Servidores GNU/Linux

Estos materiales se licencian bajo la «Creative Commons Reconocimiento-CompartirIgual License España». Para ver una copia de esta licencia, se puede visitar http://creativecommons.org/licenses/by-sa/3.0/es/

Autores:

- Pablo Boronat Pérez (Universitat Jaume I)
- Miguel Pérez Francisco (Universitat Jaume I)
- David Rubert Viana (Universitat Jaume I)

Introducción

- GNU/Linux en Wikipedia
- Uso de Linux en los grandes servidores. (linux 80% de cuota total).
- Servidores de hosting más fiables (Junio 2011).

Los sistemas GNU/Linux son ideales para albergar cualquier tipo de servicio, en nuestro caso para todos aquellos servicios relacionados con el buen funcionamiento de la red (DNS, Web, Correo, etc.).

La principal característica de estos sistemas en un entorno de computadores servidor es por un lado la fiabilidad y rendimiento del sistema, y por otro lado la libre disposición de todo tipo de herramientas open-source. La práctica totalidad de servicios que montemos en estas computadores tienen disponibles las fuentes de las aplicaciones en repositorios abiertos, por lo que esto nos facilitará cualquier labor administrativa, como puede ser compilar un pequeño cambio o realizar una configuración del entorno a medida.

Qué distribución utilizar

- http://es.wikipedia.org/wiki/Distribución_linux

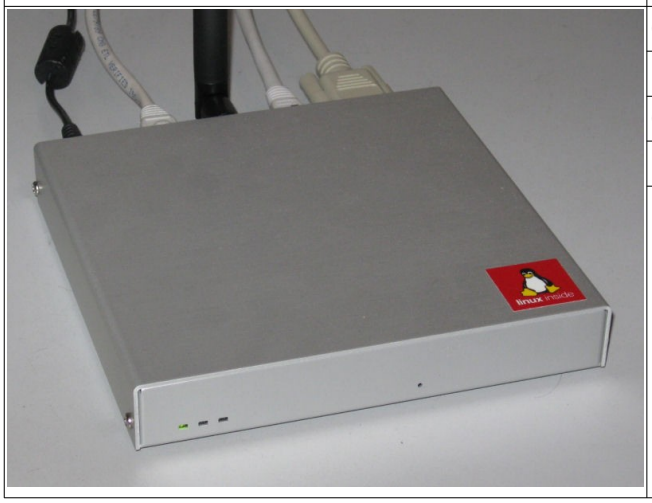
Una distribución de Linux nos proporciona un kernel de Linux junto con una serie de utilidades administrativas y un sistema de empaquetado de software (binario normalmente), personalizado en función del mercado al que vaya dirigido (escritorio, servidor, portátiles).

La distribución que deberemos utilizar es aquella con la que nos sentamos más cómodos. Dependiendo del hardware y de nuestra experiencia pueden ser más o menos recomendables unas distribuciones sobre otras.

- Debian. Distribución por excelencia para servidores.
- Ubuntu (Server edition).
- Fedora.
- Arch Linux.
- Gentoo.
- Optware + OpenWRT.

Hardware para una distribución UNIX

La potencia que nos puede proporcionar una distribución Linux, capaz de realizar cualquier tarea de propósito general, en conjunto con una hardware adecuado (bajo consumo, bajo coste) optimizará al máximo la solución a nuestras necesidades. En las imágenes, ponemos como ejemplo hardware de Alix, RaidSonic, Synology o Qnap por tener algunas referencias de hardware específico pensado para funcionar con una distribución Linux, aunque hay que recordar que podemos montar nuestro servidor sobre cualquier PC genérico.

Alix	Características (Aproximadas)
	**Consumo**: 5W
	RAM: 128 o 256MB
	CPU: 433 o 500Mhz
	USB, Ethernet, MiniPCI
	Bajo precio, bajo consumo, rendimiento moderado

http://pcengines.ch/alix.htm

RaidSonic ICY BOX 4220	Características (Aproximadas)
	**Consumo**: 25W
	RAM: 256MB
	CPU: 400Mhz
	USB, Ethernet
	Bajo precio, bajo consumo, rendimiento adecuado

http://nas-4220.org

Synology 411	Características (Aproximadas)
	**Consumo**: 60W
	RAM: 1GB
	CPU: 1,8Gz dual core
	USB, Ethernet gigabit
	Alto precio, bajo consumo, alto rendimiento

http://www.synology.com/mx/products/DS411+II/index.php

QNAP 459	Características (Aproximadas)
	**Consumo**: 35W
	RAM: 1GB
	CPU: Atom dual core
	USB, Ethernet gigabit
	Alto precio, bajo consumo, alto rendimiento

http://www.qnap.com/pro_detail_feature.asp?p_id=144

Webmin

- http://www.webmin.com
- http://es.wikipedia.org/wiki/Webmin

Webmin es una herramienta de apoyo a la administración/configuración de diferentes servicios del sistema (o incluso el mismo sistema) vía web, facilitándonos el proceso de puesta en marcha de cualquier utilidad gracias a un entorno pensado para enmascarar los detalles más técnicos (diferentes archivos de configuración, diferentes sintaxis, etc). Además nos permite asignar privilegios específicos a usuarios a los que queremos delegar la responsabilidad sobre determinados apartados de nuestro servidor.

Configuración

La práctica totalidad de configuración de Webmin se realiza vía web, pero hay algunos apartados que no hay más remedio que configurarlos a través de archivos de configuración. Veamos los principales archivos que deberemos modificar para poner en marcha Webmin en nuestro sistema.

miniserv.conf

En este archivo de configuración definiremos, a través de directivas, el funcionamiento del servicio web a través del cual funcionará **Webmin**. Veamos las más importantes:

```
port=10000
```

```
allow=127.0.0.1 192.168.1.0/24
```

La autenticación de usuarios de Webmin se realiza a través de un archivo de contraseñas, definido mediante la siguiente directiva en miniserv.conf:

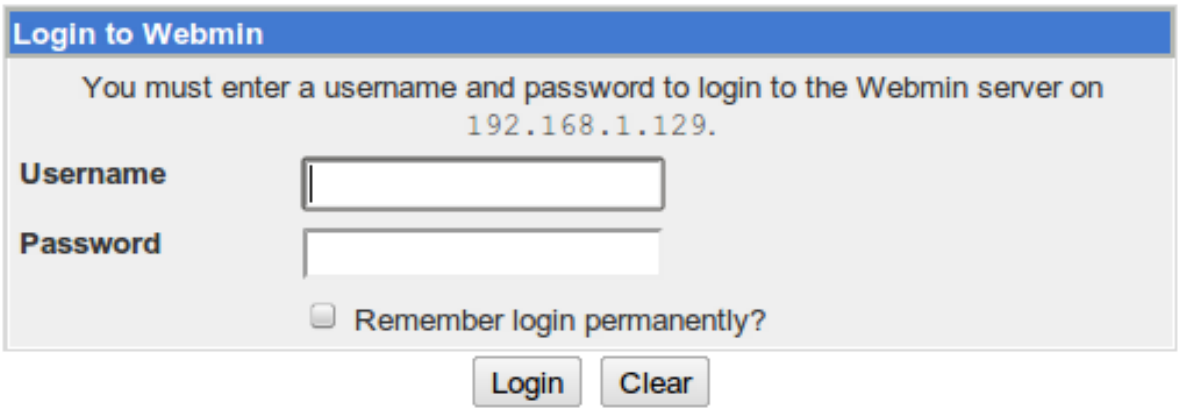
```
userfile=/etc/webmin/miniserv.users
```

Podemos cambiar la contraseña o el usuario de administración de Webmin desde línea de comandos utilizando un script llamado **changepass.pl**.

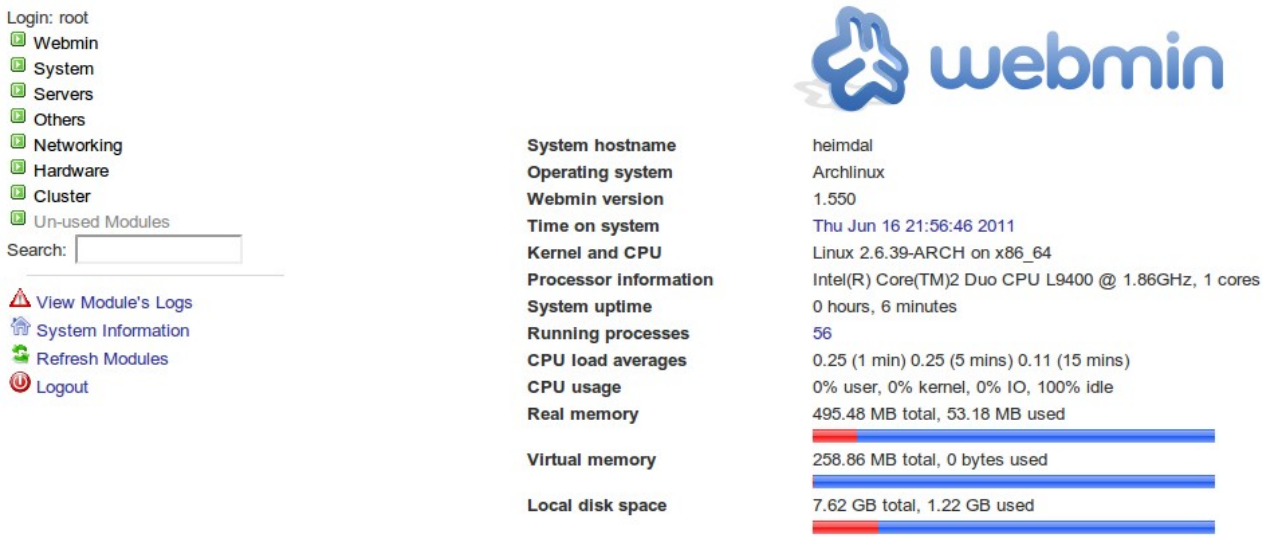
```
/usr/libexec/webmin/changepass.pl /etc/webmin admin foo
```

Capturas de pantalla de Webmin

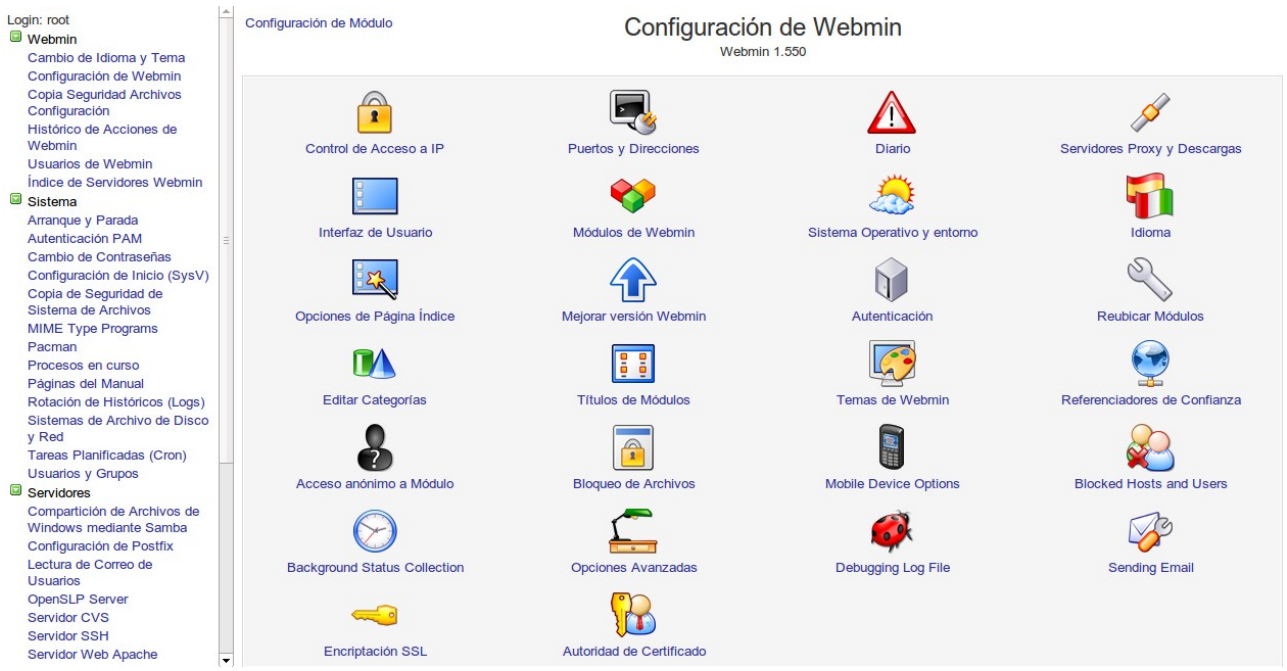
Login:



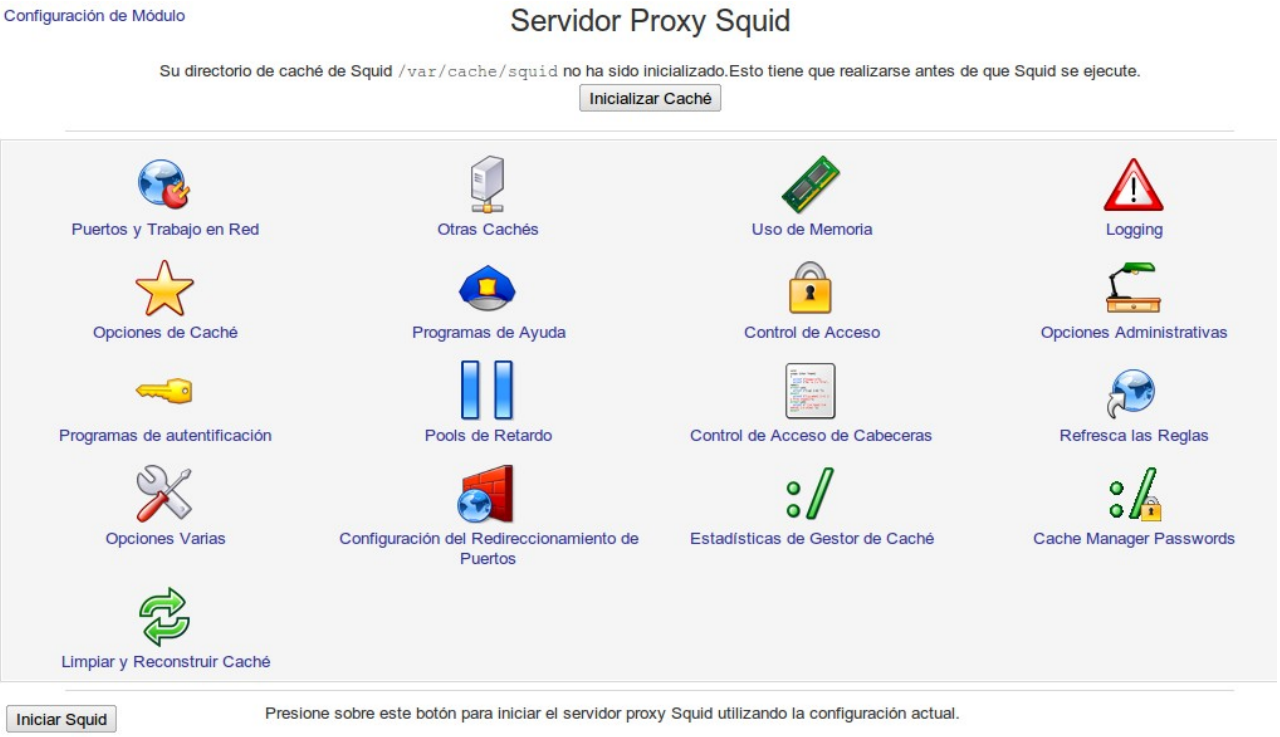
Página principal:



Configuración de Webmin:

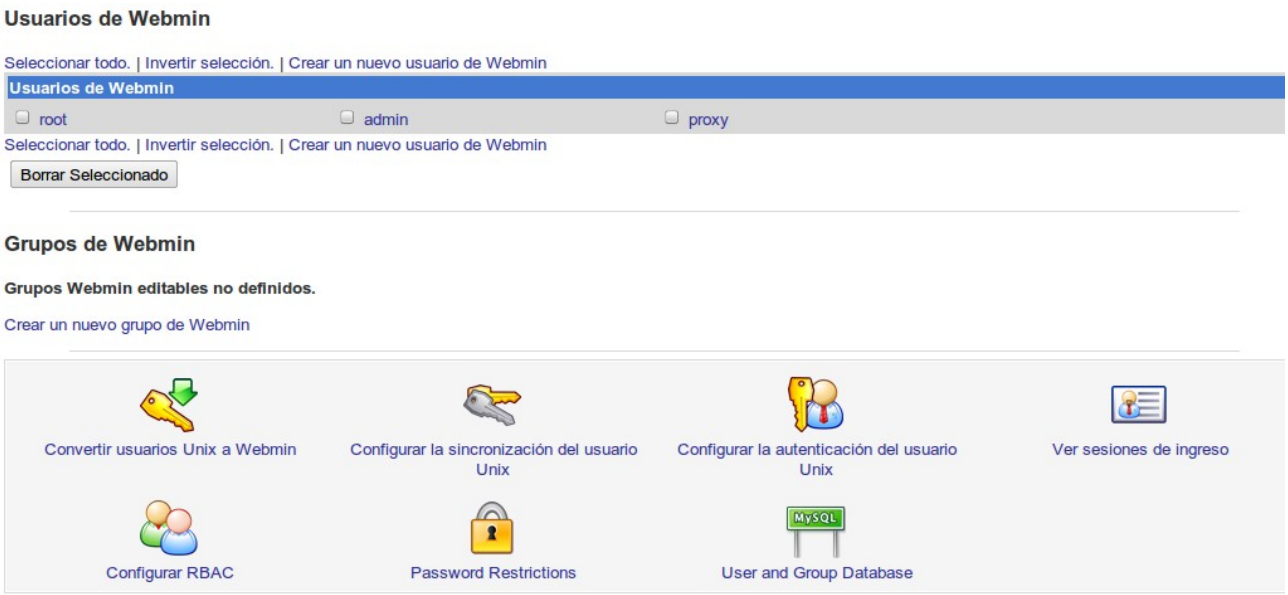


Configuración del proxy Squid:



Control de acceso

Una de las ventajas de Webmin, además de unificar la administración del sistema en un interfaz web, es que nos permite de manera sencilla delegar determinadas labores de administración a usuarios concretos. Por ejemplo, podríamos crear un usuario encargado únicamente de la gestión del proxy, o del servidor web, del DNS, etc. Esto se realiza desde el apartado de gestión de usuarios de Webmin, que a su vez nos permite definir el tipo de rol que queremos asignarle al usuario.



Módulos adicionales

En la distribucion de **Webmin** vienen incluidos una gran cantidad de módulos para gestionar servicios a través de Webmin. Podéis ver una lista completa de módulos “standard” aquí:

http://www.webmin.com/standard.html

Si no encontramos el servicio/herramienta que queremos gestionar vía **Webmin** en los módulos standard, podemos buscarlo en la lista de módulos desarrollados por terceros:

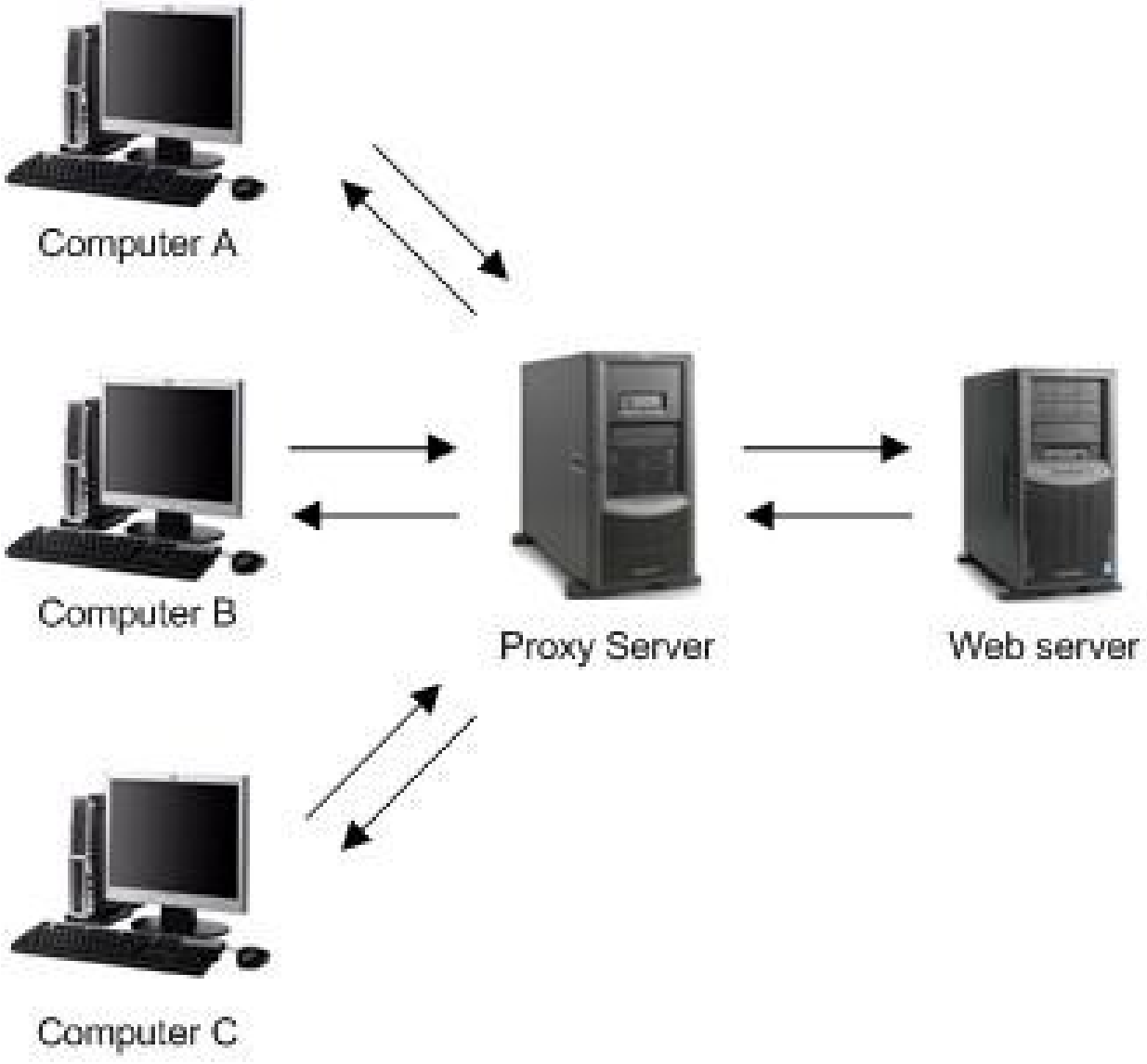
http://www.webmin.com/third.html

Ejercicio 6.1 (Sólo presencial):

Proxy web Squid

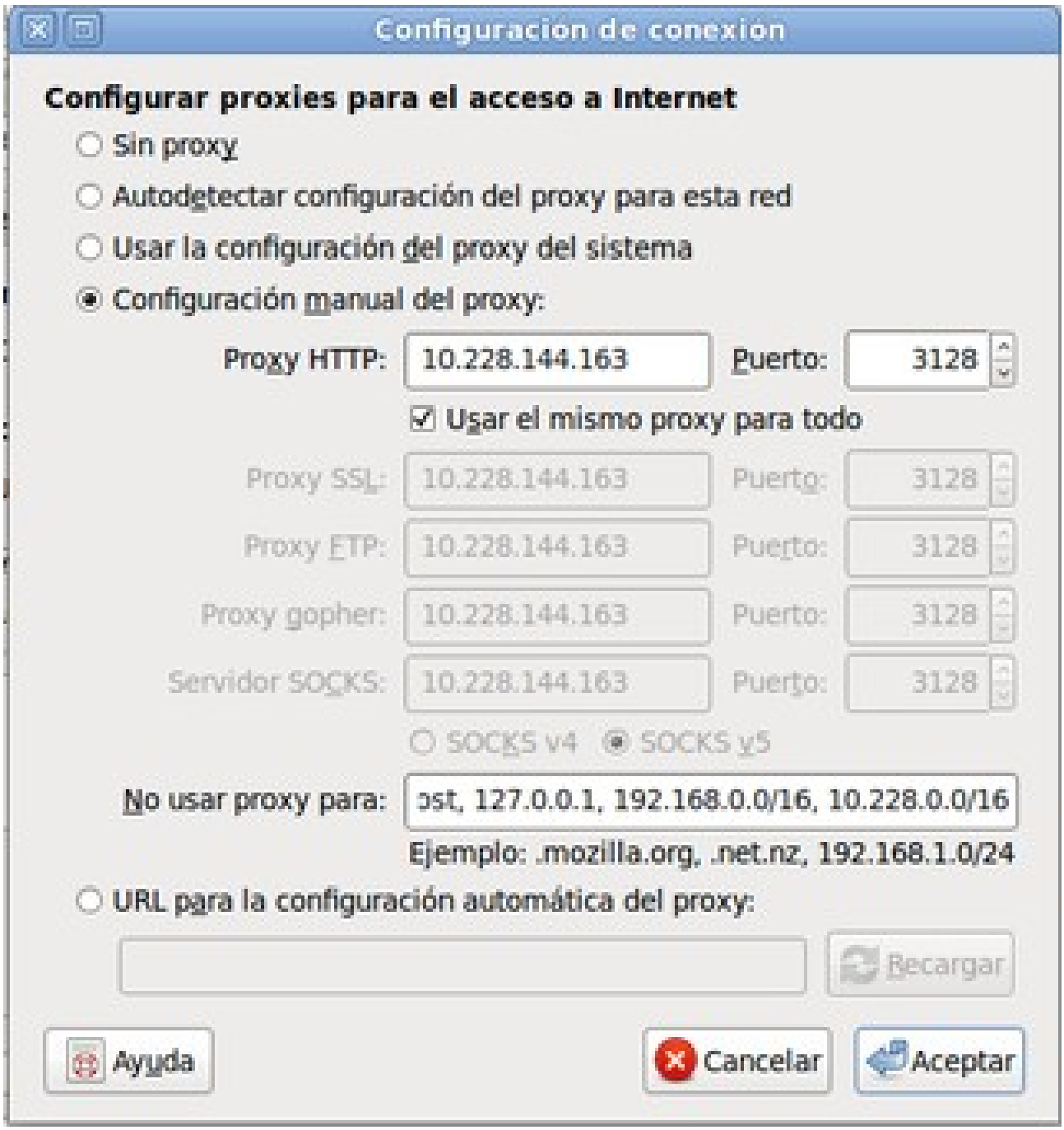
- http://www.squid-cache.org/
- http://es.wikipedia.org/wiki/Squid_(programa)

Squid es un servidor proxy basado en código abierto. La misión de un servidor proxy es hacer de intermediario entre una petición de cliente y un servidor remoto, almacenando (cacheando) la información por si nuevas peticiones de clientes pudieran reaprovecharla. Tuvieron mucho auge hace unos años ya que gracias a ellos se conseguía ahorrar mucho ancho de banda, sobre todo con contenidos estáticos y públicos. Hoy en día que el ancho de banda sobra y los contenidos son muy personalizados para el cliente no tienen tanto sentido, excepto en aquellos sitios en los que se quiere tener una red privada controlada donde todas las máquinas conectan a internet (principalmente la web) a través de un único punto de salida, en este caso el servidor proxy.



Los clientes de un servidor proxy

Cualquier navegador moderno tiene su apartado de configuración donde permite acceder a internet a través de proxy. Veamos como ejemplo la configuración necesaria en Firefox para acceder a un proxy.



Una de las ventajas de un servidor proxy es que él mismo nos realizará las peticiones de resolución de nombres, por lo que en el ordenador cliente no hará falta que tenga ni siquiera configurado un DNS.

Configuración: squid.conf

Toda la configuración de squid se centraliza en un archivo: **squid.conf**. El archivo en sí no es complejo pero sí muy extenso, aunque sólo deberemos configurar aquellas preferencias que cambien el comportamiento por defecto. Disponemos de una versión documentada de este archivo donde se nos explicará cada una de las directivas y su sintáxis, además de una página web de referencia donde encontraremos ejemplos de uso de cada una de ellas: config. Veamos algunas de las opciones más importantes con las que nos encontraremos.

Puerto de escucha de Squid

Por defecto, squid escuchará en el puerto 3128. Esta directiva es la que configura esta característica del servidor:

```
http_port 10.228.144.163:3128
```

Directorio de caché de archivos

```
# ufs: Formato de almacenamiento de caché
# /var/cache/squid: Directorio del sistema donde almacenar la
cache
# 100: Número de megabytes máximos a utilizar en la caché
# 16 256: Número de directorios de primer y segundo nivel aa crear
(mejor no cambiar).
cache_dir ufs /var/cache/squid 100 16 256
```

Limitando el acceso al proxy a una subred específica

```
acl ip_acl src 192.168.1.0/24
http_access allow ip_acl
http_access deny all
```

Control de acceso

Los permisos y restricciones de acceso a nuestro proxy Squid las realizamos a través de ACL's (*Access Control List*). Podemos definir diferentes tipos de ACL's, a las que posteriormente les podremos aplicar, por orden, prioridades de acceso.

Tenemos ACL's para controlar, principalmente, acceso a los puertos y acceso por subredes.

Hoy en día la navegación web es el principal exponente de Internet, y justamente en un proxy son los puertos de la web los que van a tener más éxito entre nuestros usuarios. Si nuestra política en el proxy va a ser cerrar todos los puertos menos los que queremos abrir (lo más recomendable) utilizaremos la siguiente configuración.

```
acl SSL_ports port 443
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443         # https
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl CONNECT method CONNECT

# Deny requests to certain unsafe ports
http_access deny !Safe_ports
```

```
# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports
```

Los puertos SSL necesitan una configuración específica, ya que la conexión se debe hacer a través de un túnel utilizando los comandos CONNECT que nos proporciona el protocolo HTTP.

El log de squid

- http://wiki.squid-cache.org/SquidFaq/SquidLogs

Toda la información relativa al funcionamiento y uso de nuestro servidor proxy irá a parar a los archivos de log de squid, veamos en detalle su sintáxis y cómo tratarlos para extraer la información que nos interesa.

logformat

Define la sintaxis con la que queremos almacenar el log en el archivo. Lo mejor es utilizar las definiciones que nos vienen por defecto, ya que así podremos utilizar programas de parseo directamente, sin tener que informarle de nuestras modificaciones.

```
logformat squid %ts.%03tu %6tr %>a %Ss/%03>Hs %<st %rm %ru %un %Sh/%<A %mt
```

access_log

Define dónde se ubica el archivo de log de accesos a Squid, donde se almacenarán los accesos de nuestros clientes al proxy. Toma como parámetro el archivo en nuestro sistema de ficheros, y el tipo de formato de log que queremos aplicarle.

```
access_log /var/log/squid/access.log squid
```

cache_log

Define la localización del archivo de log principal de squid, donde se guardará información general sobre el funcionamiento del proxy.

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

Utilidades de mantenimiento

Generador de estadísticas

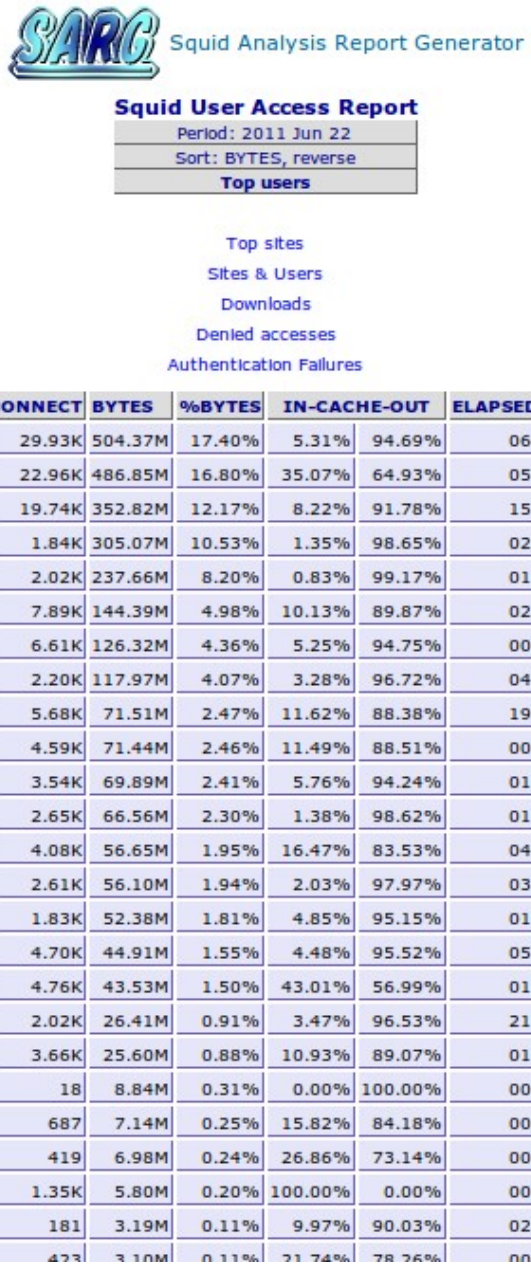
El log de Squid contiene mucha información interesante que puede ayudarnos a conocer mejor cuál es el uso del proxy que se hace día día por nuestros usuarios. **Sarg** es un programa que se encargará de parsear el log de squid y generar una serie de reports diarios, semanales, o mensuales.

- http://sarg.sourceforge.net/

Squid Analysis Report Generator

Squid User Access Report

FILE/PERIOD	CREATION DATE	USERS	BYTES	AVERAGE
2011Jun22-2011Jun22	Thu Jun 23 00:01:57 2011	29	2.89G	99.94M
2011Jun21-2011Jun21	Wed Jun 22 00:01:54 2011	27	2.38G	88.25M
2011Jun20-2011Jun20	Tue Jun 21 00:01:50 2011	23	3.28G	142.69M
2011Jun19-2011Jun19	Mon Jun 20 00:01:48 2011	33	4.25G	128.97M
2011Jun18-2011Jun18	Sun Jun 19 00:01:42 2011	27	3.56G	131.99M
2011Jun17-2011Jun17	Sat Jun 18 00:01:42 2011	21	4.30G	205.08M
2011Jun16-2011Jun16	Fri Jun 17 00:01:41 2011	30	2.41G	80.60M
2011Jun15-2011Jun15	Thu Jun 16 00:01:34 2011	29	1.92G	66.30M
2011Jun14-2011Jun14	Wed Jun 15 00:01:34 2011	21	2.49G	118.87M
2011Jun13-2011Jun13	Tue Jun 14 00:01:31 2011	23	2.37G	103.18M
2011Jun12-2011Jun12	Mon Jun 13 00:01:29 2011	25	4.51G	180.58M
2011Jun11-2011Jun11	Sun Jun 12 00:01:26 2011	25	2.72G	109.02M
2011Jun10-2011Jun10	Sat Jun 11 00:01:24 2011	25	2.74G	109.95M
2011Jun09-2011Jun09	Fri Jun 10 00:01:22 2011	21	3.47G	165.62M
2011Jun08-2011Jun08	Thu Jun 9 00:01:19 2011	21	913.89M	43.51M
2011Jun07-2011Jun07	Wed Jun 8 00:01:17 2011	20	33.30M	1.66M
2011Jun06-2011Jun06	Tue Jun 7 00:01:19 2011	25	5.97G	238.87M
2011Jun05-2011Jun05	Mon Jun 6 00:01:18 2011	28	6.18G	220.73M
2011Jun04-2011Jun04	Sun Jun 5 00:01:15 2011	26	4.29G	165.23M
2011Jun03-2011Jun03	Sat Jun 4 00:01:11 2011	22	4.79G	218.08M
2011Jun02-2011Jun02	Fri Jun 3 00:01:09 2011	27	3.32G	123.23M
2011Jun01-2011Jun01	Thu Jun 2 00:01:07 2011	28	4.61G	164.76M
2011May31-2011May31	Wed Jun 1 00:01:42 2011	26	3.84G	147.99M
2011May30-2011May30	Tue May 31 00:01:42 2011	23	4.59G	199.96M
2011May29-2011May29	Mon May 30 00:01:40 2011	23	4.11G	178.85M
2011May28-2011May28	Sun May 29 00:01:34 2011	23	2.37G	103.40M



Squid Analysis Report Generator

Squid User Access Report

Period: 2011 Jun 22

Sort: BYTES, reverse

Top users

Top sites

Sites & Users

Downloads

Denied accesses

Authentication Failures

NUM		USERID	CONNECT	BYTES	%BYTES	IN-CACHE-OUT		ELAPSED TIME	MILLISEC	%TIME
1			29.93K	504.37M	17.40%	5.31%	94.69%	06:41:22	24.082.895	6.43%
2			22.96K	486.85M	16.80%	35.07%	64.93%	05:40:35	20.435.050	5.45%
3			19.74K	352.82M	12.17%	8.22%	91.78%	15:05:17	54.317.107	14.49%
4			1.84K	305.07M	10.53%	1.35%	98.65%	02:25:16	8.716.012	2.33%
5			2.02K	237.66M	8.20%	0.83%	99.17%	01:09:08	4.148.101	1.11%
6			7.89K	144.39M	4.98%	10.13%	89.87%	02:33:41	9.221.449	2.46%
7			6.61K	126.32M	4.36%	5.25%	94.75%	00:52:11	3.131.930	0.84%
8			2.20K	117.97M	4.07%	3.28%	96.72%	04:19:21	15.561.861	4.15%
9			5.68K	71.51M	2.47%	11.62%	88.38%	19:04:19	68.659.924	18.32%
10			4.59K	71.44M	2.46%	11.49%	88.51%	00:22:26	1.346.894	0.36%
11			3.54K	69.89M	2.41%	5.76%	94.24%	01:11:49	4.309.449	1.15%
12			2.65K	66.56M	2.30%	1.38%	98.62%	01:19:51	4.791.142	1.28%
13			4.08K	56.65M	1.95%	16.47%	83.53%	04:45:36	17.136.141	4.57%
14			2.61K	56.10M	1.94%	2.03%	97.97%	03:47:44	13.664.668	3.65%
15			1.83K	52.38M	1.81%	4.85%	95.15%	01:35:31	5.731.416	1.53%
16			4.70K	44.91M	1.55%	4.48%	95.52%	05:39:03	20.343.195	5.43%
17			4.76K	43.53M	1.50%	43.01%	56.99%	01:03:03	3.783.836	1.01%
18			2.02K	26.41M	0.91%	3.47%	96.53%	21:06:21	75.981.109	20.27%
19			3.66K	25.60M	0.88%	10.93%	89.07%	01:45:25	6.325.998	1.69%
20			18	8.84M	0.31%	0.00%	100.00%	00:45:52	2.752.570	0.73%
21			687	7.14M	0.25%	15.82%	84.18%	00:14:56	896.782	0.24%
22			419	6.98M	0.24%	26.86%	73.14%	00:06:12	372.176	0.10%
23			1.35K	5.80M	0.20%	100.00%	0.00%	00:00:00	170	0.00%
24			181	3.19M	0.11%	9.97%	90.03%	02:00:05	7.205.020	1.92%
25			423	3.10M	0.11%	21.74%	78.26%	00:06:17	377.864	0.10%

Proxies en guifi.net

Dentro de guifi.net, justamente cobra sentido un servidor proxy, donde los clientes que conectan a través de él no tienen conexión directa a internet, por lo que pueden utilizarlo como pasarela centralizada a Internet. Con una única conexión a Internet podemos dar acceso a mucha gente, teniendo un control absoluto de lo que se permite hacer.

Una de las implementaciones más exitosas dentro de la red guifi.net es la llamada “Federación de proxies”. La idea consiste en dar de alta un proxy en la web de guifi.net, y permitir conectar a dicho proxy a aquellos usuarios de guifi.net que nosotros queramos. ¿Cómo hacerlo?

Primero que nada, deberemos decidir a qué grupo de usuarios queremos permitir el acceso a Internet a través de nuestro proxy. Tenemos 2 opciones:

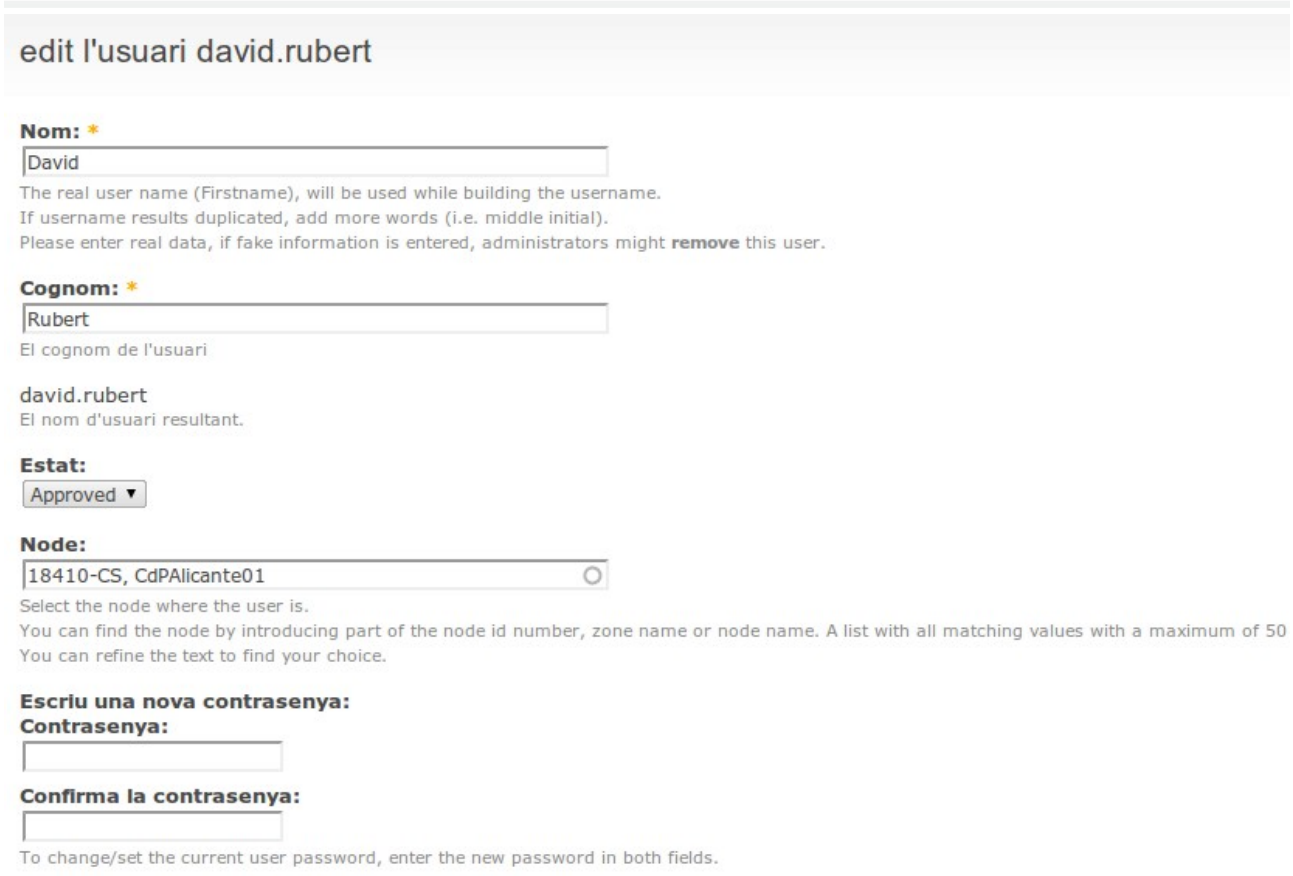
- Permitir a cualquier usuario que se ha dado de alta en guifi.net conectar

a través de nuestro proxy con su usuario/contraseña.
- Permitir únicamente a los usuarios que nosotros damos de alta conectar con su usuario/contraseña.

Siguiendo una serie de pasos en la configuración del Squid, y descargando periódicamente una base de datos de usuarios/contraseña que se han dado de alta en la web, podemos crear un sistema controlado de acceso a nuestro proxy, evitando la anonimidad de los accesos.

Los usuarios de los proxies federados pueden darse de alta ellos mismos en la web de guifi.net, siguiendo estos pasos:

- Primero que nada, deberán tener dado de alta un nodo con una antena cliente operativa.
- En la página de su nodo, hay una sección de “usuarios”. Ahí pueden dar de alta un nuevo usuario, que será de la forma “nombre.apellidos”. El usuario pasará al estado “Pendiente de revisión”.
- Alguien de guifi.net revisa constantemente esa lista de usuarios, y aprobará aquellos usuarios que están en la cola de pendientes.
- Con ese usuario/contraseña, ya pueden entrar a cualquier proxy federado de guifi.net.



Configuración principal de Squid para integrarse con los proxies federados

Esta sería la configuración principal de un servidor **Squid** para hacerlo funcionar contra el archivo de usuarios/contraseñas de guifi.net.

```
acl SSL_ports port 443
acl Safe_ports port 80          # http
acl Safe_ports port 443         # https
```

```
acl CONNECT method CONNECT

# Autenticacio
auth_param basic program /usr/libexec/ncsa_auth
/etc/squid/guifinet_passwd
acl usuaris proxy_auth REQUIRED

# Blocked domains
acl blockeddomain dstdomain "/etc/squid/blocked.domains.acl"

# Access priority
http_access deny CONNECT !SSL_ports
http_access deny !Safe_ports
http_access deny blockeddomain
http_access allow usuaris
http_access deny all
```

Script de actualización de los usuarios de los proxies federados de guifi.net

Si tenemos montado un proxy federado y queremos que la lista de autorizados se actualice en función de la información que se va generando en la web, deberemos utilizar un script que obtenga la lista de usuarios autorizados en nuestro proxy, la guarde, y reinicie squid. Este script es conveniente ejecutarlo a intervalos regulares de tiempo (por ejemplo, cada 30min.) en un cron.

```
#!/bin/bash

PROXY_ID=31803
TMP=$(mktemp)

/bin/wget --timeout=240
http://www.guifi.net/es/node/$PROXY_ID/view/federated -qO $TMP
/bin/touch /etc/squid/guifinet_passwd
OK=`/bin/cat $TMP|wc -l`
NEW=`/usr/bin/diff /etc/squid/guifinet_passwd $TMP|wc -l`
if [ $OK != "0" ]; then
  if [ $NEW != "0" ]; then
    /bin/cp $TMP /etc/squid/guifinet_passwd
    /etc/init.d/squid reload
  fi;
fi;
rm -f $TMP
```

Ejercicio 6.2http://guifi.net

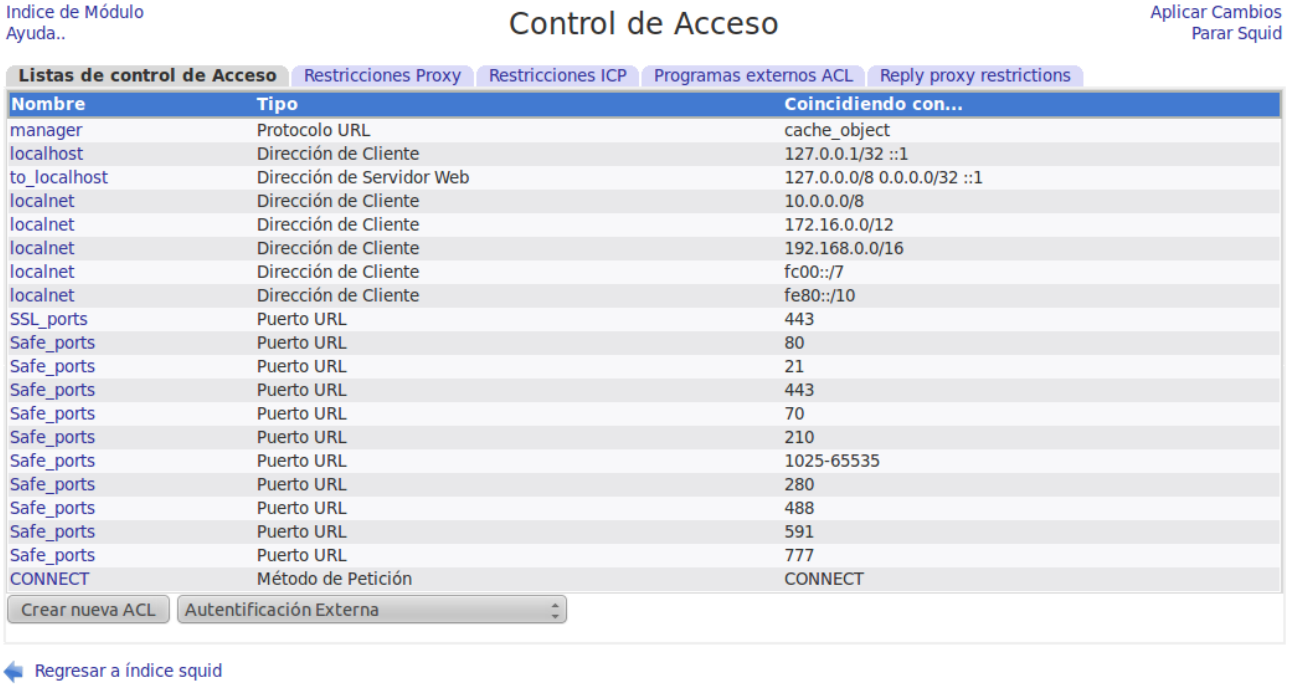
Squid con Webmin

La herramienta de configuración de sistema vía web **Webmin** de la que hemos hablado antes nos permite administrar de una manera cómoda todas las funciones principales de **Squid**. Veamos las partes de configuración web más

importantes.

Control de acceso

Podemos definir los tipos de control de acceso que queremos aplicar, y el orden en el que los queremos aplicar para que unos tengan prioridad sobre otros.



Usuarios y autenticación básica

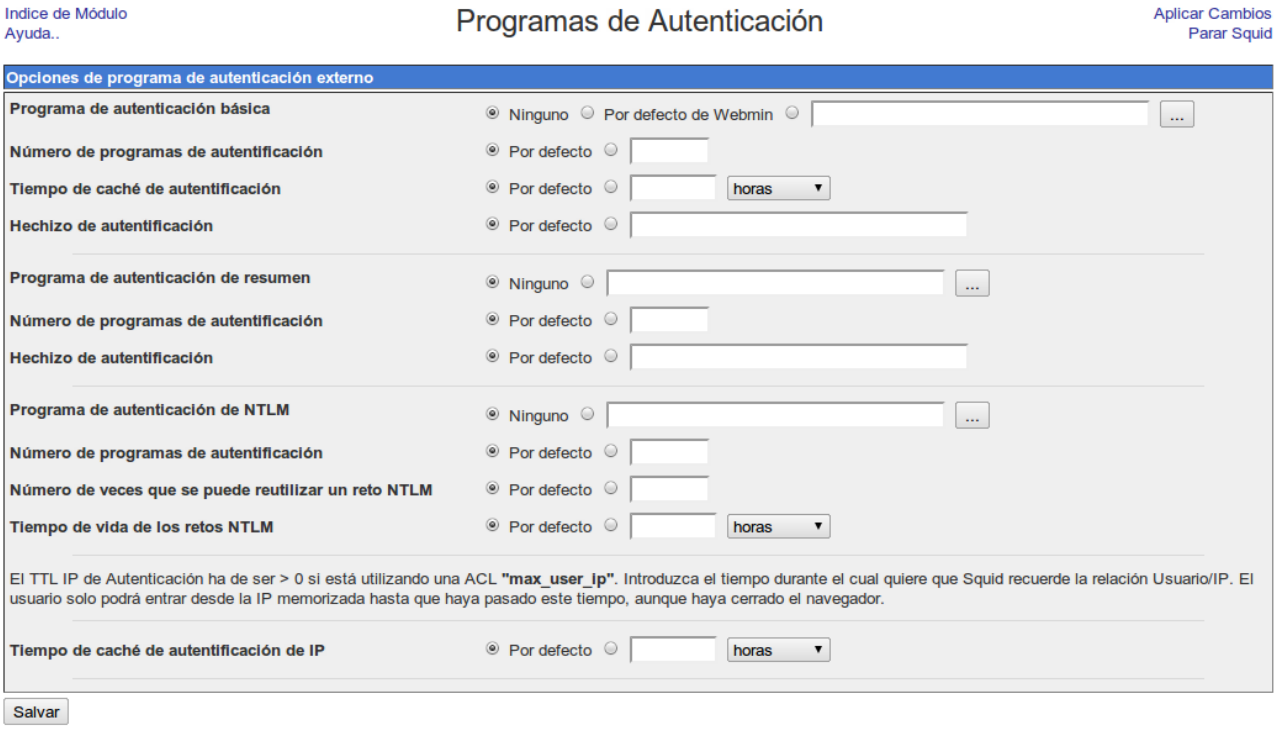
Si queremos tener una base de datos de usuario/contraseña autorizados para acceder al proxy, y no queremos hacer un montaje de proxies federados, podemos realizar la siguiente configuración desde Webmin.

1. Añadimos un “Programa de autentificación” del tipo “Autenticación básica” , “Por defecto de Webmin”. Es decir, vamos a utilizar autenticación básica en el navegador autenticando contra un archivo de usuarios del estilo de Apache.

2. Añadiremos los usuarios en la nueva sección de “Usuarios” que nos aparece.

3. En el control de acceso, añadimos una nueva ACL del tipo “Autentificación Externa” llamada “autenticados”.

4. Le damos prioridad a esta ACL, de manera que si el usuario se autentica se le permita conectarse al proxy.



Indice de Módulo

Crear ACL

Autentificación Externa ACL

Nombre ACL	autenticados
Usuarios Externos Autorizados	(•) Todos los usuarios ( ) Sólo los listados..
URL de Fallo	
Almacenar ACL en archivo	(•) Configuración Squid ( ) Separate file ...
	[ ] ¿Usar sólo contenidos existentes del archivo?

Salvar

Regresar a Lista ACL | Regresar a Índice

Listas de control de Acceso | **Restricciones Proxy** | Restricciones ICP | Programas externos ACL | Reply proxy restrictions

Añadir restricción proxy

	Acción	ACLs
[ ]	Permitir	manager localhost
[ ]	Denegar	manager
[ ]	Denegar	!Safe_ports
[ ]	Denegar	CONNECT !SSL_ports
[ ]	Denegar	all
[ ]	Permitir	localnet
[ ]	Permitir	localhost

Añadir restricción proxy

Delete Selected Restrictions

Regresar a índice squid

Indice de Módulo

Crear Restricción de Proxy

Restricción de Proxy

Acción	(•) Permitir ( ) Denegar	
Coincidir con ACLs	all manager localhost to_localhost localnet SSL_ports Safe_ports CONNECT autenticados	**No coincidir con ACLs** all manager localhost to_localhost localnet SSL_ports Safe_ports CONNECT autenticados

Salvar

Regresar a Lista de ACL | Regresar a Índice

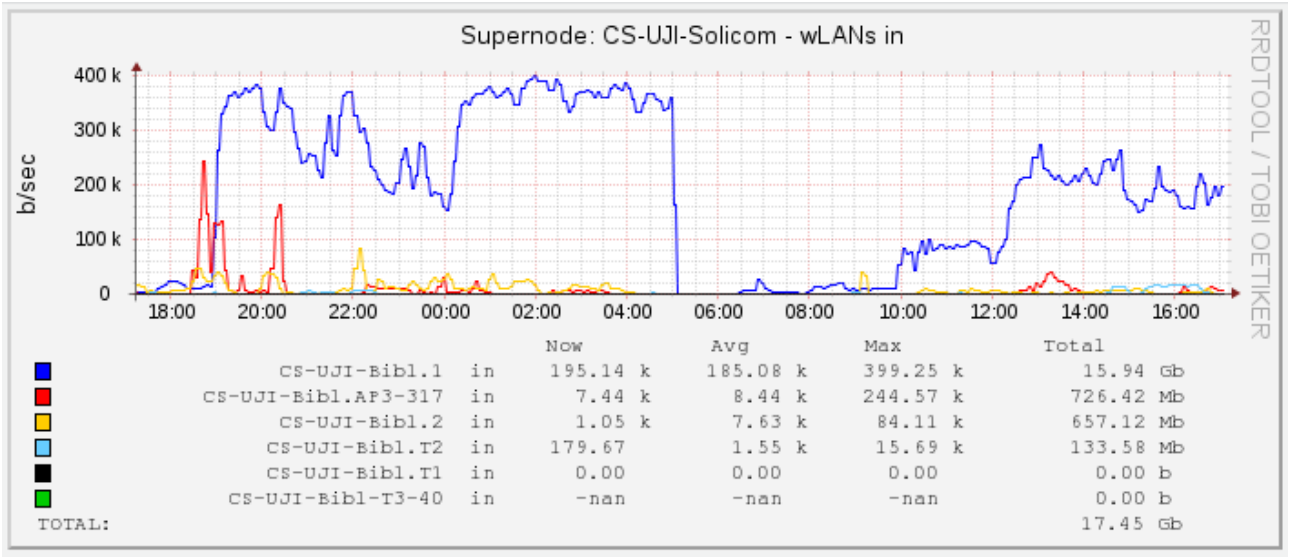
Ejercicio 6.3 (presencial)

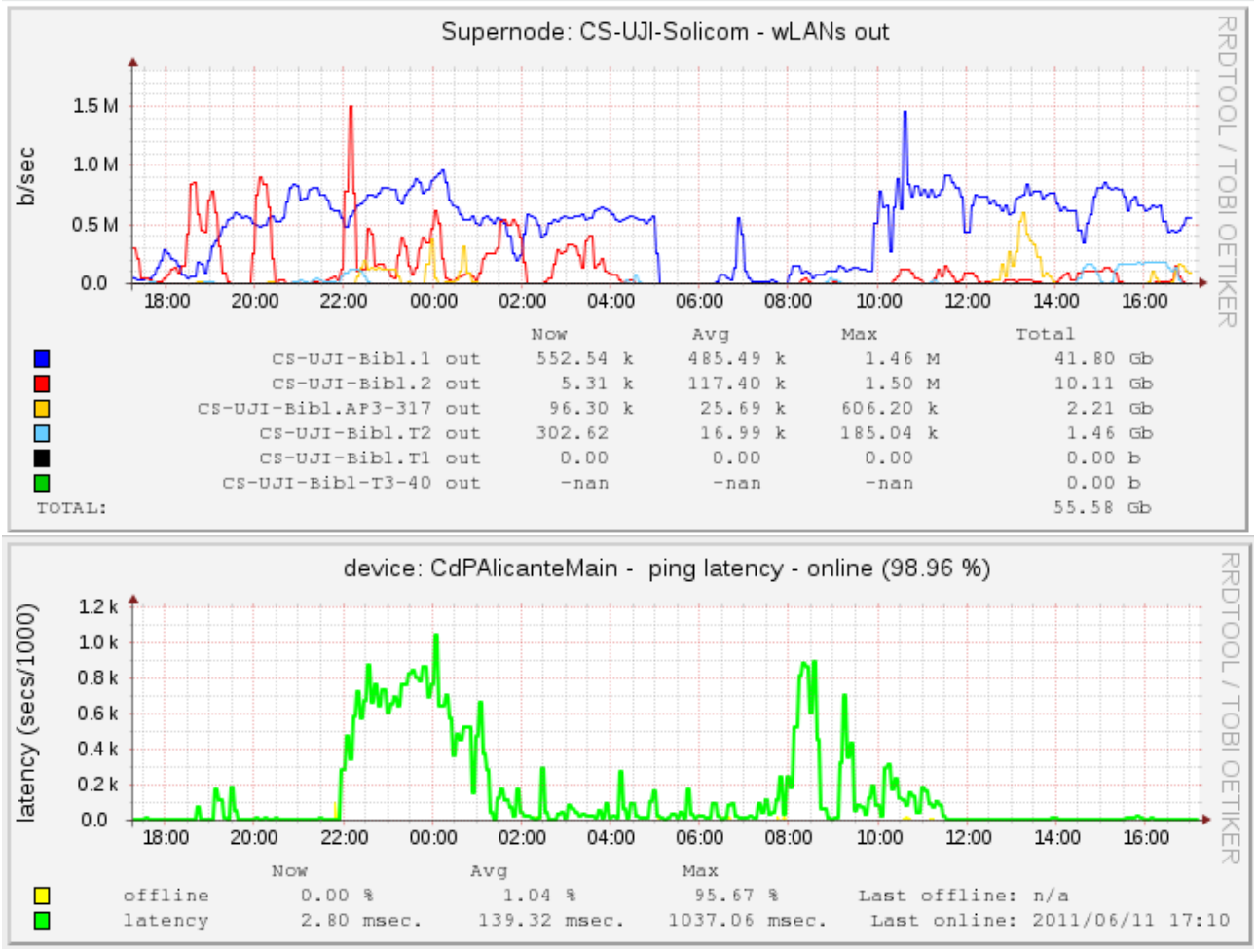
`http://150.128.49.223/squid-reports`

Servidor de gráficas

- http://es.wiki.guifi.net/wiki/Servidor_de_gráficas
- https://gitorious.org/guifi/snpservices

Una de las herramientas más útiles de las que disponemos en la web de guifi.net, es la de poder ver en cada momento el estado de un supernodo gracias a unas gráficas donde se representa el tráfico transferido en cada uno de los interfaces, así como otras gráficas con la latencia del ping. Ejemplos:





Estas gráficas nos proporcionan un histórico a 24h, 1 semana, 1 mes o 1 año del estado de nuestro nodo.

Pero estas gráficas no se generan directamente desde la web de guifi.net. ¿Cómo es posible que desde Internet podamos ver las gráficas que monitorizan diferentes nodos/supernodos, incluso aquellos ubicados en zonas sobre las que no podemos acceder? Veamos cómo funciona.

Para entender cómo llega a funcionar este servicio, tenemos que explicar un poco la infraestructura utilizada. Partimos de las siguientes premisas:

- No es un servicio centralizado, sino distribuido. Hay repartidos decenas de servidores de gráficas por toda la geografía.
- Cada servidor de gráficas se encarga de una o varias zonas. Para definir de qué zona se encarga cada servidor deberemos tener acceso a la edición de zonas (Castelló de la Plana, Almassora, Atzeneta, etc.)
- La máquina que vaya a hacer de servidor de gráficas es muy recomendable que tenga doble pata Internet-guifi.net, de manera que a medida que monitoriza y guarda información sobre los nodos de su zona publicará dicha información en forma de gráficas por la pata de Internet.
- Una vez montado el servicio, todos los nuevos nodos que se añadan en las zonas que monitoriza el servidor de gráficas se procesarán automáticamente, sin intervención por nuestra parte.

Por tanto, partimos de una máquina con conectividad guifi.net, dispuesta a monitorizar una serie de nodos conectados a su red, y que además tiene

conectividad con Internet, donde publicará las gráficas de cada uno de estos nodos que está monitorizando. ¿Qué mejor sistema que un GNU/Linux para llevarlo a cabo? Veamos las herramientas que necesitaremos para llevar a buen puerto el montaje de este servicio.

Ejercicio 6.4

- http://guifi.net/uji
- http://guifi.net/ca/node/35901

snpservices. Dependencias

La herramienta desarrollada desde guifi.net que consigue el propósito que hemos comentado se llama **snpservices**. Esta herramienta tiene dependencia con varios proyectos de software libre que seguro que os suenan y que pasamos a detallar.

Apache

- http://es.wikipedia.org/wiki/Servidor_HTTP_Apache
- http://httpd.apache.org/

Apache es el servidor web de código abierto más utilizado del mundo.

PHP

- http://es.wikipedia.org/wiki/Php
- http://php.net/

PHP es el lenguaje de scripting orientado a la web más utilizado del mundo.

RRDtool

- http://es.wikipedia.org/wiki/RRDtool
- http://oss.oetiker.ch/rrdtool/

Round Robin Database Tool. Herramienta de registro y creación de gráficas para datos procesables en series temporales.

MRTG

- http://es.wikipedia.org/wiki/MRTG
- http://oss.oetiker.ch/mrtg/

Herramienta de supervisión y monitorización de hardware de red (típicamente routers), totalmente dependiente de RRDtool.

Aunque no tenemos que conocer a fondo cada uno de estos programas, sí que

tenemos que saber qué misión cumplen en el conjunto para llegar a dar el servicio de **snpservices**.

SNMP

- http://es.wikipedia.org/wiki/SNMP

Es el protocolo utilizado para intercambiar información de estado entre dispositivos de comunicación. Es perfecto para monitorizar tráfico, estado y disponibilidad de diferentes interfaces de red de un dispositivo. Funciona a través del puerto UDP/161, y aunque la estructura de datos intercambiada es compleja, únicamente deberemos saber cómo habilitarlo en un dispositivo, ya que la faena sucia de consultar, tratar y almacenar los datos la hará por nosotros el mrtg.

La única configuración que deberemos realizar en el servicio SNMP es definir la comunidad, en este caso la standard para permitir el acceso en modo lectura a los datos: **public**.

Instalando y configurando snpservices

Podemos obtener **snpservices** directamente desde el repositorio donde se realiza el desarrollo, o empaquetado para nuestra distribución si utilizamos Debian/Ubuntu.

Repositorio GIT del desarrollo principal:

- https://gitorious.org/guifi/snpservices

Repositorio GIT del desarrollo del script de empaquetado .deb:

- https://www.gitorious.org/guifi/snpservices-debian

Si no queremos complicarnos la vida, podemos añadir directamente un repositorio donde encontramos los binarios empaquetados directamente para Debian/Ubuntu:

```
# vi /etc/apt/sources.list
deb http://repo.vic.guifi.net/debian/  ./
# apt-get update
# apt-get install snp-services
```

En este artículo del wiki oficial de guifi.net nos lo explican:

- http://es.wiki.guifi.net/wiki/Servidor_de_gráficas

Apache

Como sabéis, la configuración de Apache se puede complicar extremadamente. En este caso vamos a presuponer que queremos la configuración más sencilla posible, donde tenemos un servidor Apache corriendo en un servidor, y el módulo de PHP5 funcionando en él.

Para habilitar la publicación de las gráficas desde Apache deberemos permitir la ejecución de PHP en el directorio /usr/share/snpservices, habilitándola en la

URL /snpservices/ de nuestro servidor. Esta sería la directiva que necesitaríamos en el archivo **apache.conf**:

```
Alias /snpservices/ /usr/share/snpservices/
```

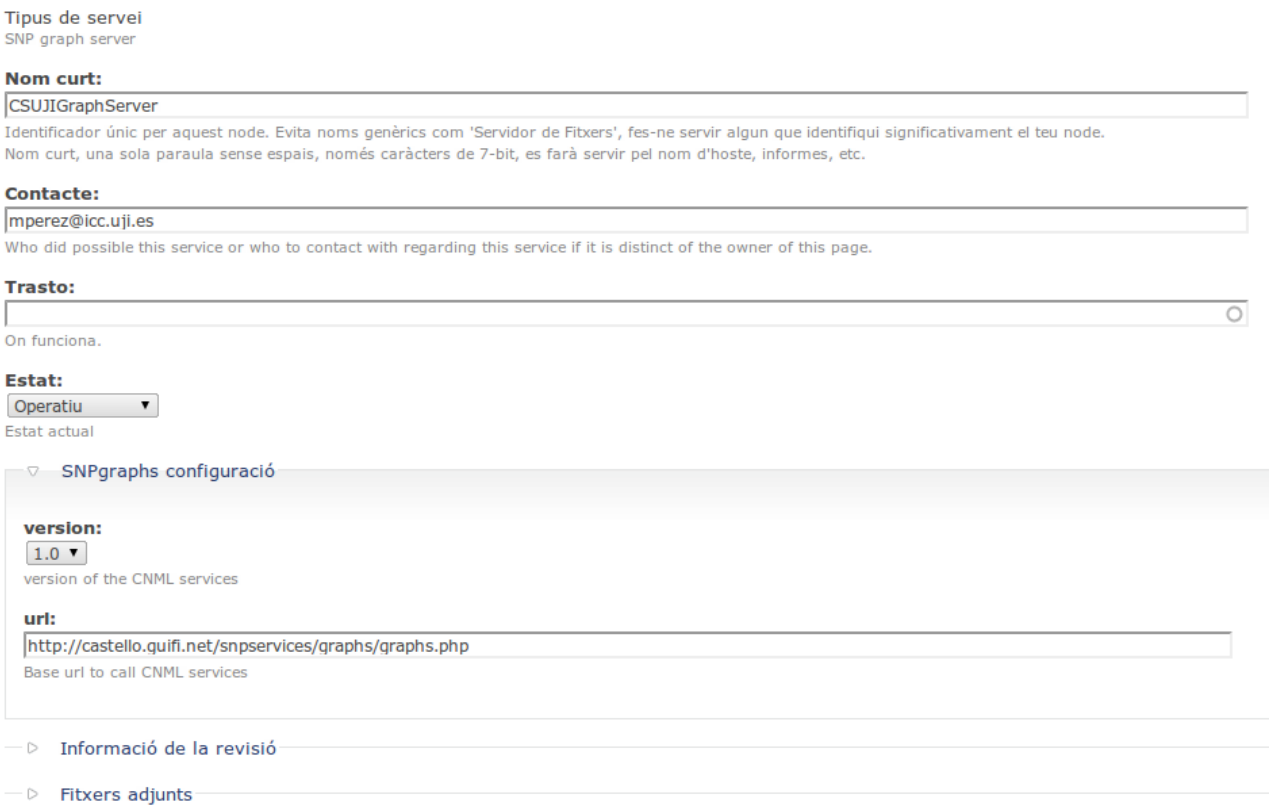
Una vez hecho esto, ya tendremos operativa la URL necesaria para la publicación de gráficas, que tendrá esta pinta:

```
http://direccion-ip-del-servidor/snpservices/graph/graph.php
```

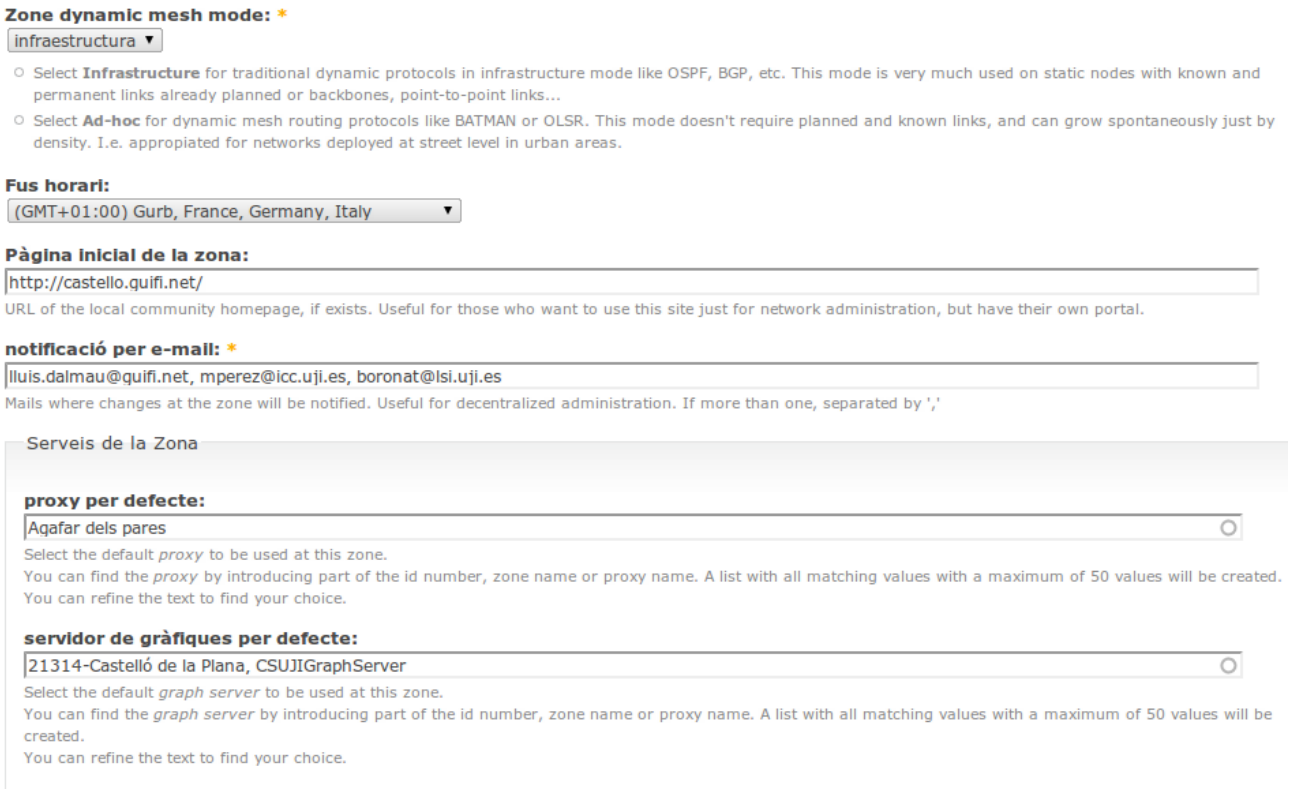
Dar de alta el servicio de gráficas en guifi.net

Antes de continuar con la instalación/configuración de snpservices, veamos cómo dar de alta el servicio en guifi.net. Este paso es muy importante, ya que, como sabéis, la integración de la información de la web con la configuración del sistema es vital para que funcionen las cosas en guifi.net. Más si cabe en este servicio.

Al dar de alta el nuevo servicio de gráficas, nos va a pedir una información clave, que es la URL donde publica las gráficas este servidor. Tenemos que hacer coincidir esta información con la URL que hemos dado de alta previamente en **Apache**. Además de esto, nos pedirá que asociemos el servicio a un servidor que previamente debemos haber registrado y asignado una IP pública.



Posteriormente, deberemos indicarle a la zona guifi.net que este nuevo servidor de gráficas será el que se encargará de proporcionar las graficas. Para ello, editaremos la zona y rellenaremos el campo de “servidor de gràfiques per defecte”.



Ejercicio 6.5 “”

- http://guifi.net/castello

- ¿Cuál es la URL donde publica las gráficas el servidor de gráficas de la UJI?
- ¿Qué servidor de gráficas se encarga de la zona de Albocàsser?

RRDTOOL & MRTG

Como ya hemos comentado son unas herramientas escritas en perl, cuya misión es:

- RRDTOOL: Almacena datos de cualquier tipo en series progresivas de tiempo y los representa en gráficas.
- MRTG: Consulta dispositivos de red, y obtiene y procesa los datos necesarios para almacenarlos en la base de datos de RRDTOOL.

Las herramientas son bastante complejas de configurar. Es por esto que el paquete snpservices automatizará la creación de los archivos de configuración por nosotros, además de actualizar constantemente la información sobre los trastos a monitorizar.

Configuración de SNPSERVICES

Con el ID de servicio de SNP Graph Server que hemos obtenido registrando nuestro servicio previamente en guifi.net, editamos el siguiente archivo para introducir 2 variables:

```
# vi /etc/snpservices/config.php
$SNPGraphServerId = 35207; // ID del nodo de nuestro servicio
$rootZone = 18688; // ID del nodo de la zona raiz
```

¿Qué nos falta por configurar? Pues poca cosa más. Expliquemos un poco por encima qué es lo que haría el montaje.

Hay un cron ubicado en /etc/cron.d/snpservices, que realiza 2 misiones:

- Cada 30 minutos se conecta a la web de guifi.net, y descarga todos los nuevos nodos que hay que monitorizar para las zonas que están asignadas a nuestro servidor de gráficas. Generará el archivo de configuración necesario para MRTG (está ubicado en /var/lib/snpservices/data/mrtg.cfg).
- Cada 5 minutos, ejecutará los procesos de monitorización de perl MRTG que se encargarán de ir a consultar cada uno de los cacharros montados en nuestra zona por SNMP, y con la información obtenida guardarla para generar posteriormente las gráficas. Veremos constantemente los siguientes procesos en el sistema:

```
malva-serv:/usr/share/snpservices/data# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME
COMMAND
root         1  0.0  0.0   2032   532 ?        Ss   Jun23   0:15
init [2]
root         2  0.0  0.0      0     0 ?        S    Jun23   0:00
[kthreadd]
...
root     32275  1.0  1.6  25760 17060 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32276  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32277  0.0  0.0   1772   560 ?        S    18:55
0:00 /bin/ping -c5 -i 0.2 10.228.178.17 -q
root     32279  0.0  0.0   1772   556 ?        S    18:55
0:00 /bin/ping -c5 -i 0.2 10.228.178.68 -q
root     32280  0.0  1.6  25760 17060 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32283  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32284  0.0  1.6  25760 17060 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32288  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32290  0.0  1.5  25760 16520 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
```

```
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32292  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32293  0.0  0.1   2620  1056 ?        S    18:55
0:00 /bin/sh /usr/share/snpservices/common/ping.sh 10.228.135.4
root     32295  0.0  0.0   1772   560 ?        S    18:55
0:00 /bin/ping -c5 -i 0.2 10.228.135.4 -q
root     32296  1.0  1.6  25760 17060 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32299  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32301  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32302  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32303  0.0  1.6  25760 17024 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32304  3.0  1.6  25760 17096 ?        S    18:55
0:00 /usr/bin/perl -w /usr/bin/mrtg
/var/lib/snpservices/data/mrtg.cfg --lock-file /var/lock/mrtg
root     32331  0.0  0.1   2620  1056 ?        S    18:55
0:00 /bin/sh /usr/share/snpservices/common/ping.sh 10.228.135.49
root     32332  0.0  0.0   1772   556 ?        S    18:55
0:00 /bin/ping -c5 -i 0.2 10.228.135.49 -q
root     32450  0.0  0.1   2620  1056 ?        S    18:55
0:00 /bin/sh /usr/share/snpservices/common/ping.sh 10.228.166.7
root     32454  0.0  0.0   1772   556 ?        S    18:55
0:00 /bin/ping -c5 -i 0.2 10.228.166.7 -q
```

Habilitar SNMP en nuestro dispositivo

Dependiendo del tipo de dispositivo que queramos monitorizar activaremos el demonio SNMP de la manera que nos proporcione el sistema operativo. Por ejemplo, para un router Mikrotik, tenemos la siguiente utilidad:

```
[admin@Castalia] /snmp> print
       enabled: yes
       contact: david.rubert@gmail.com
      location: Castalia
     engine-id:
   trap-target: 0.0.0.0
trap-community: public
  trap-version: 1
```

Consulta del estado del servicio SNMP

Para comprobar el correcto funcionamiento del SNMP en aquellos routers que queramos monitorizar tenemos la herramienta **snmpwalk**. Con dicha herramienta, podemos lanzar una consulta puntual SNMP de diferentes valores del router, asegurándonos así que nuestro servidor de gráficas puede monitorizarlo. Por ejemplo:

```
snmpwalk -c public -v1 10.228.144.161
```

Este comando nos muestra muchísima información relativa a nuestro nodo en un momento del tiempo determinado. Si un proceso se encarga de consultar esa información a intervalos regulares, podría conseguir el muestreo necesario para generar un histórico de información. Esto es lo que hace mrtg por nosotros.

Ejercicio 6.6

```
# ip route add 10.0.0.0/8 via 150.128.49.223
```

Resolución de problemas típicos

Una vez tenemos nuestro servidor de gráficas funcionando, es posible que nos encontremos con casos en que las gráficas funcionan para unos supernodos pero que no funcionen bien para otros (gráficas vacías). Esto es debido a que las consultas que realiza MRTG sobre los dispositivos depende totalmente de cómo se ha dado de alta el router en la web de guifi.net.

Por ejemplo, si damos de alta un router del tipo DD-WRT, entonces MRTG irá a consultarle por el interfaz “vlan1”, mientras que si damos de alta un router mikrotik con tarjetas inalámbricas, irá a consultarle por los interfaces “wlan1”, “wlan2”, etc.

DNS. Dnsmasq y Powerdns

- http://es.wikipedia.org/wiki/Dns
- http://www.thekelleys.org.uk/dnsmasq/doc.html
- http://www.powerdns.com/

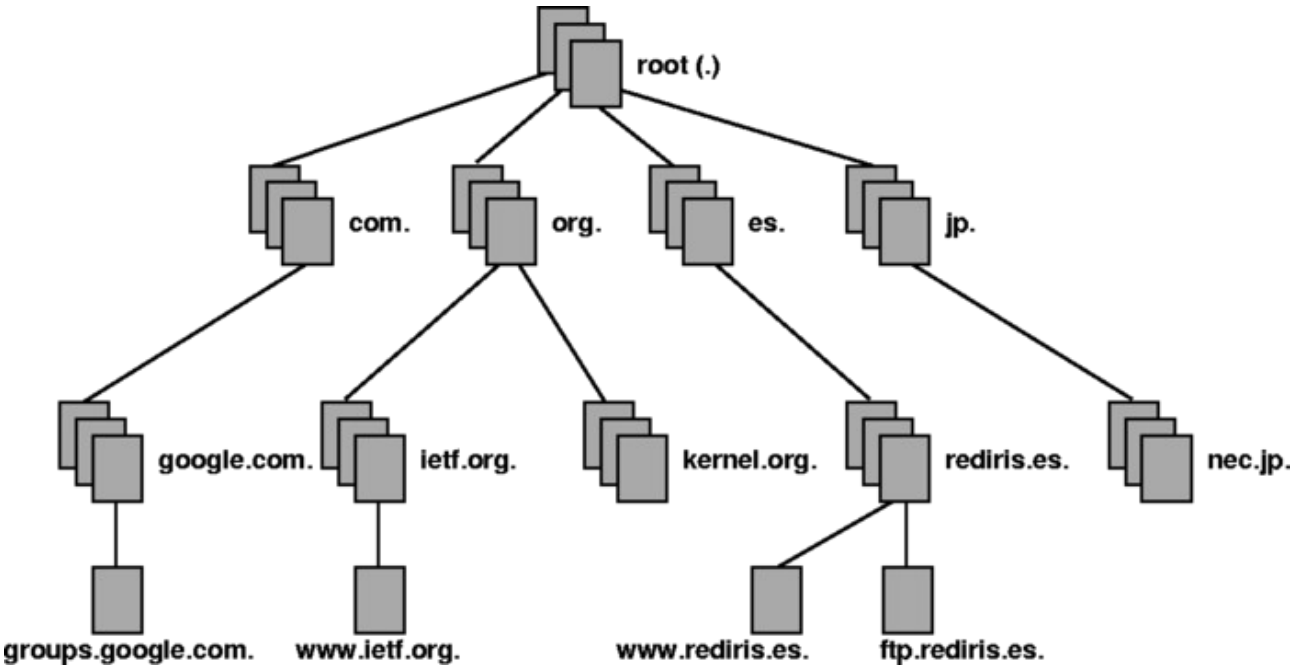
El DNS es uno de los servicios fundamentales en Internet, que pese a su simpleza conceptual mantiene una complejidad técnica de montaje importante, que se acrecenta más todavía cuando queremos utilizarlo sobre guifi.net.

El concepto del DNS es, como hemos comentado, muy simple. Se encarga de traducir los nombres de internet a direcciones IP, o las direcciones IP en nombres de Internet. Esta funcionalidad es requerida por casi cualquier aplicación de hoy en día.

Arquitectura DNS

La arquitectura que se define en el DNS es jerárquica y distribuida. Dado que sería imposible mantener una base de datos con todos los dominios y sus zonas centralizadas en una única máquina a nivel mundial, se van delegando

las zonas a diferentes servidores DNS autorizados, en función del nivel jerárquico en el que nos encontremos. Veámoslo en un ejemplo:



Además de esto, el propio RFC del DNS define un sistema de caché que implementan todos los servidores principales, donde se define un tiempo de vida máximo para cada registro consultado, de manera que si un servidor realiza una consulta remota, la guarda durante el tiempo máximo que le indica el servidor autorizado para esa zona.

Los dominios de primer nivel (TLDs) los gestiona la ICANN, así como los servidores raiz. Además, determina las empresas registradoras a las que se les permite gestionar las compras de los dominios de segundo nivel hechas por cualquiera de nosotros (Godaddy.com, nic.es).

El servicio de DNS funciona en el puerto 53 y el protocolo que utiliza para las consultas es el UDP.

Tipos de servidores DNS

Por tanto, podemos definir 4 o 5 comportamientos totalmente independientes en un servidor DNS:

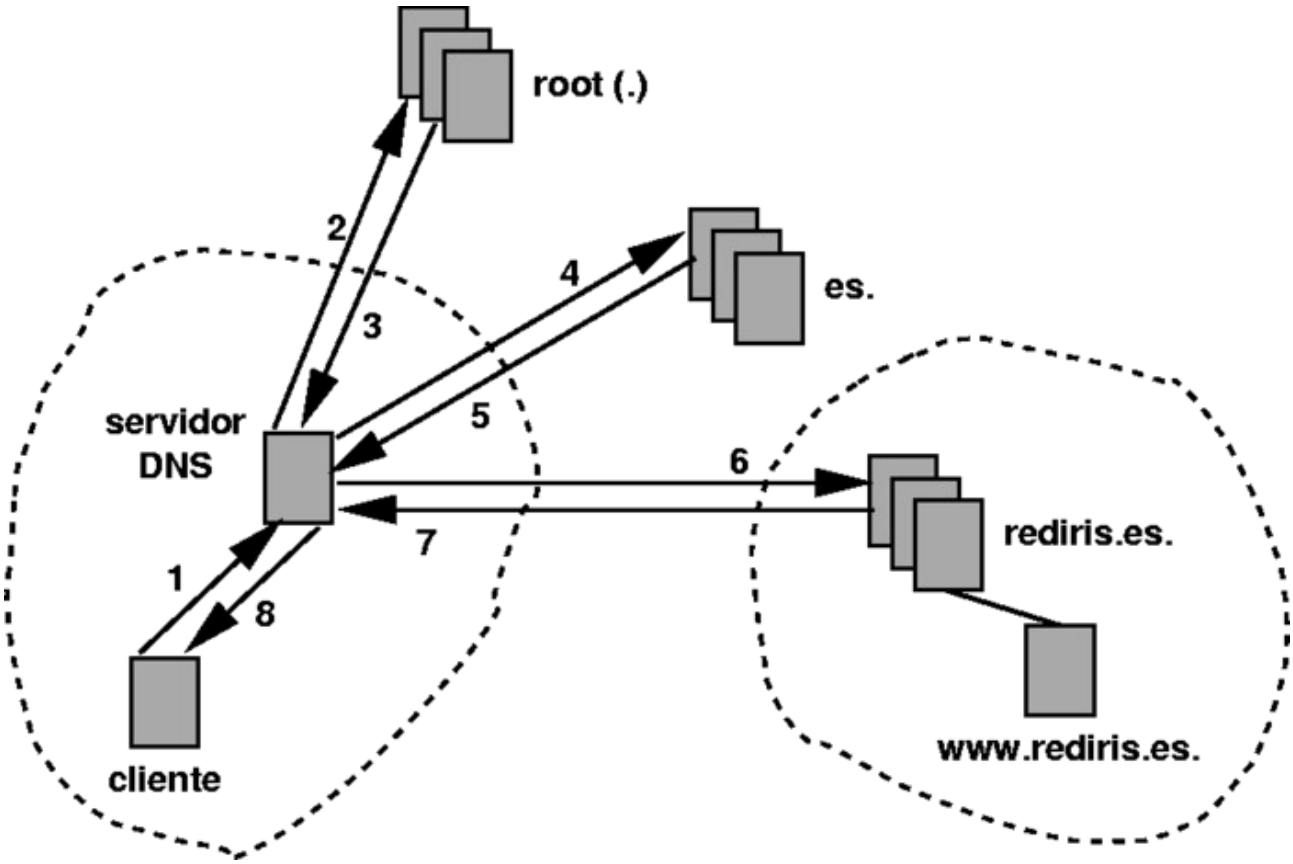
- **DNS forwarder**. Es el tipo de servidor DNS más simple. Únicamente redirige las peticiones hacia otro servidor DNS más completo.
- **DNS recursor**. Servidor recursivo. Es un servidor DNS que acepta consultas de clientes sobre todo tipo de registros (p.e. www.google.es). Si no tiene la información, irá a preguntarla a la infraestructura DNS, y posteriormente cacheará el registro el tiempo autorizado para tener la respuesta preparada ante las siguientes consultas.
- **Cache DNS**. Utilizado en conjunto con el servidor recursivo. Almacena todas las consultas de los clientes para tener la respuesta preparada para posteriores solicitudes de resolución.
- **Primary DNS**. Servidor maestro de su zona. Es un servidor DNS que tiene la autoridad para decidir a qué IP's resuelve cada uno de los registros de su zona. Puede gestionar una o múltiples zonas.

- **Secondary DNS**. Es un servidor que replica la zona de un servidor primario. Se encarga de tener actualizada la zona principal con los últimos cambios que se han producido.

Ejemplo teórico de petición de cliente DNS

Cuando un cliente Windows/Linux pregunta a su servidor DNS cuál es la IP de www.rediris.es, se desencadena un proceso similar al siguiente:

- El DNS que tiene configurado el cliente mira en su caché a ver si tiene el registro y todavía no ha explirado, si lo tiene, responde de inmediato.
- Si no tiene el registro, se va a preguntar a los servidores ROOT-DNS quién gestiona la zona de primer nivel .es.
- El root DNS responde con la IP del servidor que gestiona esos datos, y nuestro DNS va a preguntarle al nuevo servidor por rediris.es.
- El servidor DNS de primer nivel (.es) responde con la IP del servidor DNS de segundo nivel que gestiona el dominio rediris.es.
- Este servidor DNS del dominio de segundo nivel (rediris.es) es el maestro de la zona, así que podrá responder con el registro que solicita nuestro DNS, que a su vez responde a su cliente y cachea el registro para posteriores ocasiones.



Implementaciones: DNSMasq

DNSMasq es un servidor DNS (y DHCP) muy simple en su concepto. Es un DNS forwarder (no es DNS por sí mismo sino que depende de otros) y es que con una configuración mínima nos va a permitir tener operativo un servidor DNS

capaz de resolver cualquier consulta de resolución de nombres.

Instalación y configuración

Veamos cómo instalarlo para una distribución Debian/Ubuntu, no tiene ningún misterio:

```
# apt-get install dnsmasq
# /etc/init.d/dnsmasq restart
```

El servidor de nombres al que irá a preguntar será el que tengamos definido en el archivo /etc/resolv.conf, por ejemplo:

```
nameserver 10.228.130.162
```

Podemos parametrizar diferentes aspectos del funcionamiento de dnsmasq a través del archivo de configuración /etc/dnsmasq.conf. Os dejo que le echéis un vistazo si queréis cambiar algún funcionamiento en concreto.

El servidor DNS que lleva Mikrotik integrado es muy parecido a DNSMasq. Lo configuramos así:

```
[admin@Castalia] > /ip dns print
                 servers: 10.228.144.163
  allow-remote-requests: yes
    max-udp-packet-size: 512
              cache-size: 2048KiB
          cache-max-ttl: 1w
             cache-used: 17KiB
```

Implementaciones: BIND

Servidor DNS más utilizado de Internet, desarrollado actualmente por el ISC. Lleva mucho tiempo en liza y es una apuesta segura, pero tiene una serie de desventajas:

- Complejo de configurar para soluciones sencillas.
- Almacenamiento de las zonas en archivos de texto.
- Desarrollado como un sistema arcaico, poco flexible.

BIND puede actuar como servidor DNS del tipo que queramos: recursivo, cache, primario, secundario. Pero para configurarlo en cualquiera de estas modalidades deberemos dedicar mucho tiempo en entender cómo funciona.

Implementaciones: PowerDNS

PowerDNS es una implementación más moderna que Bind de un servidor DNS, con las siguientes características:

```
* Tiene una implementación independiente de un servidor recursivo
(pdns-recursor).
* Soporta múltiples backends (mysql, oracle, texto, BerkeleyDB,
geo, ldap, odbc, etc.)
```

Instalacion: pdns-recursor

Al instalar pdns-recursor estamos instalando un servidor 100% completo, capaz de resolver por sí mismo cualquier consulta y haciendo caché de los resultados.

Para instalarlo en Debian/Ubuntu, tenemos un paquete ya a medida en la distribución estándar:

```
# apt-get install pdns-recursor
```

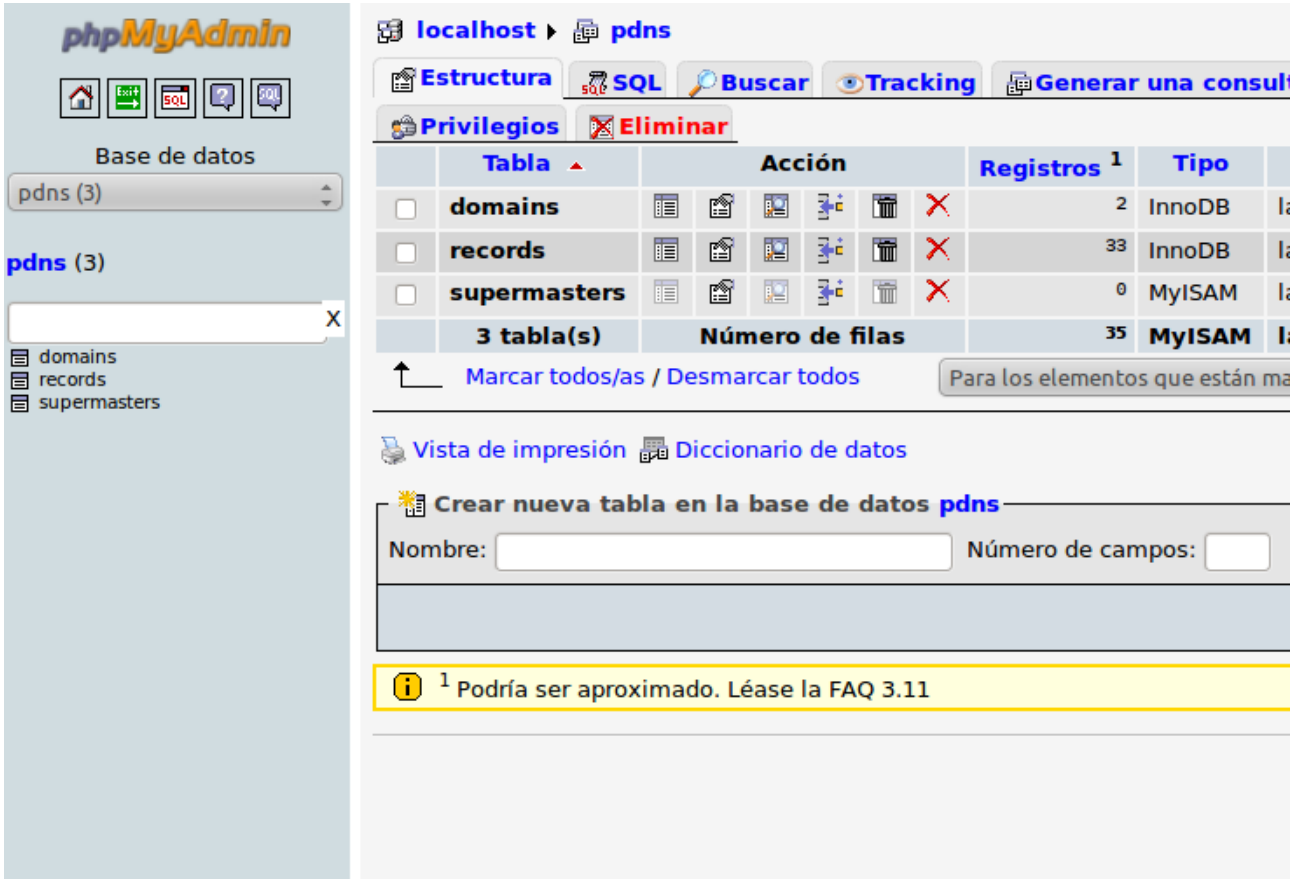
A partir de ese momento ya podemos hacer todo tipo de consultas a nuestro servidor:

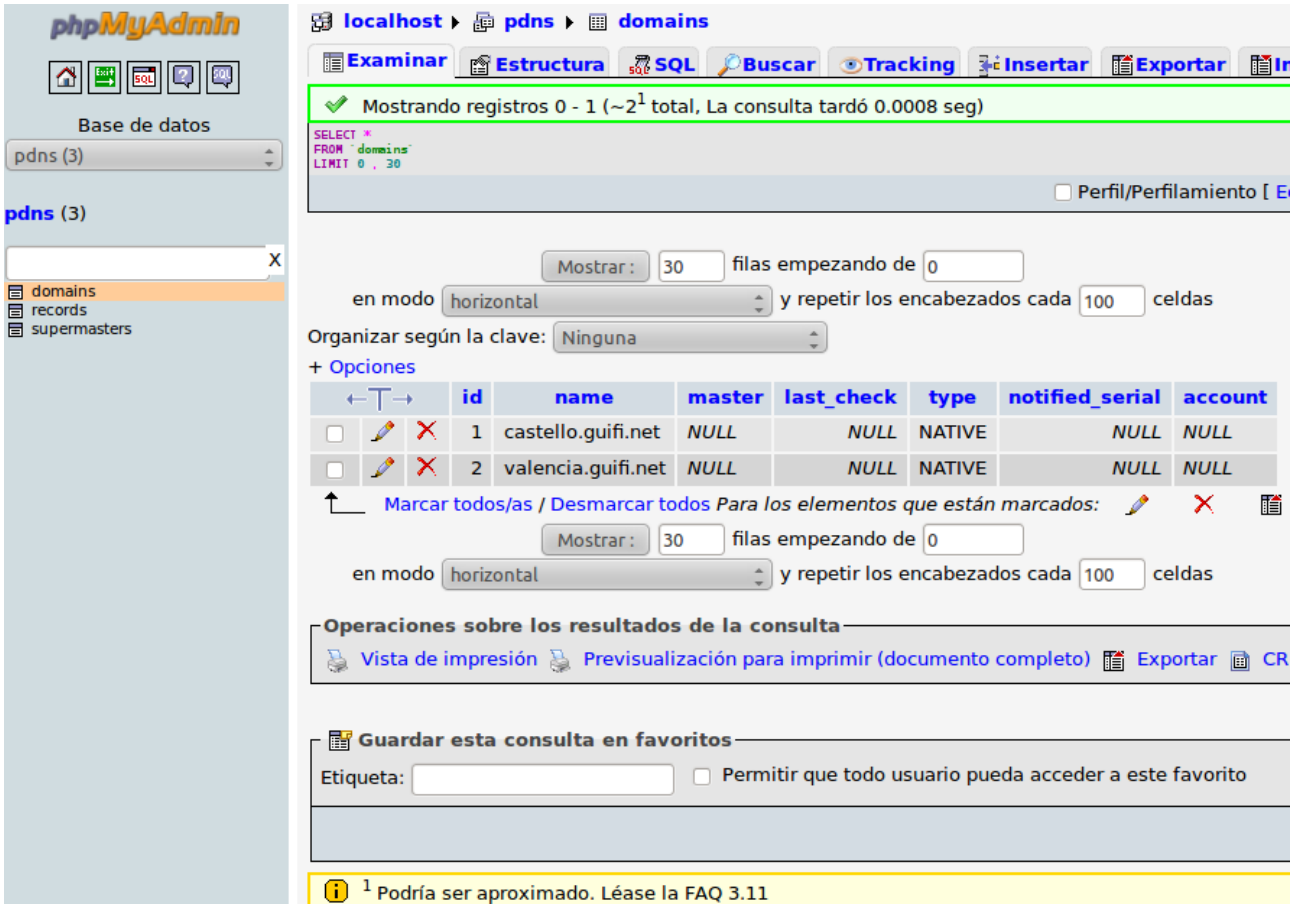
```
$ host www.google.es localhost
www.google.es is an alias for www.google.com.
www.google.com is an alias for www.l.google.com.
www.l.google.com has address 74.125.39.104
www.l.google.com has address 74.125.39.103
www.l.google.com has address 74.125.39.147
www.l.google.com has address 74.125.39.99
www.l.google.com has address 74.125.39.106
www.l.google.com has address 74.125.39.105
```

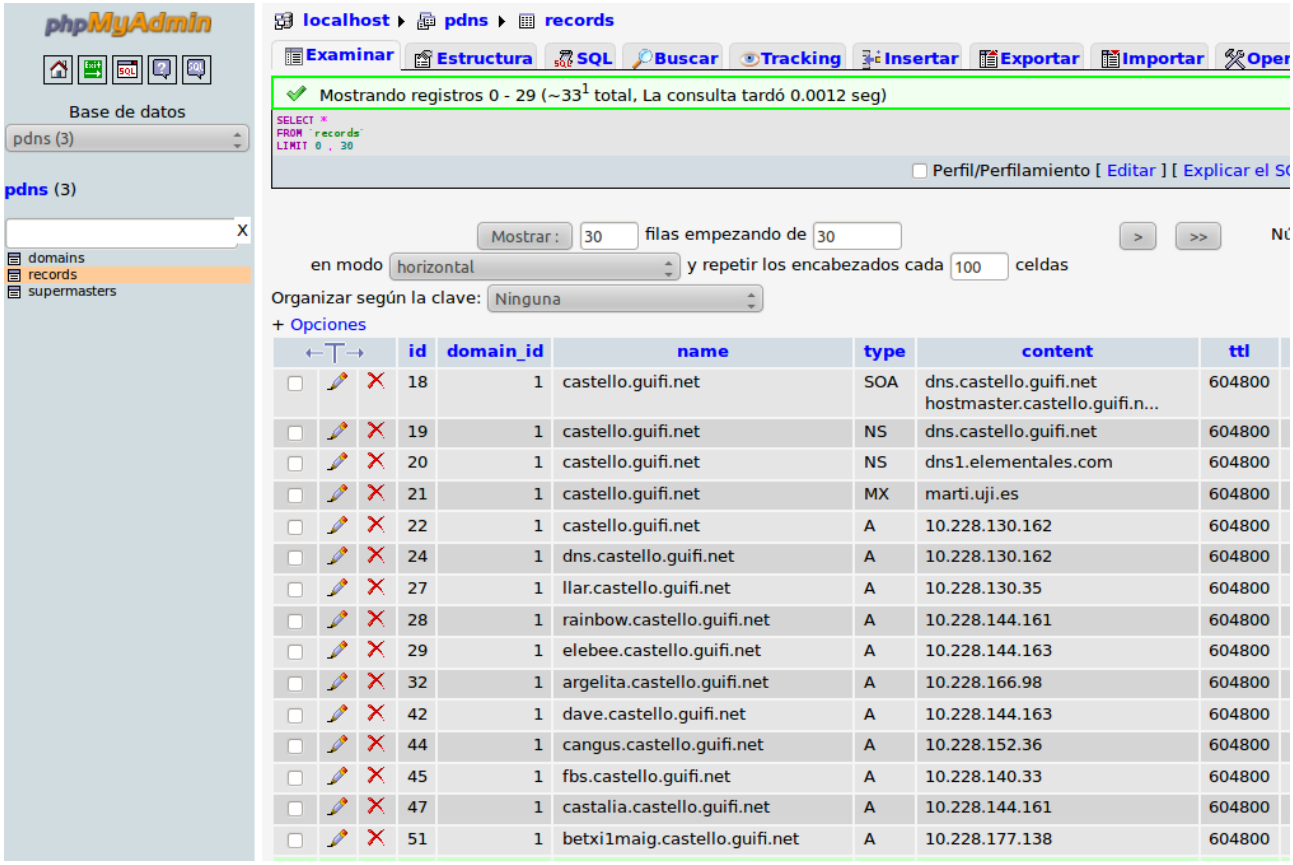
Instalacion: PowerDNS Full

PowerDNS tiene un archivo de configuración extenso pero bastante comprensible, donde seleccionaremos entre otras cosas, el backend donde guardaremos la información de las zonas, si vamos a actuar como DNS recursivo, si vamos a actuar como DNS forwarder, en qué interfaz escucharemos, en qué puerto, etc.

La información de la zona podemos almacenarla, por ejemplo, en una base de datos mysql con la siguiente estructura de tablas como ejemplo:


phpMyAdmin
Base de datos
pdns (3)
pdns (3)
domains
records
supermasters
localhost ▸ pdns
Estructura
SQL
Buscar
Tracking
Generar una consul
Privilegios
Eliminar
Tabla
Acción
Registros 1
Tipo
domains 2 InnoDB
records 33 InnoDB
supermasters 0 MyISAM
3 tabla(s) Número de filas 35 MyISAM
Marcar todos/as / Desmarcar todos
Para los elementos que están ma
Vista de impresión
Diccionario de datos
Crear nueva tabla en la base de datos pdns
Nombre:
Número de campos:
1 Podría ser aproximado. Léase la FAQ 3.11


phpMyAdmin
Base de datos
pdns (3)
pdns (3)
domains
records
supermasters
localhost ▸ pdns ▸ domains
Examinar
Estructura
SQL
Buscar
Tracking
Insertar
Exportar
Mostrando registros 0 - 1 (~2 1 total, La consulta tardó 0.0008 seg)
SELECT *
FROM `domains`
LIMIT 0 , 30
Perfil/Perfilamiento [
Mostrar : 30 filas empezando de 0
en modo horizontal y repetir los encabezados cada 100 celdas
Organizar según la clave: Ninguna
+ Opciones
id name master last_check type notified_serial account
1 castello.guifi.net NULL NULL NATIVE NULL NULL
2 valencia.guifi.net NULL NULL NATIVE NULL NULL
Marcar todos/as / Desmarcar todos Para los elementos que están marcados:
Mostrar : 30 filas empezando de 0
en modo horizontal y repetir los encabezados cada 100 celdas
Operaciones sobre los resultados de la consulta
Vista de impresión
Previsualización para imprimir (documento completo)
Exportar
Guardar esta consulta en favoritos
Etiqueta:
Permitir que todo usuario pueda acceder a este favorito
1 Podría ser aproximado. Léase la FAQ 3.11



id	domain_id	name	type	content	ttl
18	1	castello.guifi.net	SOA	dns.castello.guifi.net hostmaster.castello.guifi.n...	604800
19	1	castello.guifi.net	NS	dns.castello.guifi.net	604800
20	1	castello.guifi.net	NS	dns1.elementales.com	604800
21	1	castello.guifi.net	MX	marti.uji.es	604800
22	1	castello.guifi.net	A	10.228.130.162	604800
24	1	dns.castello.guifi.net	A	10.228.130.162	604800
27	1	llar.castello.guifi.net	A	10.228.130.35	604800
28	1	rainbow.castello.guifi.net	A	10.228.144.161	604800
29	1	elebee.castello.guifi.net	A	10.228.144.163	604800
32	1	argelita.castello.guifi.net	A	10.228.166.98	604800
42	1	dave.castello.guifi.net	A	10.228.144.163	604800
44	1	cangus.castello.guifi.net	A	10.228.152.36	604800
45	1	fbs.castello.guifi.net	A	10.228.140.33	604800
47	1	castalia.castello.guifi.net	A	10.228.144.161	604800
51	1	betxi1maig.castello.guifi.net	A	10.228.177.138	604800

Herramientas y ejemplos de DNS

Tenemos a nuestra disposición varias herramientas para diagnosticar el correcto funcionamiento de un DNS en la red. Recordemos que el DNS funciona con tráfico UDP por el puerto 53, y que una petición de resolución puede d

HOST & DIG

Host y Dig son las herramientas de diagnóstico DNS que sustituyen al clásico “nslookup” para consultar registros DNS en un servidor. “Dig” es para realizar consultas más completas y “host” para consultas simples.

Sintaxis de uso:

```
$ dig @server hostname type
# server: dirección IP del servidor al que queremos consultar
# hostname: registro que queremos consultar
# type: tipo de registro que queremos consultar (A, CNAME, NS)

$ host -t type hostname server
# server: dirección IP del servidor al que queremos consultar
# hostname: registro que queremos consultar
# type: tipo de registro que queremos consultar (A, CNAME, NS)
```

Veamos un ejemplo:

```
malva-serv:/home/dave# dig www.guifi.net

; <<>> DiG 9.7.3 <<>> www.guifi.net
;; global options: +cmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28078
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL:
2

;; QUESTION SECTION:
;www.guifi.net.              IN   A

;; ANSWER SECTION:
www.guifi.net.      2746 IN   CNAME     guifi.net.
guifi.net.          944  IN   A    109.69.8.5

;; AUTHORITY SECTION:
guifi.net.          944  IN   NS   ns1.guifi.net.
guifi.net.          944  IN   NS   ns2.guifi.net.

;; ADDITIONAL SECTION:
ns1.guifi.net.      2746 IN   A    109.69.8.6
ns2.guifi.net.      2746 IN   A    87.216.179.57

;; Query time: 4 msec
;; SERVER: 10.228.130.162#53(10.228.130.162)
;; WHEN: Wed Jul  6 22:14:17 2011
;; MSG SIZE  rcvd: 129
```

Veamos ahora los ejemplos con host:

```
dave@casa:~$ host castello.guifi.net
castello.guifi.net has address 150.128.97.38

dave@casa:~$ host -t NS guifi.net 150.128.98.10
guifi.net name server ns1.guifi.net.
guifi.net name server ns2.guifi.net.
```

DNSTRACER

Utilizando la herramienta **dnstracer** podemos obtener un resultado visual y real de lo que acabamos de contar.

```
dave@haddock:~$ dnstracer -c -4 -s "." www.rediris.es
Tracing to www.rediris.es[a] via A.ROOT-SERVERS.NET, maximum of 3
retries
A.ROOT-SERVERS.NET [.] (198.41.0.4)
 |\___ ns15.communitydns.net [es] (194.0.1.15) Got authoritative
answer
 |\___ ns1.cesca.es [es] (84.88.0.3)
 |     |\___ chico.rediris.es [rediris.es] (130.206.1.3) Got
authoritative answer
 |     |\___ sun.rediris.es [rediris.es] (130.206.1.2) Got
authoritative answer
```

```
 |     |\___ ns02.fccn.pt [rediris.es] (193.136.2.228) Got
authoritative answer
 |     |\___ ns15.communitydns.net [rediris.es] (194.0.1.15) Got
authoritative answer
 |      \___ scsnms.switch.ch [rediris.es] (130.59.1.30) Got
authoritative answer
 |      \___ scsnms.switch.ch [rediris.es] (130.59.10.30) Got
authoritative answer
 |\___ f.nic.es [es] (130.206.1.2) Got authoritative answer
 |\___ a.nic.es [es] (194.69.254.1)
 |     |\___ ns15.communitydns.net [rediris.es] (194.0.1.15) Got
authoritative answer
 |     |\___ ns02.fccn.pt [rediris.es] (193.136.2.228) Got
authoritative answer
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.1.30) Got
authoritative answer
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.10.30) Got
authoritative answer
 |     |\___ chico.rediris.es [rediris.es] (130.206.1.3) Got
authoritative answer
 |      \___ sun.rediris.es [rediris.es] (130.206.1.2) Got
authoritative answer
 |\___ ns-ext.nic.cl [es] (200.1.123.14)
 |     |\___ ns15.communitydns.net [rediris.es] (194.0.1.15) Got
authoritative answer
 |     |\___ chico.rediris.es [rediris.es] (130.206.1.3) Got
authoritative answer
 |     |\___ sun.rediris.es [rediris.es] (130.206.1.2) Got
authoritative answer
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.1.30) Got
authoritative answer
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.10.30) Got
authoritative answer
 |      \___ ns02.fccn.pt [rediris.es] (193.136.2.228) Got
authoritative answer
 |\___ sns-pb.isc.org [es] (192.5.4.1)
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.1.30) Got
authoritative answer
 |     |\___ scsnms.switch.ch [rediris.es] (130.59.10.30) Got
authoritative answer
 |     |\___ ns02.fccn.pt [rediris.es] (193.136.2.228) Got
authoritative answer
 |     |\___ ns15.communitydns.net [rediris.es] (194.0.1.15) Got
authoritative answer
 |     |\___ chico.rediris.es [rediris.es] (130.206.1.3) Got
authoritative answer
 |      \___ sun.rediris.es [rediris.es] (130.206.1.2) Got
authoritative answer
  \___ ns3.nic.fr [es] (192.134.0.49)
       |\___ ns15.communitydns.net [rediris.es] (194.0.1.15) Got
authoritative answer
       |\___ sun.rediris.es [rediris.es] (130.206.1.2) Got
```

```
authoritative answer
       |\___ ns02.fccn.pt [rediris.es] (193.136.2.228) Got
authoritative answer
       |\___ scsnms.switch.ch [rediris.es] (130.59.1.30) Got
authoritative answer
       |\___ scsnms.switch.ch [rediris.es] (130.59.10.30) Got
authoritative answer
        \___ chico.rediris.es [rediris.es] (130.206.1.3) Got
authoritative answer

[...]
```

Ejemplo de integración con la web

Desde la propia web de castello.guifi.net, tenemos una integración de drupal con PowerDNS, de manera que cualquier usuario registrado puede añadir un registro de dns dentro del subdominio .castello.guifi.net. Gracias a PowerDNS la operación es tan sencilla como añadir un nuevo registro a la base de datos MySQL.

Podéis probarlo vosotros/as mismos/as:

- http://castello.guifi.net/content/gestiona-tus-propios-registros-dns-de-castelloguifinet

Tracedump:

```
newBaseSize: 12pt
newBaseSizeInPt: 12
```