. Esto plantea un reto considerable ya que la mayoría de estos dispositivos solo tienen 8MB o 16MB en el mejor de los casos de espacio de almacenamiento de datos. Además habría que recompilar todo el software para la arquitectura MIPS y optar por el uso de alternativas más "ligeras" como Lighttpd[93] en vez de Apache ya que estos dispositivos apenas tienen una CPU con 200Mhz. Este desarrollo permitiría integrar en el punto de acceso inalámbrico el sistema desarrollado. Otra alternativa sería implementar en el dispositivo solo la parte de control del ancho de banda y filtrado de red y utilizar un servidor RADIUS externo, y posiblemente centralizado, para autenticar a los usuarios.

[93] http://www.lighttpd.net/

- Posibilidad de definir para cada usuario filtrado de sitios web basándose en su *URL*, por ejemplo para impedir el acceso a sitios pornográficos. Podrían importarse listas de *URLs* que contuviesen un listado completo de dichos sitios "*no deseados*".

- Creación de un apartado en la interfaz de administración para monitorizar a los usuarios en tiempo real pudiendo observando a donde se conectan y la tasa de transferencia de datos en tiempo real. Además se mostrarían diversos gráficos de barras y sectores con las estadísticas por usuario y globales del sistema.

- Implementar la posibilidad de que el gestor de ancho de banda trabaje completamente en la capa 2 del modelo TCP/IP (layer2) utilizando ebtables, de esta forma no sería necesario hacer NAT y el cliente tendría la sensación de que está conectado directamente a Internet (IP Pública).

- Añadir un proxy-cache (Squid[94]) para almacenar las páginas estáticas y servirlas a los clientes de forma automática con el consiguiente ahorro de ancho de banda respecto a Internet y la mejora de calidad de servicio que de ellos se derivaría. El proxy-cache sería completamente transparente al usuario, en ningún momento se percataría de la existencia del mismo.

- Añadir soporte para VLAN (802.11Q)

- Añadir soporte para SNMP

[94] http://www.squid-cache.org/

Apéndices

A. Manual de uso

El sistema se ha diseñado intentando ser lo más intuitivo posible de forma que requiera la mínima configuración por parte del cliente. De la misma forma se ha diseñado la interfaz Web de administración. No obstante procederemos a explicar el uso, manejo y configuración del sistema.

A.1. Manual de uso del administrador de usuarios

El administrador de usuarios administrará y gestionará a los usuarios de forma remota a través de la página Web de administración. Para acceder a dicha página Web debe teclear en su navegador lo siguiente:

- **https://admin.nombre_servidor.lan/**
  Donde nombre_servidor es el nombre que se ha configurado para el servidor, por defecto será https://admin.ubalda.lan

- **https://ipdel.servidor.en.internet/**
  Tambien se permite el acceso desde la interfaz WAN para administrar el gestor de ancho de banda remotamente. Por ejemplo https://202.45.32.1/ (suponiendo que esa sea la ip pública del servidor en Internet)

Al intentar acceder a la Interfaz de administración el sistema solicitará las credenciales necesarias que han de introducirse

Usuario: altas
Contraseña: *****
La contraseña deberá solicitársela al administrador del sistema

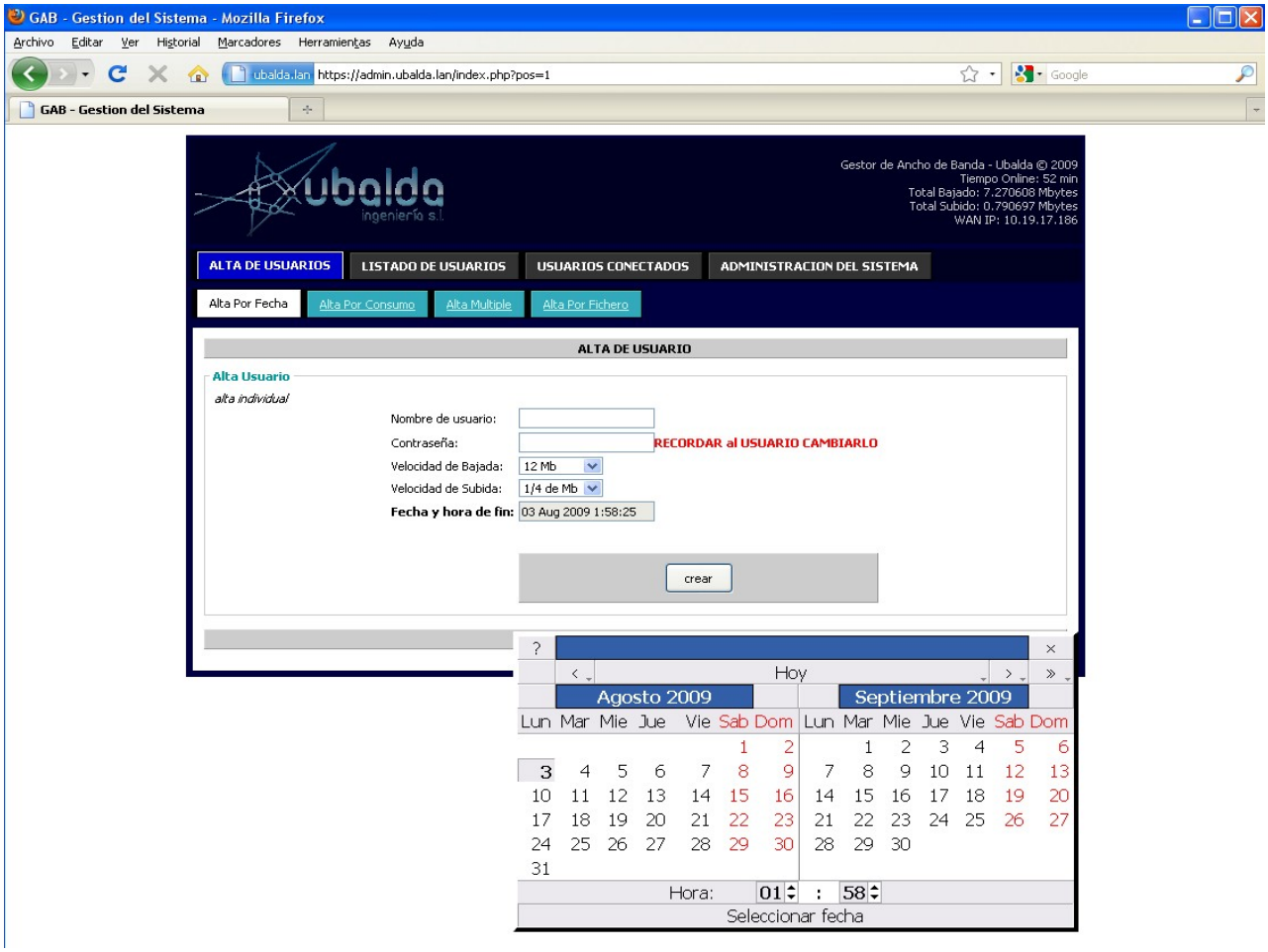
A.1.1. Alta de usuarios

El administrador de usuarios podrá dar de alta usuarios, para ello ha de acceder a la interfaz Web de administración como se ha explicado y hacer clic en el botón “ALTA DE USUARIOS” y a continuación hara clic en el tipo de alta que desee realizar, pudiendo escoger entre las siguientes opciones

- Alta de usuarios por fecha limite de uso
- Alta de usuarios por consumo tope de uso
- Alta múltiple de usuarios
- Alta por fichero

A.1.1.1. Alta de usuarios por fecha

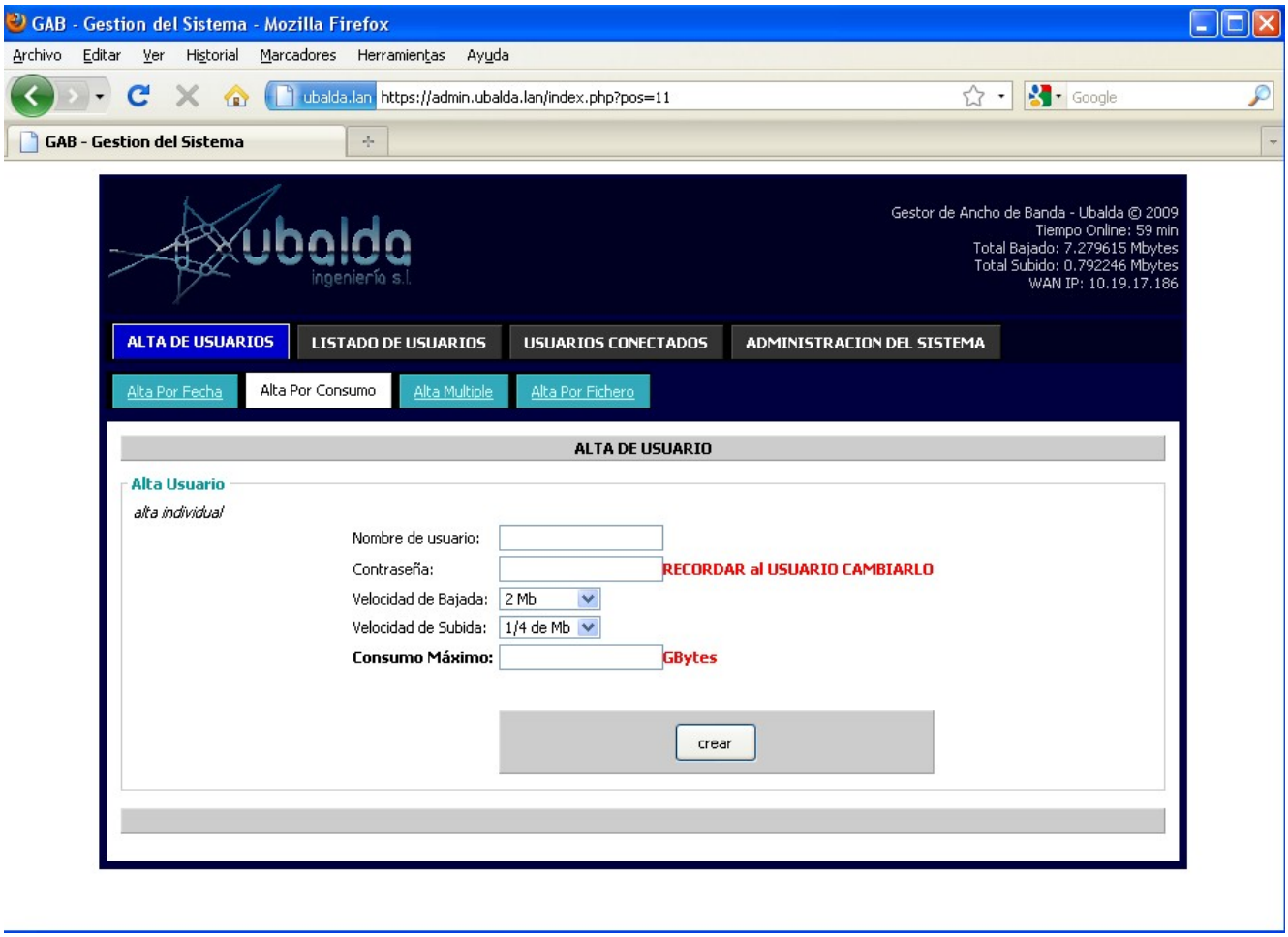
El administrador rellenará los campos necesarios, es decir, el nombre de usuario y la contraseña. Seleccionará además el ancho de banda de bajada y subida del que dispondrá el usuario y la fecha hasta la que se le permite el acceso al sistema.



Una vez rellenados los campos pulsará el botón crear y el usuario será creado si no ha habido ningún error, en caso contrario el sistema informará del error (el nombre de usuario ya existía por ejemplo) y le solicitará al administrador que lo reintente.

A.1.1.2. Alta de usuarios por consumo.

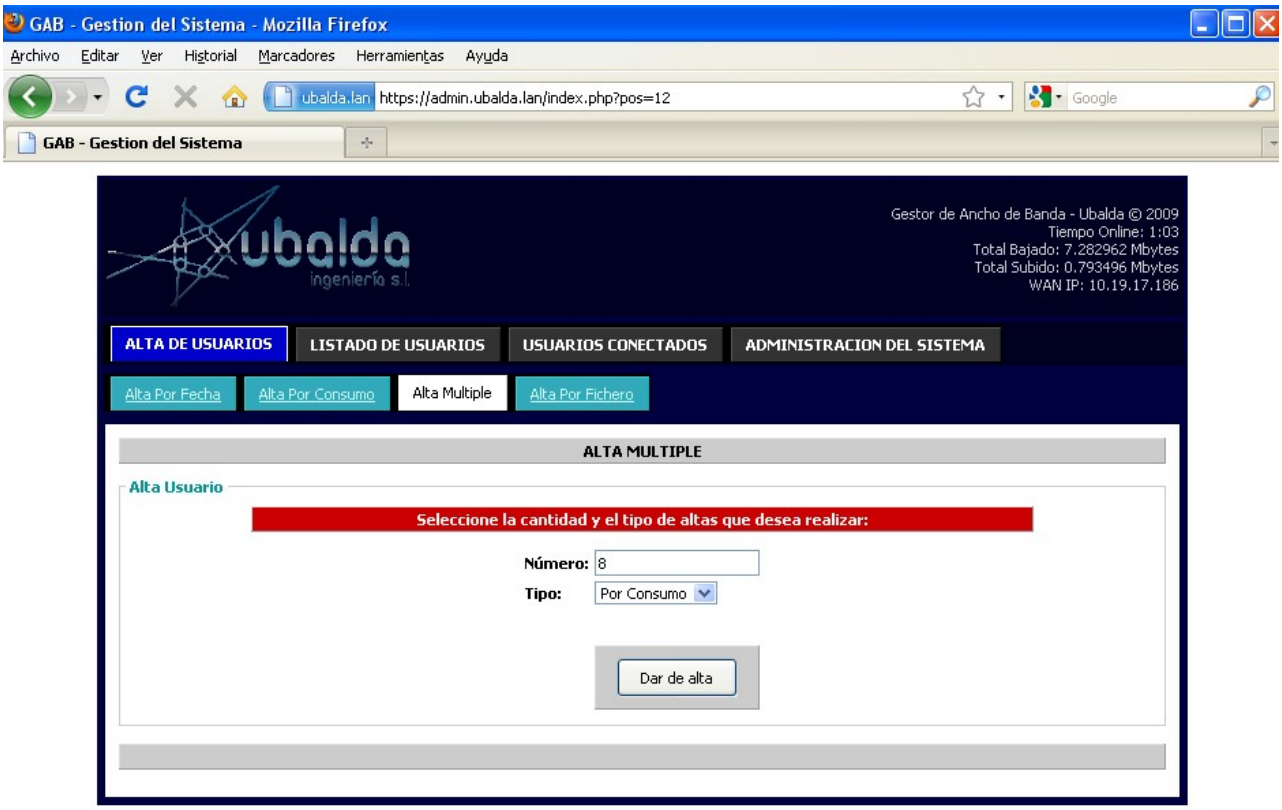
El administrador hará clic en "Alta por consumo" y rellenará los datos necesarios. En este tipo de alta se tiene que especificar el consumo máximo en *Gigabytes* del cual el usuario dispondrá.



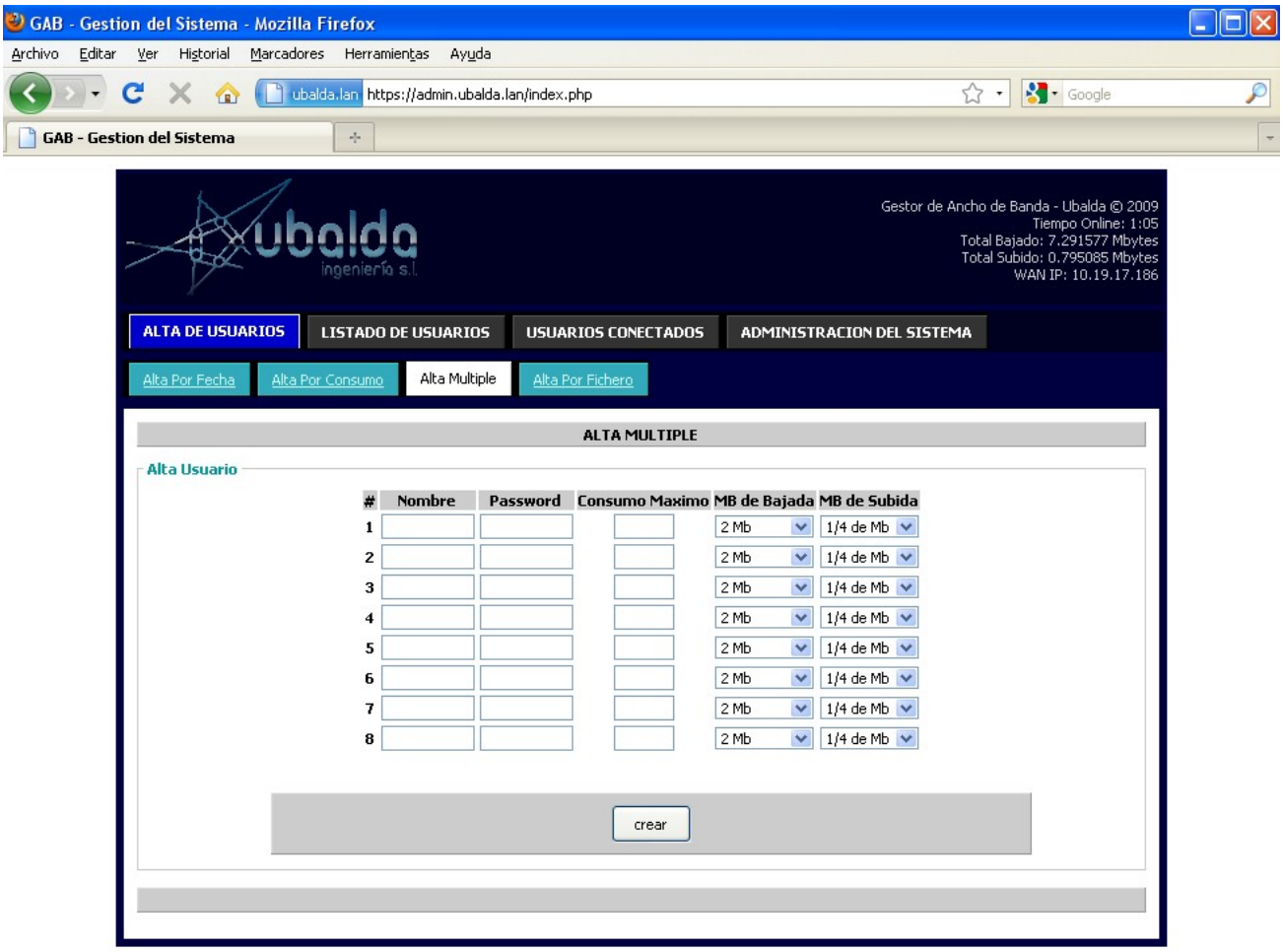
Una vez rellenados los datos necesarios el administrador pulsará en el botón "crear" y el sistema procederá a dar de alta al usuario.

A.1.1.3. Alta múltiple de usuarios.

Este tipo de alta permite al administrador dar de alta múltiples usuarios de forma sencilla. El administrador hará clic en “Alta múltiple” y seleccionará cuantos usuarios quiere dar de alta y que tipo de altas serán (por consumo máximo o por fecha tope).



El administrador pulsara el botón “dar de alta” y a continuación rellenará los datos necesarios para cada uno de los usuarios.

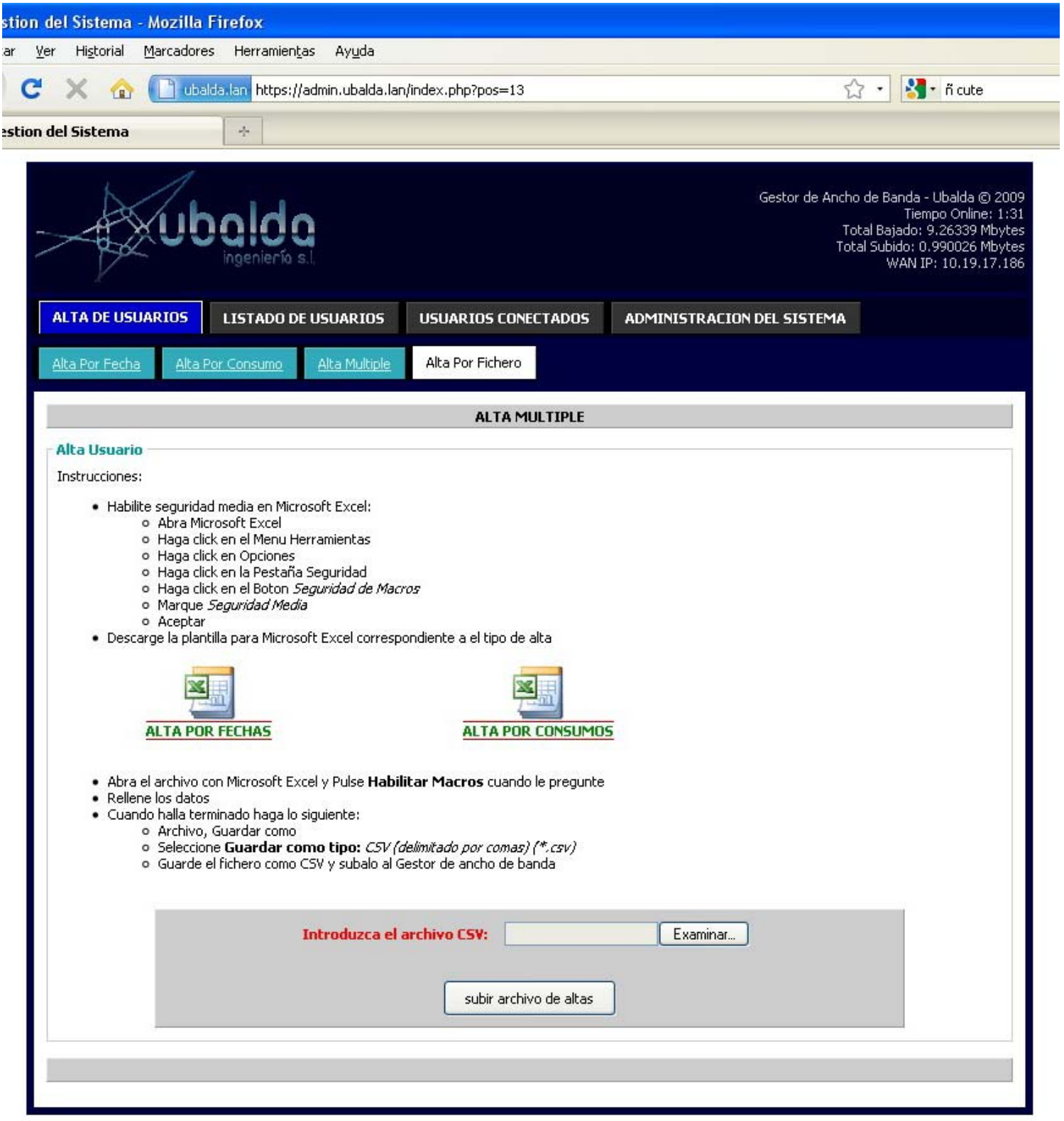


Una vez rellenados los datos pulsará el botón “crear” y el sistema procederá a dar de alta a todos los usuarios informando del éxito de la operación o de algún error como que el nombre del usuario ya existía con antelación o que la fecha/consumo introducidos no son correctos.

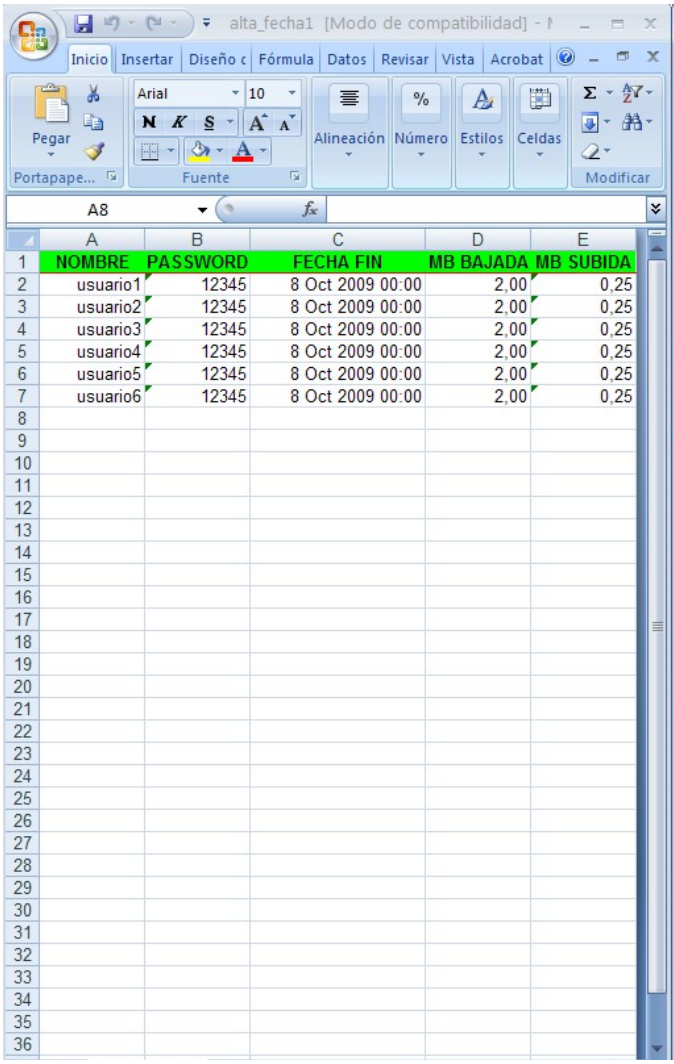
A.1.1.4. Alta por fichero.

Este tipo de alta permite al administrador dar de alta múltiples usuarios utilizando una plantilla de Microsoft Excel, lo cual puede ser útil en caso de que se disponga de una lista de usuarios que deben ser dados de alta de forma automática.

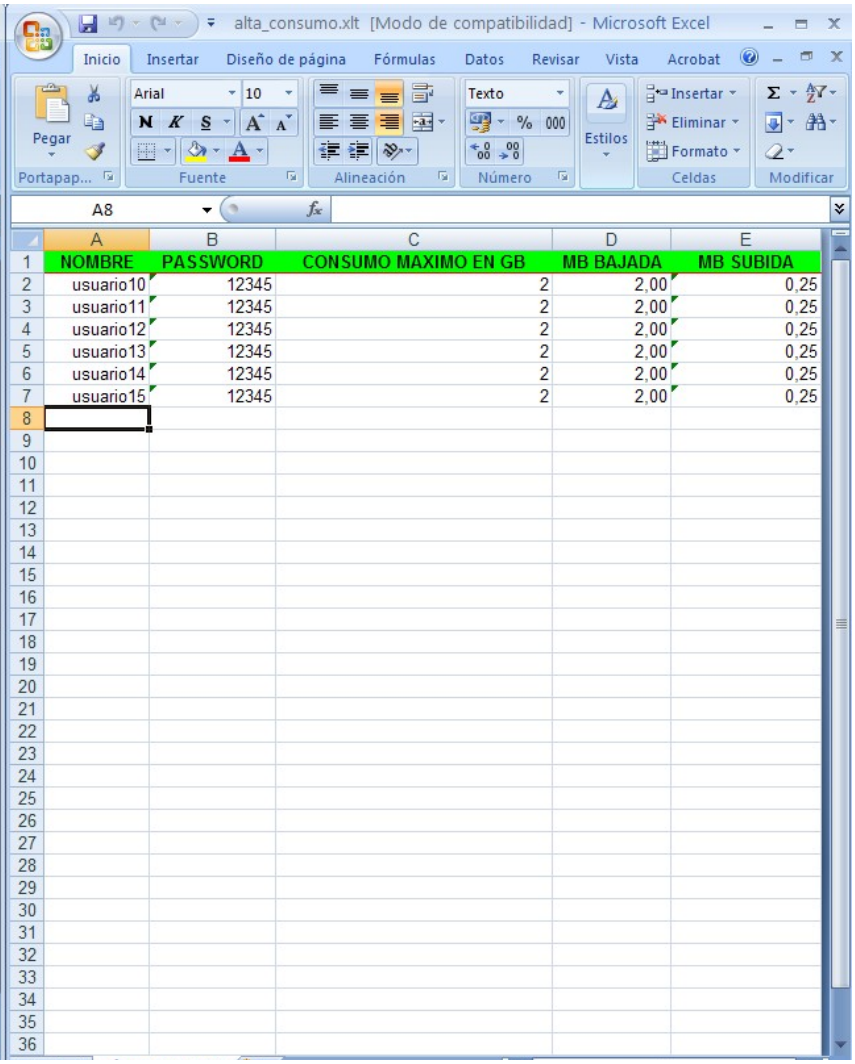
En administrador hará clic en “Alta por fichero” y seguirá las instrucciones que se indican.



El administrador rellenará la plantilla con los datos de los usuarios, la guardará en formato CSV y la subirá al sistema eligiendo el archivo correspondiente y haciendo clic en “subir archivo de altas”



	A	B	C	D	E
1	NOMBRE	PASSWORD	FECHA FIN	MB BAJADA	MB SUBIDA
2	usuario1	12345	8 Oct 2009 00:00	2,00	0,25
3	usuario2	12345	8 Oct 2009 00:00	2,00	0,25
4	usuario3	12345	8 Oct 2009 00:00	2,00	0,25
5	usuario4	12345	8 Oct 2009 00:00	2,00	0,25
6	usuario5	12345	8 Oct 2009 00:00	2,00	0,25
7	usuario6	12345	8 Oct 2009 00:00	2,00	0,25

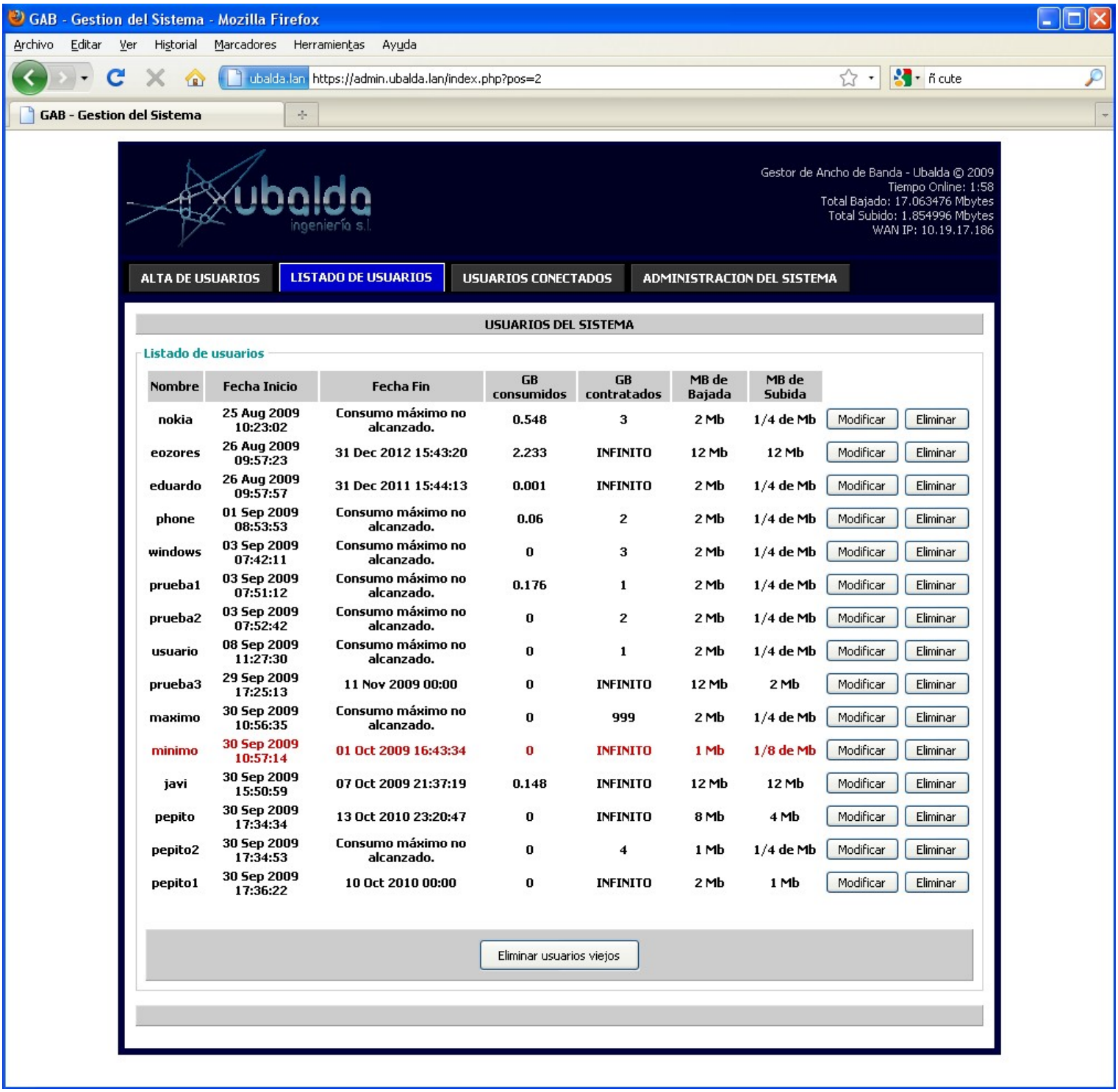


	A	B	C	D	E
1	NOMBRE	PASSWORD	CONSUMO MAXIMO EN GB	MB BAJADA	MB SUBIDA
2	usuario10	12345	2	2,00	0,25
3	usuario11	12345	2	2,00	0,25
4	usuario12	12345	2	2,00	0,25
5	usuario13	12345	2	2,00	0,25
6	usuario14	12345	2	2,00	0,25
7	usuario15	12345	2	2,00	0,25

El sistema procederá ha dar de alta todos los usuarios e informará del éxito del proceso o de las razones del error (usuario ya existente, datos no validos o formato del fichero incorrecto).

A.1.2. Listado de usuarios

El administrador de usuarios podrá ver una lista con los usuarios dados de alta en el sistema donde se indican las características del acceso de cada uno de ellos. Para acceder a dicha lista pulsara en el botón “LISTADO DE USUARIOS”

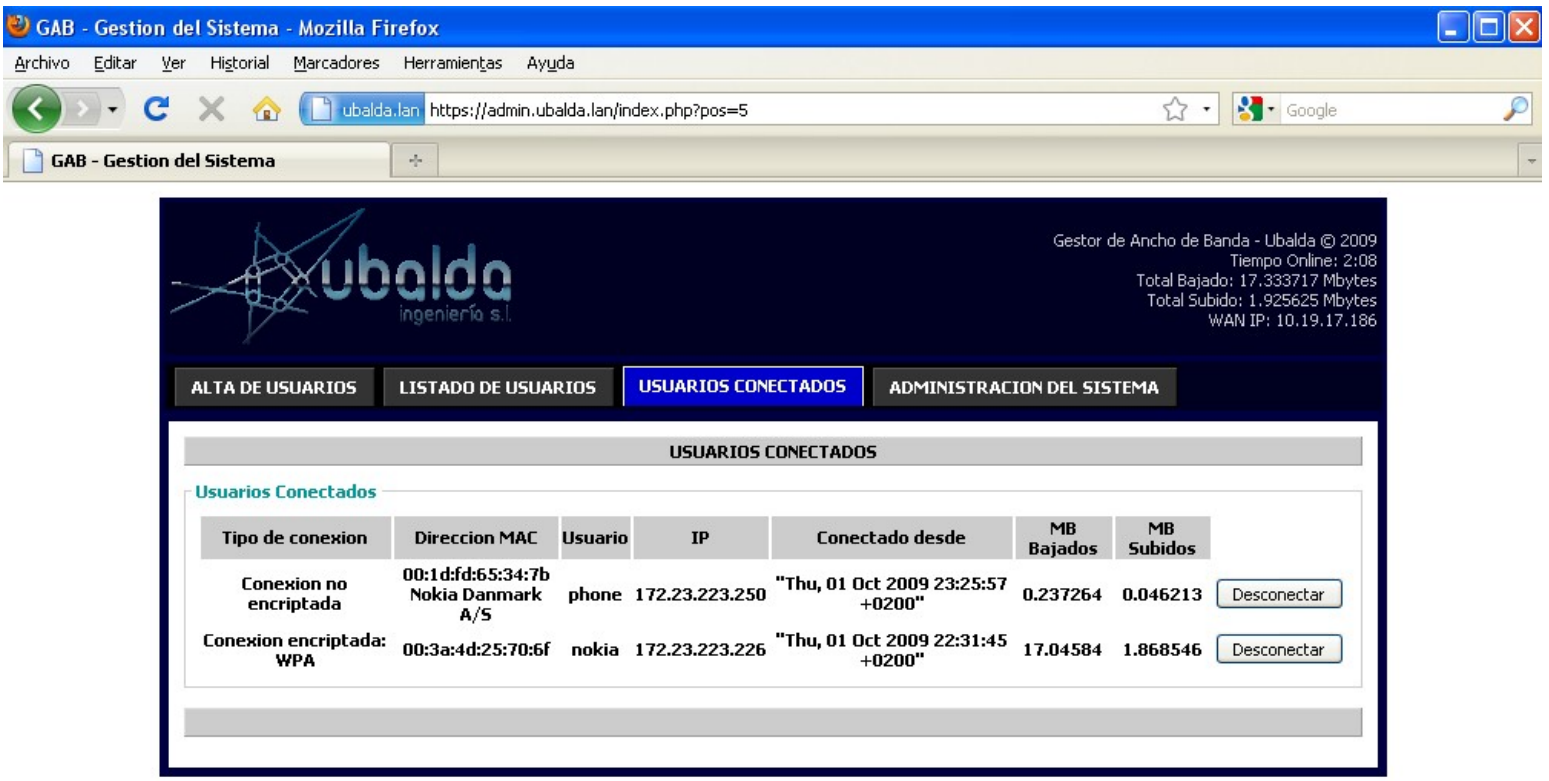


El administrador podrá cambiar los datos del usuario haciendo clic en el botón “modificar” relativo al usuario o también podrá eliminarlo haciendo clic en el botón “eliminar”.

Los usuarios que han agotado su acceso debido a que han consumido los Gigabytes que contrataron o ha llegado su fecha límite de acceso aparecerán en color rojo. El administrador podrá eliminarlos automáticamente pulsando en el botón “eliminar usuarios viejos”.

A.1.3. Usuarios conectados

El administrador de usuarios podrá ver una lista con los usuarios conectados al sistema en ese preciso instante donde se mostrará el tipo de conexión que están utilizando (normal, encriptación WPA o encriptación VPN), la dirección IP, la fecha en la que se conectaron y los Megabytes que han descargado o subido. Para acceder a dicha lista pulsara en el botón "USUARIOS CONECTADOS"



El administrador de usuarios podrá desconectar a un usuario determinado haciendo clic en botón "desconectar" relativo al usuario.

A.2. Manual de uso del administrador del sistema

El administrador del sistema podrá administrar los usuarios de la misma forma que el administrador de usuarios y además podrá gestionar las opciones de configuración del sistema Para acceder a la página Web de administración seguirá los mismos pasos que el administrador de usuarios pero introducirá como nombre de usuario "admin." en vez de "altas"

El acceso a la interfaz Web de administración se realiza tecleando en el navegador Web la siguiente dirección:

- **https://admin.nombre_servidor.lan/**
  Donde nombre_servidor es el nombre que se ha configurado para el servidor, por defecto será https://admin.ubalda.lan

- **https://ipdel.servidor.en.internet/**
  También se permite el acceso desde la interfaz WAN para administrar el gestor de ancho de banda remotamente. Por ejemplo https://202.45.32.1/ (suponiendo que esa sea la ip pública del servidor en Internet)

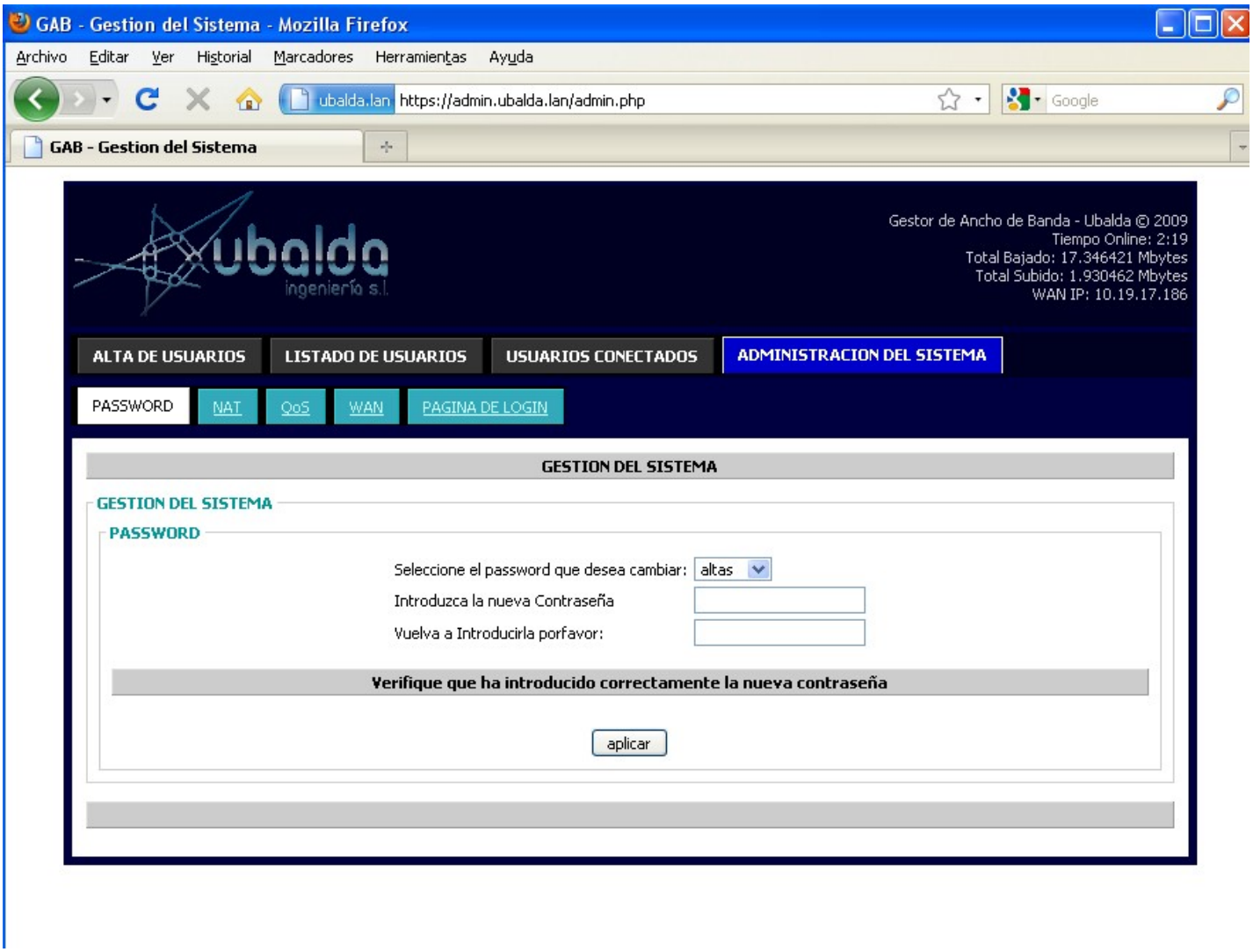
Al intentar acceder a la Interfaz de administración el sistema solicitará las credenciales necesarias que han de introducirse

Usuario: admin
Contraseña: *****
La contraseña deberá conocerla, por defecto es admin12345

A.2.1. Cambiar contraseñas

El administrador del sistema podrá cambiar su contraseña de acceso y además podrá cambiar la contraseña de acceso del administrador de usuarios. Para ello hará clic en “ADMINISTRACIÓN DEL SISTEMA” y a continuación hará clic en “PASSWORD”

Después seleccionará la contraseña que desea cambiar e introducirá la nueva dos veces. Hará clic en aplicar y el sistema verificará que la nueva contraseña es correcta (debe incluir números y letras y tener mas de 6 caracteres de longitud). Una vez verificada se establecerá la nueva contraseña.

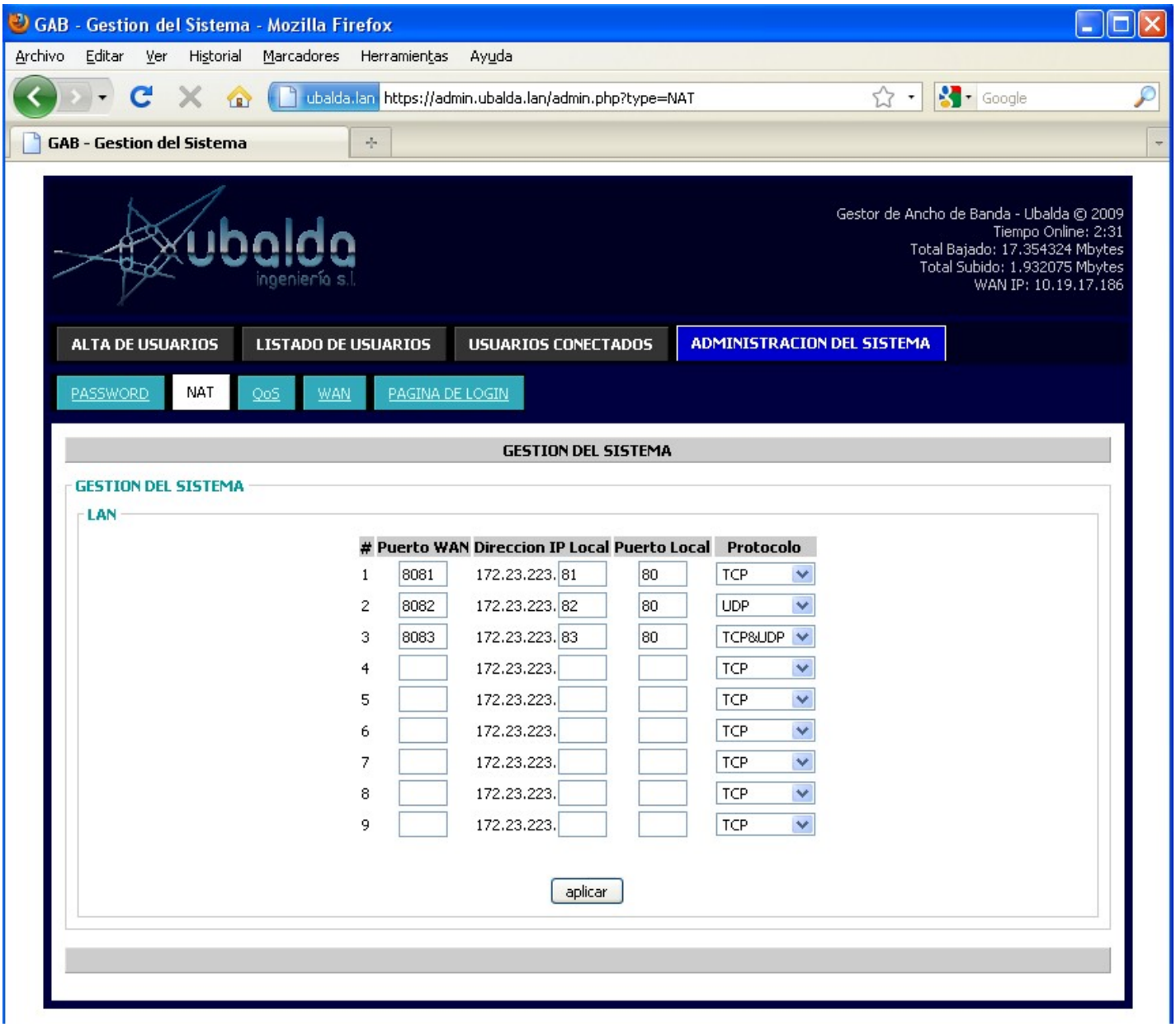


A.2.2. Definir NATeos estáticos

El administrador del sistema podrá definir una serie de *NATeos*[95]. Para ello hará clic en "ADMINISTRACIÓN DEL SISTEMA" y a continuación hará clic en "NAT"

Después introducirá para cada *NATeo* el puerto exterior (puerto WAN), la dirección IP local, el puerto local y el protocolo que puede ser TCP, UDP o ambos. Una vez definidos los *NATeos* hará clic en aplicar y el sistema procederá a aplicar los cambios.

[95] NATeo: Consiste en definir una IP de la red local del gestor de ancho de banda y un puerto al que se podrá acceder desde el exterior (Internet). Esto es útil en caso de que se quiera permitir por ejemplo el acceso a las Interfaces de administración de los puntos de acceso conectados al sistema desde el exterior.



A.2.3. Definir características de QoS

El administrador del sistema podrá definir una serie de características *QoS*[96]. Para ello hará clic en "ADMINISTRACIÓN DEL SISTEMA" y a continuación hará clic en "QoS". Dichas características quedan reflejadas en la siguiente captura de pantalla.

[96] QoS: Quality of Service. características de calidad del servicio.

A.2.4. Definir parámetros TCP/IP

El administrador del sistema podrá definir los parámetros TCP/IP del sistema, es decir, dirección IP del sistema en Internet, mascara de red, puerta de enlace y servidores DNS. Además podrá definir el nombre del servidor.

Para ello hará clic en “ADMINISTRACIÓN DEL SISTEMA” y a continuación hará clic en “WAN”. Una vez introducidos los nuevos datos el administrador hará clic en “aplicar” y el sistema procederá a reiniciarse con los nuevos parámetros.

A.2.5. Definir aspecto de la página de bienvenida

El administrador del sistema podrá definir el aspecto de la página de bienvenida y acceso de los usuarios. Para ello hará clic en “ADMINISTRACIÓN DEL SISTEMA” y a continuación hará clic en “PAGINA DE LOGIN”.

El administrador dispondrá de un editor de texto para cambiar los siguientes parámetros:

- Titulo central
- Mensaje lateral

Además podrá cambiar los logotipos izquierdo y derecho de la página de bienvenida, para ello seleccionará una imagen de su disco duro pulsando en el botón “Examinar…” relativa al logotipo que desea cambiar. El sistema se encargará de redimensionar la imagen para que quede bien centrada.

Una vez hechos los cambios deseados el administrador hará clic en el botón “Guardar cambios” tras lo cual el sistema procederá a aplicar los cambios.

Gestor de Ancho de Banda - Ubalda © 2009
Tiempo Online: 2:41
Total Bajado: 17.366737 Mbytes
Total Subido: 1.936084 Mbytes
WAN IP: 10.19.17.186

ALTA DE USUARIOS | LISTADO DE USUARIOS | USUARIOS CONECTADOS | ADMINISTRACION DEL SISTEMA

PASSWORD | NAT | QoS | WAN | PAGINA DE LOGIN

GESTION DEL SISTEMA

GESTION DEL SISTEMA

PAGINA DE LOGIN

Tiitulo central

Estilos | Formato | Fuente | Tamaño

Universidad de
La Coruña

Ruta:

Mensaje lateral

Estilos | Párrafo | Fuente | 3 (12pt)

Ha entrado en el Sistema de Acceso a Internet de la Universidad de La Coruña.

Si todavía no está dado de alta, pida en Secretaría un nombre de usuario y contraseña.

Puede acceder al sistema mediante una conexión segura y encriptada creando una VPN al servidor vpn.ubalda.lan con su nombre de usuario y contraseña.

Ruta: p » span » span

Logo izquierdo

Examinar...

Logo derecho

Examinar...

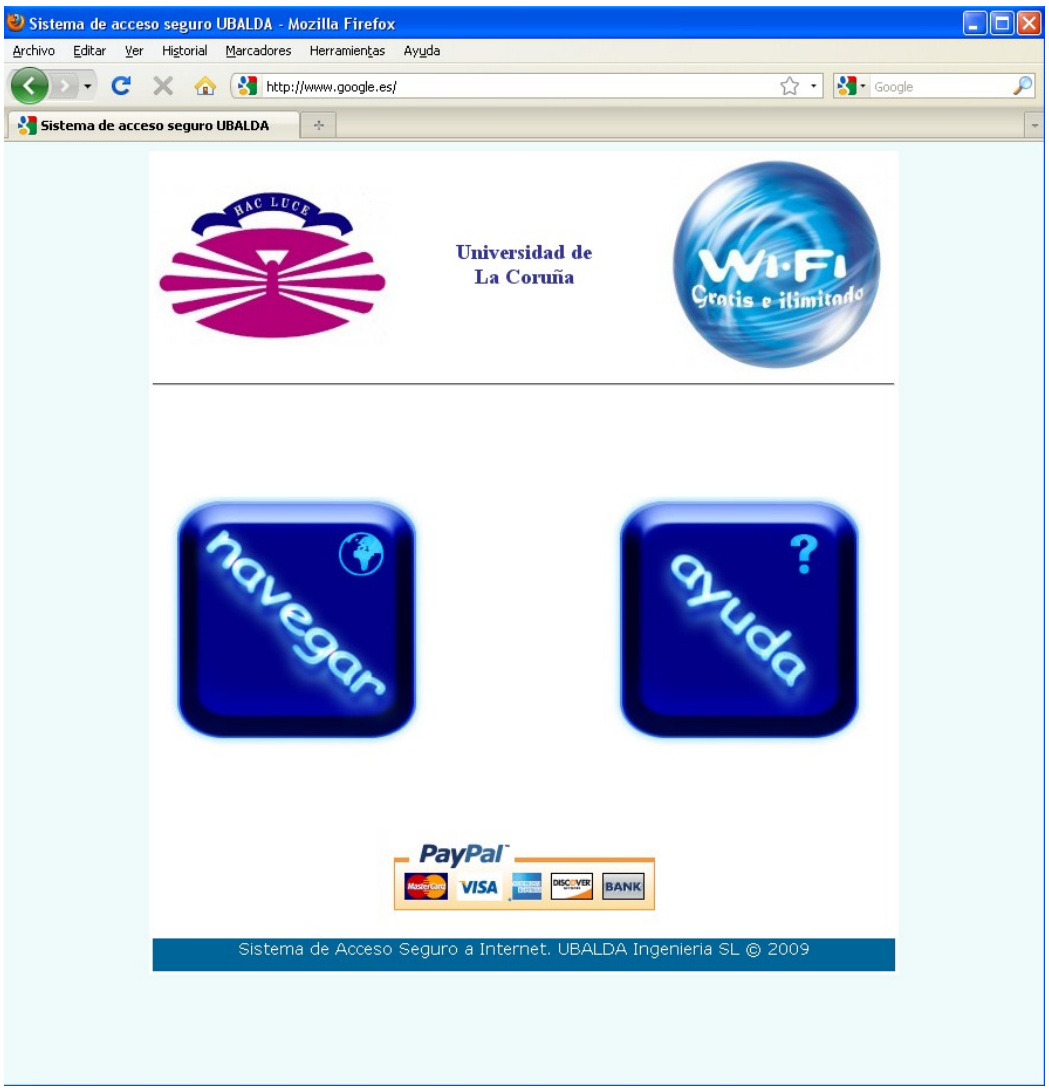
Guardar cambios

A.3. Manual de uso del usuario

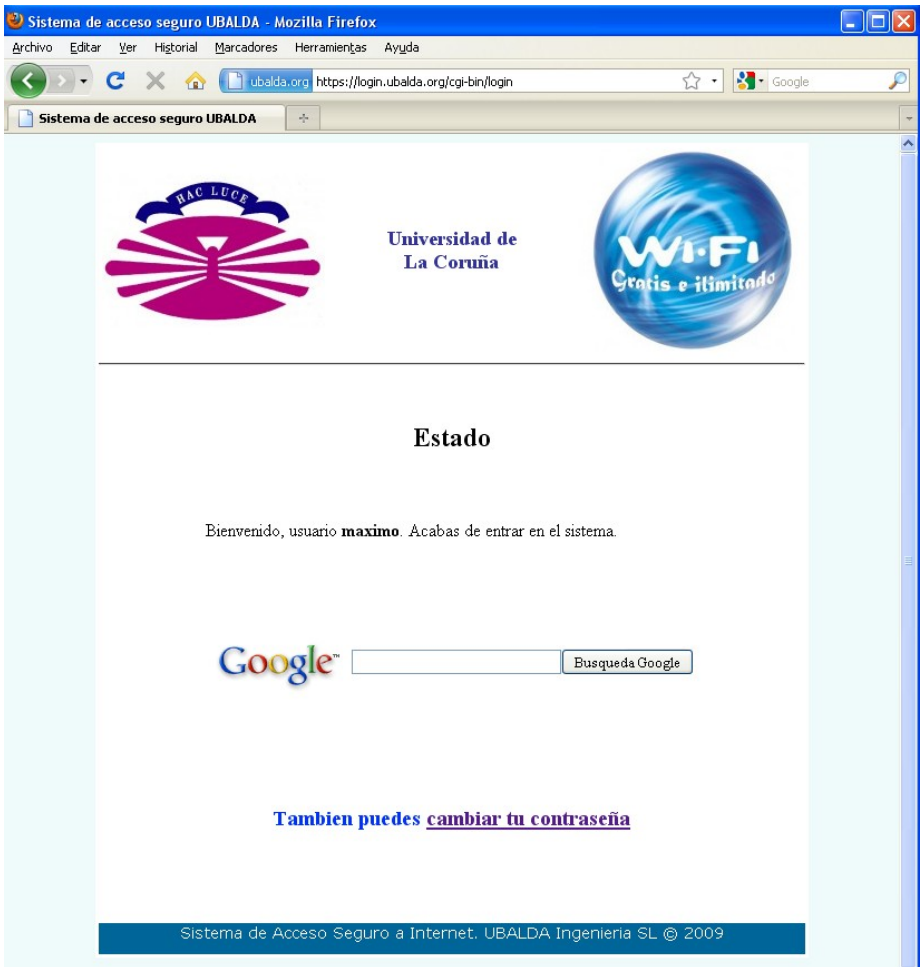
El sistema intenta ser todo lo intuitivo posible para el usuario, no obstante se procederá a describir el uso de la Página de bienvenida y como configurar la conexión VPN y WPA.

A.3.1. Manual de uso de la página de bienvenida

El usuario al intentar conectarse al sistema se encontrará con una página de bienvenida donde podrá obtener ayuda sobre como configurar su conexión y utilizar el sistema. Para navegar pulsara en el botón “navegar”.



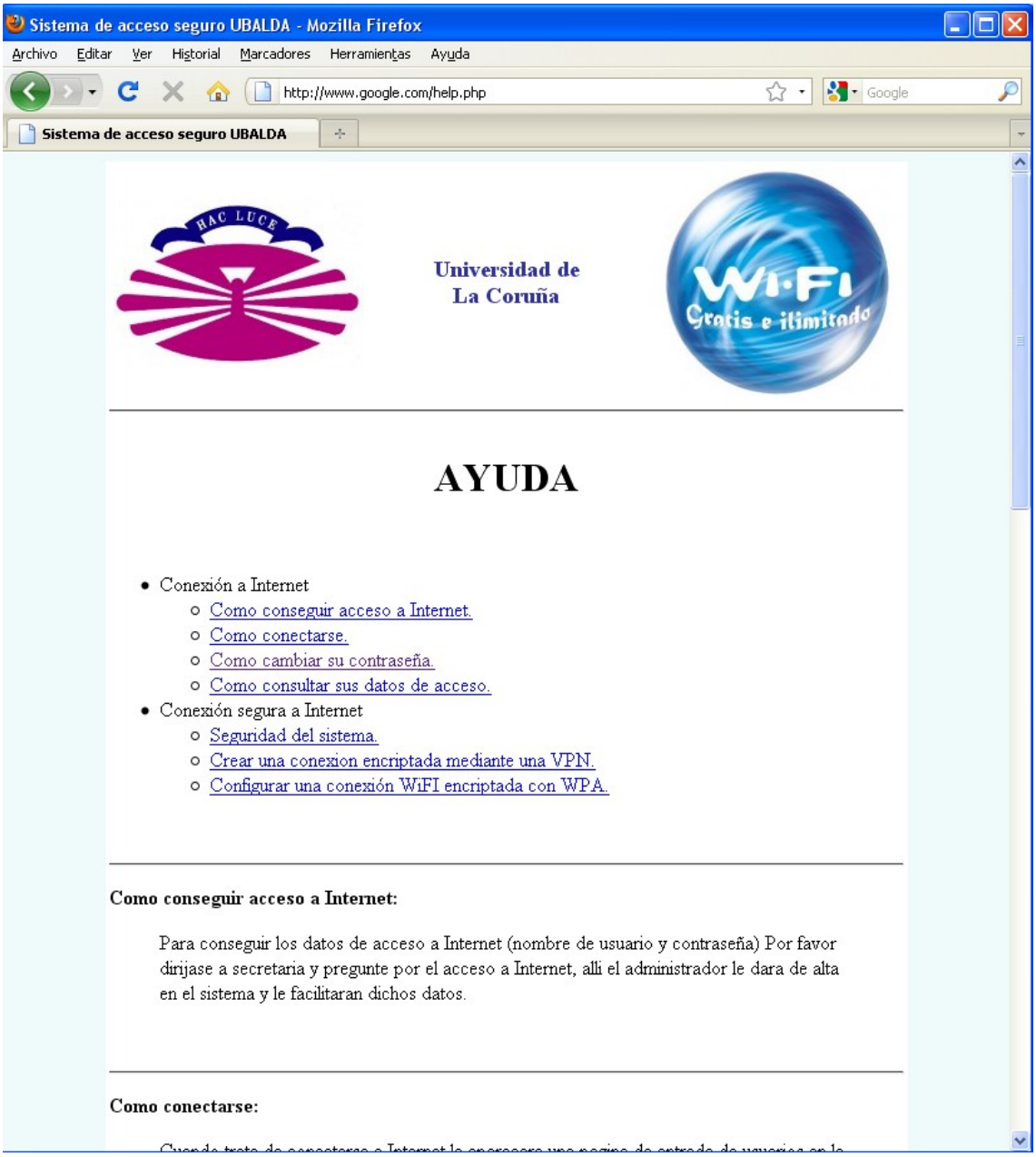
Después deberá introducir sus credenciales y pulsar en el botón “conectarse” el sistema comprobará los datos y le dará acceso al sistema o mostrará un mensaje de error en caso contrario.


Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://login.ubalda.org/cgi-bin/login
Sistema de acceso seguro UBALDA
Universidad de La Coruña
Wi-Fi Gratis e ilimitado
Estado
Bienvenido, usuario maximo. Acabas de entrar en el sistema.
Google
Busqueda Google
Tambien puedes cambiar tu contraseña
Sistema de Acceso Seguro a Internet. UBALDA Ingenieria SL © 2009


Sistema de acceso seguro UBALDA - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
https://login.ubalda.org/passwd.php?user=maximo
Sistema de acceso seguro UBALDA
Universidad de La Coruña
Wi-Fi Gratis e ilimitado
Instrucciones:
Introduzca su usuario, la contraseña antigua y su nueva contraseña dos veces. Ante cualquier duda, consulte con su administrador de sistemas.
Cambiar la contraseña
Identificador de usuario:
Contraseña antigua:
Nueva contraseña:
Confirme la nueva contraseña:
Cambiar
Volver
Sistema de Acceso Seguro a Internet. UBALDA Ingenieria SL © 2009

El usuario también puede modificar su contraseña haciendo clic en “cambiar contraseña”. Luego deberá introducir su contraseña antigua y la nueva deseada. El usuario también puede desconectarse a si mismo del sistema haciendo clic en “desconectar”. Se le pedirá que introduzca su contraseña para poder ser desconectado.

El usuario además podrá consultar la página de ayuda del sistema en caso de que tenga cualquier tipo de duda sobre la configuración o uso del mismo.



A.3.2. Configuración de la VPN

El usuario debe configurar una VPN usando protocolo PPTP y autentificación mschap2 al servidor ***vpn.nombre_del_servidor.lan*** (*vpn.ubalda.lan* por defecto) y utilizar sus credenciales (nombre de usuario y contraseña) para obtener acceso. Al hacerlo automáticamente obtendrá acceso a Internet con una conexión encriptada y no será necesario que utilice la página de bienvenida ya que al conectarse a través de la VPN el sistema automáticamente le

da acceso a Internet sin necesidad de que introduzca sus credenciales en la página de bienvenida.

Para configurar la VPN en Windows XP:

1. Vaya al Panel de Control
2. Haga click en Conexiones de Red
3. Haga click en "Crear una conexion nueva" y seleccione "Crear una conexion a la red de su trabajo"
4. Introduzca un nombre para la nueva conexión VPN y la dirección del servidor VPN (vpn.ubalda.lan por defecto)
5. Introduzca sus credenciales de usuario (nombre y contraseña)
6. Pulse en conectar.

A.3.3. Configuración de la red inalámbrica con encriptación WPA Empresarial

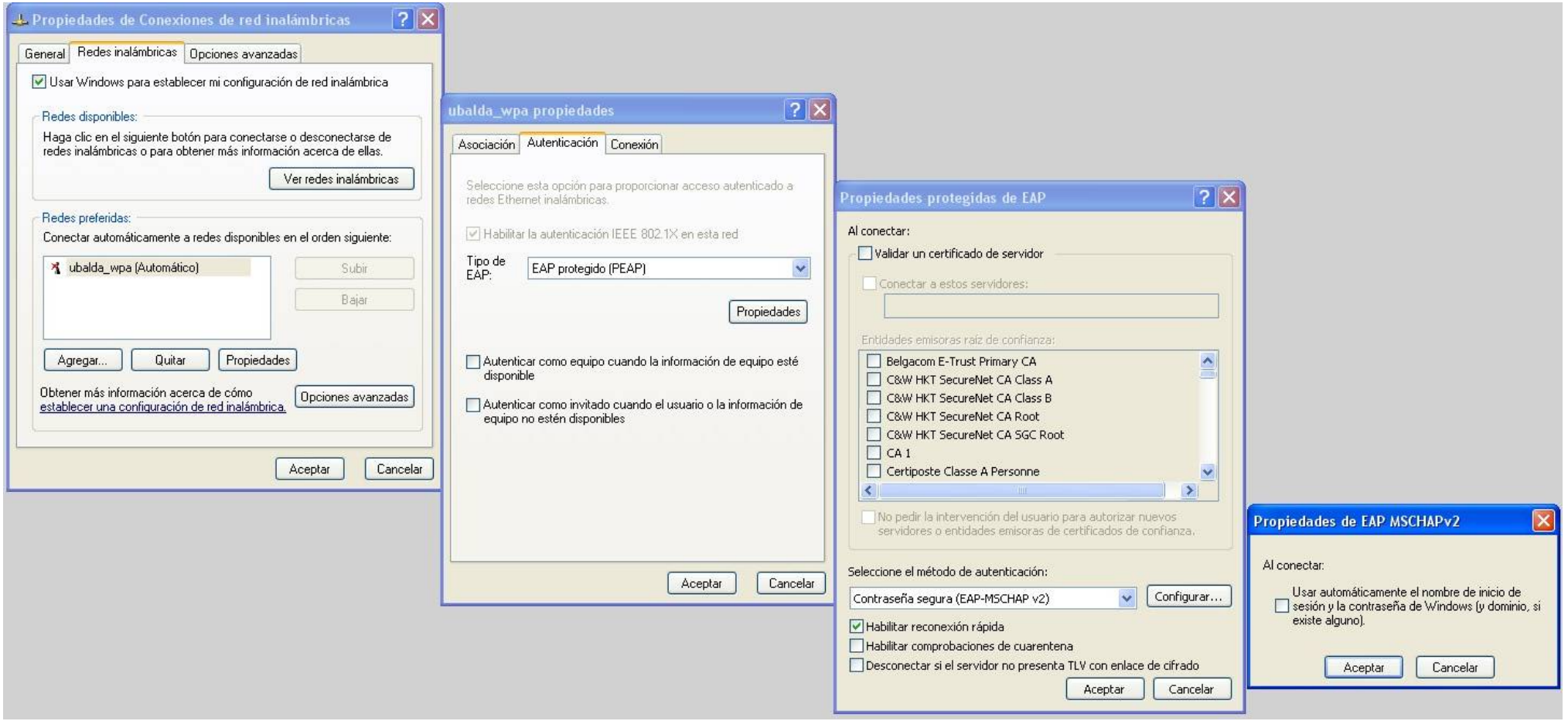
El usuario puede conectarse al sistema de forma inalámbrica utilizando encriptación WPA de tipo empresarial (WPA-PEAP). Para ello debe configurar una conexión inalámbrica de tipo WiFi (802.11) con encriptación WPA (TKIP, AES o automática) y como modo de autentificación debe usar PEAP. Deberá además introducir sus credenciales de usuario (nombre y contraseña) para la autentificación y no debe comprobar el certificado del servidor ya que este no ha sido firmado por una autoridad de certificación [97] . Existe la posibilidad de adquirir un certificado firmado por una autoridad de certificación pero su coste es elevado[98].

Para configurar la VPN en Windows XP:

7. Busque redes inalámbricas y conectese a la red con encriptación wpa (***ubalda_wpa*** por defecto)
8. Haga click en el mensaje *"windows no pudo encontrar un certificado valido"*
9. Haga click en *Propiedades* y a continuación vaya a la pestaña *redes inalámbricas*

[97] http://es.wikipedia.org/wiki/Autoridad_de_certificación
[98] http://www.verisign.com/ssl/buy-ssl-certificates/specialized-ssl-certificates/wireless-lan-security/

10. Seleccione la red inalámbrica (ubalda_wpa) y haga clic en la pestaña *autentificación*.
11. No marque la casilla *"Autentificar como equipo cuando la información de mi equipo este disponible"*
12. No marque la casilla *"Autentificar como invitado cuando el usuario o la información de equipo no esten disponibles"*
13. Seleccione en el apartado *tipo de EAP* la opción *EAP protegido (PEAP)*
14. Haga clic en *Propiedades* y no seleccione la opción *"Validar un certificado de servidor"*
15. Seleccione como método de autentificación EAP-MSCHAPv2
16. Haga clic en configurar y no marque la opción "usar automáticamente el nombre de inicio de sesión de windows"
17. Haga clic en aceptar en todos los cuadros de diálogo abiertos
18. Vuelva a conectarse a la red inalámbrica.
19. El sistema le pedirá que introduzca sus credenciales
20. Introduzca el nombre de usuario y contraseña.
21. Pulse en conectar


Propiedades de Conexiones de red inalámbricas
General
Redes inalámbricas
Opciones avanzadas
Usar Windows para establecer mi configuración de red inalámbrica
Redes disponibles:
Haga clic en el siguiente botón para conectarse o desconectarse de redes inalámbricas o para obtener más información acerca de ellas.
Ver redes inalámbricas
Redes preferidas:
Conectar automáticamente a redes disponibles en el orden siguiente:
ubalda_wpa (Automático)
Subir
Bajar
Agregar...
Quitar
Propiedades
Obtener más información acerca de cómo establecer una configuración de red inalámbrica.
Opciones avanzadas
Aceptar
Cancelar
ubalda_wpa propiedades
Asociación
Autenticación
Conexión
Seleccione esta opción para proporcionar acceso autenticado a redes Ethernet inalámbricas.
Habilitar la autenticación IEEE 802.1X en esta red
Tipo de EAP:
EAP protegido (PEAP)
Propiedades
Autenticar como equipo cuando la información de equipo esté disponible
Autenticar como invitado cuando el usuario o la información de equipo no estén disponibles
Aceptar
Cancelar
Propiedades protegidas de EAP
Al conectar:
Validar un certificado de servidor
Conectar a estos servidores:
Entidades emisoras raíz de confianza:
Belgacom E-Trust Primary CA
C&W HKT SecureNet CA Class A
C&W HKT SecureNet CA Class B
C&W HKT SecureNet CA Root
C&W HKT SecureNet CA SGC Root
CA 1
Certiposte Classe A Personne
No pedir la intervención del usuario para autorizar nuevos servidores o entidades emisoras de certificados de confianza.
Seleccione el método de autenticación:
Contraseña segura (EAP-MSCHAP v2)
Configurar...
Habilitar reconexión rápida
Habilitar comprobaciones de cuarentena
Desconectar si el servidor no presenta TLV con enlace de cifrado
Aceptar
Cancelar
Propiedades de EAP MSCHAPv2
Al conectar:
Usar automáticamente el nombre de inicio de sesión y la contraseña de Windows (y dominio, si existe alguno).
Aceptar
Cancelar

B. Manual de instalación

Se va a proceder a describir la instalación de los componentes necesarios para el funcionamiento del sistema. Muchos de estos componentes necesitan que su código fuente sea *"parcheado"* y compilado para añadir soporte para las características que se necesitan para el funcionamiento correcto del presente proyecto.

Debido a la alta integración que existe entre los componentes resulta complicado la instalación del sistema a partir de cero. El autor recomienda utilizar el sistema ya instalado y configurado. Para ello se incluye un fichero comprimido con todo el sistema listo para funcionar.

B.1. Instalación del sistema a partir de cero

Tomaremos como distribución base del sistema Ubuntu 8.04LTS Server que puede ser descargada desde

http://ftp.udc.es/ubuntu-releases/8.04.3/ubuntu-8.04.3-server-i386.iso

B.1.1. Instalación de paquetes genéricos

El software en Ubuntu se gestiona en "paquetes". Para instalar un paquete se utiliza la orden

sudo apt-get install *nombre_del_paquete*

Pueden instalarse múltiples paquetes de una sola vez escribiendo los nombres uno tras otro

sudo apt-get install *paquete1 paquete2 paquete3 …*

La utilidad apt-get se encarga de descargarse el paquete desde Internet e instalarlo. A continuación se expone la lista de paquetes necesarios para ser instalados:

NOMBRE PAQUETE	VERSION	DESCRIPCIÓN
acpid	1.0.4-5ubuntu9.3	Utilities for using ACPI power management
apache2	2.2.8-1ubuntu0.5	Next generation, scalable, extendable web se
apache2-mpm-prefork	2.2.8-1ubuntu0.5	Traditional model for Apache HTTPD
apache2-utils	2.2.8-1ubuntu0.5	utility programs for webservers
apache2.2-common	2.2.8-1ubuntu0.5	Next generation, scalable, extendable web se
libapache2-mod-perl2	2.0.3-2ubuntu2	Integration of perl with the Apache2 web ser
libapache2-mod-php5	5.2.4-2ubuntu5.6	server-side, HTML-embedded scripting languag
apache2-mpm-prefork	2.2.8-1ubuntu0.5	Traditional model for Apache HTTPD
apache2-utils	2.2.8-1ubuntu0.5	utility programs for webservers
apache2.2-common	2.2.8-1ubuntu0.5	Next generation, scalable, extendable web se
bcrelay	1.3.4-2ubuntu1	Broadcast relay daemon
bind9	1:9.4.2.dfsg.P2-2ubuntu0.1	Internet Domain Name Server
bind9-doc	1:9.4.2.dfsg.P2-2ubuntu0.1	Documentation for BIND
bind9-host	1:9.4.2.dfsg.P2-2ubuntu0.1	Version of 'host' bundled with BIND 9.X
libbind9-30	1:9.4.2.dfsg.P2-2ubuntu0.1	BIND9 Shared Library used by BIND
bind9-doc	1:9.4.2.dfsg.P2-2ubuntu0.1	Documentation for BIND
binutils	2.18.1~cvs20080103-0ubuntu1	The GNU assembler, linker and binary utiliti
conntrack	1.00~beta2-1	Program to modify the conntrack tables
libnetfilter-conntrack1	0.0.81-1	Netfilter netlink-conntrack library
dhcp3-server	3.0.6.dfsg-1ubuntu9	DHCP server for automatic IP address assignm
libdevel-symdump-perl	2.03-3	Perl module for inspecting perl's symbol tab
tcpdump	3.9.8-2	A powerful tool for network monitoring and d
ebtables	2.0.8.2-2	Ethernet bridge frame table administration
ipcalc	0.41-1	parameter calculator for IPv4 addresses
knockd	0.5-2ubuntu2	small port-knock daemon
lftp	3.6.1-1	Sophisticated command-line FTP/HTTP client p
libapache2-mod-perl2	2.0.3-2ubuntu2	Integration of perl with the Apache2 web ser
libapache2-mod-php5	5.2.4-2ubuntu5.6	server-side, HTML-embedded scripting languag
libapr1	1.2.11-1	The Apache Portable Runtime Library
libaprutil1	1.2.12+dfsg-3	The Apache Portable Runtime Utility Library
libauthen-radius-perl	0.13-1	user authentication against radius
libc6-dev	2.7-10ubuntu5	GNU C Library: Development Libraries and Hea
libck-connector0	0.2.3-3ubuntu5	ConsoleKit libraries
libconvert-asn1-perl	0.20-1	Perl module for encoding and decoding ASN.1
libcrypt-smbhash-perl	0.12-2	generate LM/NT hash of a password for samba
libdate-manip-perl	5.48-1	a perl library for manipulating dates
libdb4.2	4.2.52+dfsg-4	Berkeley v4.2 Database Libraries [runtime]
libdevel-symdump-perl	2.03-3	Perl module for inspecting perl's symbol tab
libdigest-md4-perl	1.5.dfsg-1.2	MD4 Message Digest for Perl
libdigest-sha1-perl	2.11-2	NIST SHA-1 message digest algorithm
libfreetype6	2.3.5-1ubuntu4.8.04.2	FreeType 2 font engine, shared library files
libglib2.0-0	2.16.6-0ubuntu1.1	The GLib library of C routines
libio-socket-ssl-perl	1.02-1	Perl module implementing object oriented int
libiodbc2	3.52.6-1ubuntu1	iODBC Driver Manager
libjcode-pm-perl	2.06-1	Perl extension interface to convert Japanese
libjpeg62	6b-14	The Independent JPEG Group's JPEG runtime li
libltdl3	1.5.26-1ubuntu1	A system independent dlopen wrapper for GNU
libmng1	1.0.9-1	Multiple-image Network Graphics library
libneon27	0.27.2-1ubuntu0.1	An HTTP and WebDAV client library
libnet-ldap-perl	1:0.34-1	A Client interface to LDAP servers
libnet-ssleay-perl	1.30-1	Perl module for Secure Sockets Layer (SSL)
libnetfilter-conntrack1	0.0.81-1	Netfilter netlink-conntrack library
libnfnetlink0	0.0.30-2	Netfilter netlink library
libnl-dev	1.1-1	Development library and headers for libnl
libnl1	1.1-1	Library for dealing with netlink sockets

libpcre3	7.4-1ubuntu2.1	Perl 5 Compatible Regular Expression Library
libperl5.8	5.8.8-12ubuntu0.4	Shared Perl library
libradius1	0.3.2-11	/bin/login replacement with RADIUS. Shared l
libsensors3	1:2.10.5-3ubuntu1	library to read temperature/voltage/fan sens
libslp1	1.2.1-7.1	OpenSLP libraries
libsnmp-base	5.4.1~dfsg-4ubuntu4.2	SNMP (Simple Network Management Protocol) MI
libsnmp15	5.4.1~dfsg-4ubuntu4.2	SNMP (Simple Network Management Protocol) li
libsvn1	1.4.6dfsg1-2ubuntu1.1	Shared libraries used by Subversion
libt1-5	5.1.1-5	Type 1 font rasterizer library - runtime
libtime-modules-perl	2006.0814-1	Various Perl modules for time/date manipulat
libtimedate-perl	1.1600-9	Time and date functions for Perl
libunicode-map-perl	0.112-10	Perl module for mapping charsets from and to
libunicode-map8-perl	0.12-3	Perl module to map 8bit character sets to Un
libunicode-maputf8-perl	1.11-2	Perl module for conversing between any chara
libunicode-string-perl	2.09-3	Perl modules for Unicode strings
libxcb-xlib0	1.1-1ubuntu1	X C Binding, Xlib/XCB interface library
libxcb1	1.1-1ubuntu1	X C Binding
linux-libc-dev	2.6.24-24.60	Linux Kernel Headers for development
makedev	2.3.1-84ubuntu1	creates device files in /dev
mirror	2.9-53.1	keeps FTP archives up-to-date
multitail	5.2.0-1	view multiple logfiles windowed on console
nictools-pci	1.3.8-1	Diagnostic tools for many PCI ethernet cards
odbcinst1debian1	2.2.11-16build1	Support library and helper program for acces
openssh-blacklist	0.1-1ubuntu0.8.04.1	list of blacklisted OpenSSH RSA and DSA keys
openssh-server	1:4.7p1-8ubuntu1.2	secure shell server, an rshd replacement
openssl	0.9.8g-4ubuntu3.4	Secure Socket Layer (SSL) binary and related
patch	2.5.9-4	Apply a diff file to an original
php-auth	1.5.4-1	PHP PEAR modules for creating an authenticat
php-net-ldap	1:1.0.0-1	a OO interface for searching and manipulatin
php-pear	5.2.4-2ubuntu5.5	PEAR - PHP Extension and Application Reposit
libapache2-mod-php5	5.2.4-2ubuntu5.6	server-side, HTML-embedded scripting languag
php5	5.2.4-2ubuntu5.5	server-side, HTML-embedded scripting languag
php5-cli	5.2.4-2ubuntu5.6	command-line interpreter for the php5 script
php5-common	5.2.4-2ubuntu5.6	Common files for packages built from the php
php5-gd	5.2.4-2ubuntu5.6	GD module for php5
php5-ldap	5.2.4-2ubuntu5.6	LDAP module for php5
php5-cli	5.2.4-2ubuntu5.6	command-line interpreter for the php5 script
php5-common	5.2.4-2ubuntu5.6	Common files for packages built from the php
php5-gd	5.2.4-2ubuntu5.6	GD module for php5
php5-ldap	5.2.4-2ubuntu5.6	LDAP module for php5
phpldapadmin	1.1.0.4-2ubuntu1	web based interface for administering LDAP s
pptpd	1.3.4-2ubuntu1	PoPToP Point to Point Tunneling Server
radiusclient1	0.3.2-11	/bin/login replacement which uses the RADIUS
libcrypt-smbhash-perl	0.12-2	generate LM/NT hash of a password for samba
samba	3.0.28a-1ubuntu4.7	a LanManager-like file and printer server fo
samba-common	3.0.28a-1ubuntu4.7	Samba common files used by both the server a
samba-doc	3.0.28a-1ubuntu4.7	Samba documentation
samba-common	3.0.28a-1ubuntu4.7	Samba common files used by both the server a
samba-doc	3.0.28a-1ubuntu4.7	Samba documentation
sleuthkit	2.09-2	Tools for forensics analysis
smbldap-tools	0.9.4-1	Scripts to manage Unix and Samba accounts st
sshfs	1.9-1	filesystem client based on SSH File Transfer
subversion	1.4.6dfsg1-2ubuntu1.1	Advanced version control system
unixodbc	2.2.11-16build1	ODBC tools libraries
whois	4.7.24	the GNU whois client

B.1.2. Instalación de paquetes especiales

Después es necesario la instalación de los paquetes parcheados, ya que los incluidos en los repositorios de Ubuntu no tienen las características necesarias. Se incluye el código fuente y los paquetes precompilados en formato .deb dentro del cdrom del proyecto. Estos paquetes son los siguientes:

NOMBRE PAQUETE	VERSION	DESCRIPCIÓN
freeradius	2.1.0-1ubalda2~hardy1	a high-performance and highly configurable R
freeradius-common	2.1.0-1ubalda2~hardy1	FreeRadius common files
freeradius-dbg	2.1.0-1ubalda2~hardy1	a high-performance and highly configurable R
freeradius-dialupadmin	2.1.0-1ubalda2~hardy1	set of PHP scripts for administering a FreeR
freeradius-iodbc	2.1.0-1ubalda2~hardy1	iODBC module for FreeRADIUS server
freeradius-krb5	2.1.0-1ubalda2~hardy1	kerberos module for FreeRADIUS server
freeradius-ldap	2.1.0-1ubalda2~hardy1	LDAP module for FreeRADIUS server
freeradius-mysql	2.1.0-1ubalda2~hardy1	MySQL module for FreeRADIUS server
freeradius-postgresql	2.1.0-1ubalda2~hardy1	PostgreSQL module for FreeRADIUS server
freeradius-utils	2.1.0-1ubalda2~hardy1	FreeRadius client utilities
libfreeradius2	2.1.0-1ubalda2~hardy1	FreeRADIUS shared library
freeradius-common	2.1.0-1ubalda2~hardy1	FreeRadius common files
freeradius-dbg	2.1.0-1ubalda2~hardy1	a high-performance and highly configurable R
freeradius-dialupadmin	2.1.0-1ubalda2~hardy1	set of PHP scripts for administering a FreeR
freeradius-iodbc	2.1.0-1ubalda2~hardy1	iODBC module for FreeRADIUS server
freeradius-krb5	2.1.0-1ubalda2~hardy1	kerberos module for FreeRADIUS server
freeradius-ldap	2.1.0-1ubalda2~hardy1	LDAP module for FreeRADIUS server
freeradius-mysql	2.1.0-1ubalda2~hardy1	MySQL module for FreeRADIUS server
freeradius-postgresql	2.1.0-1ubalda2~hardy1	PostgreSQL module for FreeRADIUS server
freeradius-utils	2.1.0-1ubalda2~hardy1	FreeRadius client utilities
iptables	1.4.3.2-3-ubalda-1-2	iptables + IMQ + layer7 + IPP2P
ldap-utils	2.4.9-1ubalda	OpenLDAP utilities
libfreeradius2	2.1.0-1ubalda2~hardy1	FreeRADIUS shared library
libupnp-ubalda	1.3.1-1	libupnp compilado desde fuentes
linux-headers-2.6.29.1-ubalda4-l7-p2p-imq	2.6.29.1-ubalda4-l7-p2p-imq	Header files related to Linux kernel, specif
linux-image-2.6.29.1-ubalda4-l7-p2p-imq	2.6.29.1-ubalda4-l7-p2p-imq	Linux kernel binary image for version 2.6.29
upnpd-linuxigd	1.0-1	upnp daemon - Linux IGD
slapd	2.4.9-1ubalda	OpenLDAP server (slapd)

Para instalar estos paquetes se utilizará la orden

sudo dpkg –i *nombredelficherodelpaquete.deb*

B.1.3. Instalación del núcleo del sistema

Se incluye un fichero comprimido con todos los scripts en bash y Perl que hacen funcionar el sistema, así como las Interfaces Web, tanto de la página de login como la de administración. Para instalarlo ejecutar

sudo tar xfzv nucleo_sistema.tar.gz –C /

sudo apt-get install libnet-ldap-perl libtime-modules-perl libauthen-radius-perl
sudo mkdir /var/run/cansas

B.1.4. Configuración de los paquetes

B.1.4.1. Apache

Instalar apache con modulos necesarios:

sudo apt-get install apache2-utils apache2-suexec libexpat1 ssl-cert libapache2-mod-php5 php5 php5-common
sudo apt-get install apache2 libapache2-mod-perl2 libapache2-mod-php5 libapache-mod-ssl

========CONFIGURAR APACHE==========

a2enmod rewrite
#activa modulo rewrite

a2enmod ssl
#activa modulo ssl

a2enmod suexec
#activa modulo suexec para ejecutar como root los comandos de iptables

Editar fichero /etc/sudoers y añadir lo siguiente:

ubalda ALL=(ALL) NOPASSWD:/usr/local/bin/route-clear-up.sh, /usr/local/bin/route-user-down.sh, /usr/local/bin/updateconfig.sh, /usr/local/bin/purgeoldusers.pl

www-data ALL=NOPASSWD:/usr/local/bin/route-clear-up.sh, /usr/local/bin/route-user-down.sh, /usr/local/bin/updateconfig.sh, /usr/local/bin/purgeoldusers.pl, /usr/local/bin/test_flood.sh

```
freerad ALL=NOPASSWD:/usr/local/bin/userup.sh
```

Esto permite que dichos usuarios (ublda,www-data y freerad) ejecuten dichos comandos con privilegios de administrador (root). Sustituir ubalda por el nombre del usuario predeterminado del sistema

========GENERAR CERTIFICADOS SSL==========

1) SSL

https://help.ubuntu.com/7.10/server/C/httpd.html#https-configuration

genera clave privada para SSL

```
openssl genrsa -out /etc/ssl/private/server.key 1024
```

genera el certificado para firmar

```
openssl req -new -key /etc/ssl/private/server.key -out /etc/ssl/server.csr
```

Este archivo /etc/ssl/server.csr debe ser enviado a una autoridad CA para obtener un certificado oficial

```
openssl x509 -req -days 365 -in /etc/ssl/server.csr -signkey /etc/ssl/private/server.key -out /etc/ssl/certs/server.crt
```

si no queremos un certificado oficial podemos firmarlo nosotros mismos, asumiendo el consecuente aviso de seguridad por parte del navegador al acceder por https

```
echo "Listen 443" >> /etc/apache2/ports.conf
```

Configuracion de virtualhosts....

Se incluyen en el cdrom la configuración necesaria para los virtualhosts dentro del directorio apache, comprobar estos ficheros y compiarlos a /etc/apache2/sites-enabled. Después reiniciar el servidor apache

```
/etc/init.d/apache2 restart
```

B.1.4.2. Servidor DNS bind9

Instalación

```
sudo apt-get install bind9
```

añadir zonas en /etc/bind/named.conf.local

http://www.mexicoextremo.com.mx/content/view/423/62/
http://www.gpltarragona.org/archives/421/

Se incluye configuración de ejemplo en el cdrom

B.1.4.3. OpenLDAP

Instalar openldap y phpldapadmin

```
sudo apt-get install samba samba-doc smbldap-tools
sudo apt-get install ldap-utils phpldapadmin
sudo apt-get install php5-ldap  phpldapadmin php-net-ldap php-auth


sudo cp /usr/share/doc/samba-doc/examples/LDAP/samba.schema.gz /etc/ldap/schema/
sudo gzip -d /etc/ldap/schema/samba.schema.gz
```

generar password

```
sudo dpkg-reconfigure slapd
```

Importar base de datos (incluida en el cdrom)

```
slapadd -c -l fichero.ldif

sudo chown -R openldap.openldap /etc/ldap/slapd.d
```

Comprobar funcionamiento

```
slaptest -f slapd.conf -F slapd.d
```

apt-get install freeradius freeradius-ldap
apt-get install pptpd

B.1.4.4. FreeRADIUS

Se incluyen los paquetes compilados en el cdrom, no obstante, si alguien quiere compilarlo el mismo:

```
sudo bash
apt-get install libssl0.9.8 openssl
apt-get install apt-src
apt-src update
mkdir ~/build_freeradius
cd ~/build_freeradius
apt-src install freeradius
```

Añadir soporte TTLS y LDAP.

Editar ~/build_freeradius/freeradius-x/debian/rules y añadir

```
                --with-rlm_eap_tls \
        --with-rlm_eap_ttls \
        --with-rlm_eap_peap \
        --with-rlm_ldap \
```

y quitar el exit 1 del checkeo de links openssl

añadir en el apartado build-deps para freeradius-dev
libssl-dev, libpq-dev y libssl0.9.8 para el binario

```
~/build_freeradius/freeradius-x/debian/control
añadir nueva version y creditos en
~/build_freeradius/freeradius-x/debian/changelog
apt-get install libssl-dev libpq-dev
cd ~/build_freeradius
apt-src build freeradius

cd ..
dpkg -i *.deb
```

B.1.4.5. PoPToP

```
apt-get install radiusclient1
```

http://wiki.freeradius.org/PopTop (Forzarlo a usar mschap2 con encriptacion fuerte)

B.1.4.6. DHCPD

apt-get install dhcp3-server-ldap dhcp3-server

B.2. Instalación del sistema ya configurado

Resulta mucho más sencillo y efectivo instalar ya el sistema configurado y luego modificarlo a gusto del usuario.

1. Instalaremos ubuntu 8.04LTS en un disco duro.
http://ftp.udc.es/ubuntu-releases/8.04.3/ubuntu-8.04.3-server-i386.iso

2. Arrancamos el sistema recien instalado y copiamos en una memoria usb los ficheros:
/etc/fstab
/etc/udev/rules.d/70-persistent-net.rules
Copiamos en la memoria usb ademas el fichero
sistema_configurado.tar.gz del cdrom del proyecto

3. Arrancamos el ordenador con el cdrom de ubuntu y cuando nos pregunte por el particionado pulsamos ALT+F2 lo cual nos dará acceso a una shell interactiva
4. Nos hacemos administrador
sudo bash
5. montamos el sistema instalado anteriormente en un directorio temporal

```
mkdir /sistema
mount /dev/sdxy /sistema
```

donde x es a si es el primer discoduro, b si es el segundo…
e y es 1 si es la primera partición, 2 si es la segunda…

Montamos el disco usb

```
mkdir /disco
mount /dev/sdxy /disco
```

lo mismo de antes pero para el disco usb, si tecleas *dmesg* te saldra cual es el disco usb

Descomprimimos el sistema

```
tar xfzv /disco/sistema_configurado.tar.gz –C /sistema
```

Copiamos los ficheros

```
cp /disco/fstab /sistema/etc
cp /disco/70-persistent-net.rules /sistema/etc/udev/rules.d/70-persistent-net.rules
```

Revisamos el fichero /sistema/boot/grub/menu.lst para que tenga las particiones con el nombre correcto

Reiniciamos.

Nota: La interfaz de WAN (Internet) debe ser eth0 y la de LAN (clientes) eth1. Se pueden renombrar en el fichero /etc/udev/rules.d/70-persistent-net.rules

Bibliografía

[J1 CAR] CARBONE, Gunnison y S TODDARD, Duane. Open Source Enterprise Solutions: Developing an E-Business Strategy. Wiley, New Cork, 2001.
Cs01

[K1 RAY] RAYMOND, Eric. The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. O'Relly, Sebastopol, 2001.

[D2 PAV] PAVLICEK, Rusell. Embracing Insanity: Open Source Software Development. Sams, Indianapolis, 2000.

[K63 SAN] SANDRED, Jan. Managing open source projects a Wiley tech brief. New York : John Wiley & Sons, 2001.
San01

[D29 BOE] Boehm, Barry W. Software Engineering Economics, Englewood Cliffs (New Yersey) : Prentice-Hall, cop. 1981

[TRO 08] Thomas Rosén. Open Source Business Model: Balancing Customers and Community. Department of Management and Engineering. Linköping University, SE-581 83 Linköping. 2008
http://www.ep.liu.se/smash/get/diva2:18309/FULLTEXT01

[C2 VLA] Andrew A. Vladimirov, Konstantin V. Gavrilenko, Andrei A. Mikhailovsky. Hacking wireless seguridad de redes inalámbricas. Madrid : Anaya Multimedia. 2005

[LARTC] Bert Hubert. Thomas Graf, Greg Maxwell, Remco van Mook. Linux Advanced Routing & Traffic Control
http://www.gulic.org/comos/LARTC

[WPA 08] Martin Beck, Erik Tews. Practical attacks against WEP and WPA
http://dl.aircrack-ng.org/breakingwepandwpa.pdf