1. Salvar la licencia del router mikrotik por si hay que volver a recuperar el software original de mikrotik.
   1. Desde un PC con windows conectar el mikrotik a ethernet 1.
2. Instalar Debian.
3. Preparar el entorno para el buildroot (sistema de cross-compilación de openwrt):
   1. Recomendaciones:
      1. Do everything as non-root user!
      2. Issue all OpenWrt Buildroot commands in the <buildsystem root> directory, e.g. ~/openwrt/trunk/
      3. Do not build in a directory that has spaces in its full path.
      4. Change Ownership other than root user,of the directory where you Download the OpenWrt (sudo chown -R user:user /openwrt/)
   2. Instalar subversion:

**sudo** **apt-get** update

**sudo** **apt-get** **install** subversion build-essential

* 1. Instalar git:

**sudo** **apt-get** **install** git-core

* 1. Instalar demás dependencias necesarias:

**sudo apt-get install libncurses5-dev**

**sudo apt-get install zlib-bin**

**sudo apt-get install zlib1g-dev**

**sudo apt-get install gawk**

**sudo apt-get install unzip**

**sudo apt-get install tftp**

**sudo apt-get install dnsmasq**

**sudo apt-get install atftpd**

1. Bajar el buildroot específico para mikrotik:

$ mkdir /home/user/mikrotik/

$ cd /home/user/mikrotik/

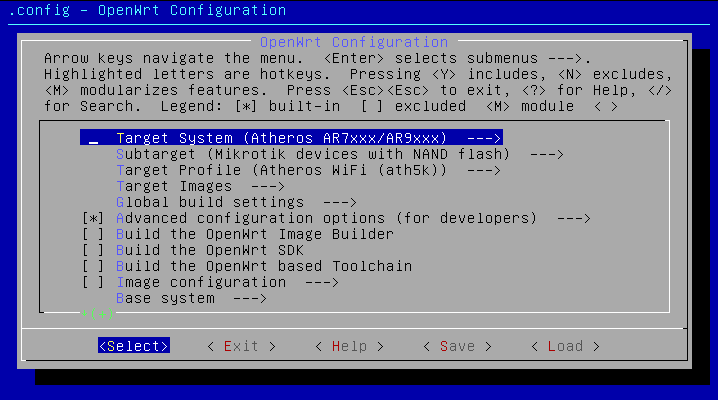
$ svn co svn://svn.openwrt.org/openwrt/trunk/

$ cd /home/user/mikrotik/trunk

1. Compilar la imagen de openwrt que se va a correr desde la RAM del dispositivo.
   1. Configurar la imagen.

$ make menuconfig

Aquí nos sale la pantalla de configuración del menuconfig, que es donde seteamos las opciones para el tipo de dispositivo y los módulos, programas, etc, que vamos a incluir en la imagen que vamos a compilar.



Para activar una opción se da "espacio", para seleccionar una opción dentro de un menú se da "enter", en dependencia del tipo de opción dentro del menú pueden haber tres valores:

* < > : significa que la opción, programa, módulo o lo que sea, no se va a instalar.
* <\*>: significa que la opción se instalará como parte de la imagen que se va a generar.
* <M> significa que la opción se va a compilar pero que no se incluirá como parte de la imagen que se va a generar, sino que se creará un paquete "ipk" para instalarlo después que openwrt esté montado en el dispositivo.

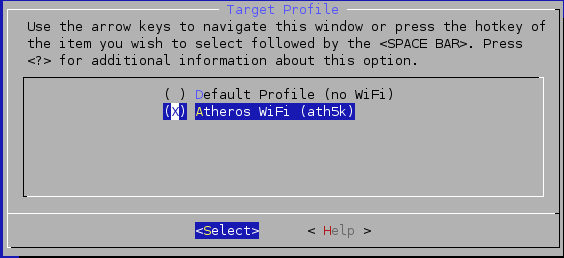
Las opciones que tienes que activar son:

Target = AR7XXX

Subtarget = Mikrotik device with NAND flash

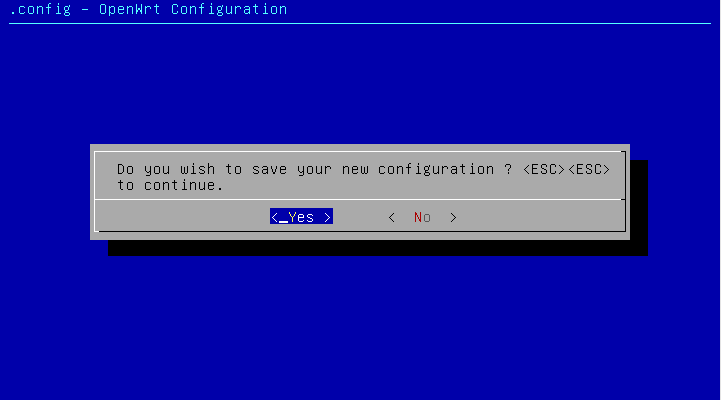
Target Images = ramdisk

En el Target Profile tienes que tener el cuidado de seleccionar "incluir wifi" que por defecto no viene.



Después hay que seleccionar los programas y módulos que nos interesa integrar en el firmware del dispositivo.

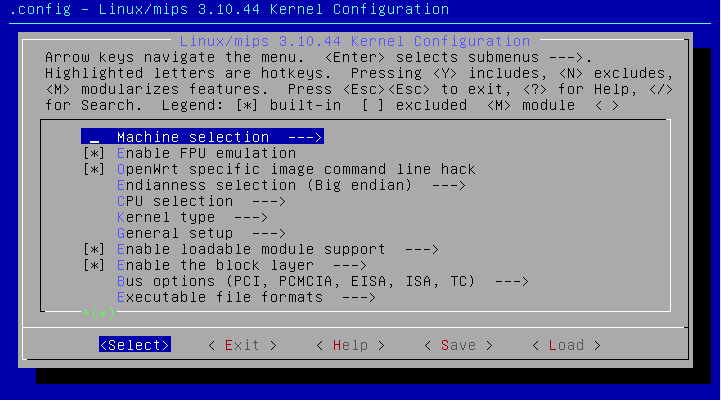
Cuando se haya terminado de seleccionar los programas y módulos se sale del menú de configuración salvando nuestra selección de opciones respondiendo Yes a la pregunta:



* 1. Después hay que descargar y configurar el kernel con:

make kernel\_menuconfig

Esta instrucción descarga el último kernel desde el sitio oficial de linux, lo configura y nos muestra una pantalla de configuración parecida a la de "make menuconfig" en la cual se seleccionan opciones específicas a la configuración del kernel que se va a incluir en openwrt.



* 1. Cuando se haya terminado de configurar el kernel se compila la imagen completa que se va a incluir en el dispositivo con la instrucción:

make V=s

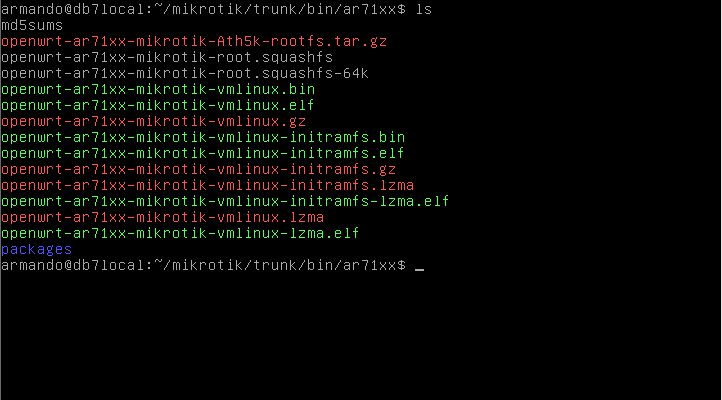
También puede usarse la instrucción:

make V=s 2>&1 | tee build.log | grep -i error

Esta instrucción muestra solo los errores en la pantalla pero guarda un log completo de todo lo que se está haciendo en el archivo trunk/build.log.

Debe recordarse que las opciones que se seleccionaron en "make menuconfig" con <M> se compilan pero no se incluyen en la imagen generada, sino que se crea un paquete ipk para su posterior instalación usando opkg. En el caso de las opciones relativas al kernel (seleccionadas en "kernel\_menuconfig") no me queda claro que esto sea así, pues se supone que aquí la opción <M> significa para los módulos del kernel que estos se van a compilar como parte interna de este o como módulos. (Al menos eso es así para la compilación normal del kernel de linux en otros sistemas).

El resultado final de la compilación se encontrará en el subdirectorio trunk/bin/ar71xx donde estarán las imágenes generadas así como un directorio "packages" dentro del cual estarán los paquetes ipk que se generaron para los programas seleccionados con la opción <M>.



En el caso de la compilación para la imagen que va a correrse en RAM (la que se acaba de realizar) el archivo que se generó fue: openwrt-ar71xx-mikrotik-vmlinux-initramfs.elf

1. Compilar la imagen que se va a flashear al dispositivo.

Para realizar este paso se vuelve al paso 4.1 y en las opciones de la instrucción: make menuconfig se cambia la de "Target Images" de *"ramdisk"* a *"tar.gz"* manteniendo todas las demás opciones como estaban.

Se vuelve a invocar a la instrucción: make V=s ( ó V=s 2>&1 | tee build.log | grep -i error) lo que volverá a compilar, esta vez generándose los archivos "openwrt-ar71xx-mikrotik-vmlinux-lzma.elf" y "openwrt-ar71xx-mikrotik-Ath5k-rootfs.tar.gz" en el subdirectorio trunk/bin/ar71xx.

1. Correr openwrt en la RAM del dispositivo:
   1. Copiar el archivo generado "openwrt-ar71xx-mikrotik-vmlinux-initramfs.elf" para el directorio <user>/tftp/ . Si ese directorio no existe crearlo.
   2. Crear un script dentro del directorio tftp llamado loader.sh que tenga el siguiente contenido:

#/bin/bash

ifconfig eth0 192.168.1.10 up

dnsmasq -i eth0 --dhcp-range=192.168.1.100,192.168.1.200 \

--dhcp-boot=openwrt-ar71xx-mikrotik-vmlinux-initramfs.elf \

--enable-tftp --tftp-root=/home/<user>/mikrotik/trunk/bin/ar71xx/ -d -u <user> -p0 -K --log-dhcp --bootp-dynamic

Aclaración: No copiar y pegar el contenido anterior en el archivo a generar, pues pueden generarse problemas en el script debido a las diferencias entre la forma en que Windows y linux interpretan los espacios y las tabulaciones.

Este script deberá tener permisos de ejecución. Esto se hace ejecutando la siguiente instrucción después de creado:

chmod +x loader.sh

* 1. Conectar el puerto ethernet 1 del dispositivo a la PC con un cable de red, sin conectar el dispositivo a la alimentación.
  2. Ubicarse en el directorio <user>/tftp
  3. Correr el script loader.sh con la instrucción:

./loader.sh

Esto configura eth0 con la dirección 192.168.1.10 y activa dnsmasq como servidor de dhcp que servirá la imagen "openwrt-ar71xx-mikrotik-vmlinux-initramfs.elf" al dispositivo cuando este se la pida usando bootp.

El sistema se quedará a la espera en el prompt.

* 1. Con una presilla mantener oprimido el botón RES que está al lado de la toma de alimentación del dispositivo.
  2. Conectar la toma de alimentación del dispositivo (manteniendo oprimido el botón RES).
  3. El dispositivo enciende el led PWR y en unos segundos empezará a flashear el led ACT, mientras esto ocurra mantener RES oprimido.
  4. Cuando el led ACT deje de flashear soltar el botón RES, y si todo se hizo bien el Debian le debe cargar al dispositivo el openwrt en la RAM, y seguido bootear openwrt.

Durante este proceso se podrá ver en la pantalla de la PC la interacción entre esta y el dispositivo a través de dhcp y bootp.

El proceso transcurre muy rápido, y después de cargada la imagen de openwrt en la RAM del dispositivo este reinicia y se carga openwrt como sistema operativo. Durante esta fase el led ACT flashea, quedándose fijo cuando ya está listo todo.

* 1. Una vez que ya openwrt está en la RAM cambiar el cable de red a otro puerto del dispositivo distinto del 1 y hacer telnet a la dirección 192.168.1.1. Si todo salió bien se entra en la interface CLI de openwrt.

1. Flashear la imagen de openwrt en la NAND del dispositivo:
   1. Estando logeado por telnet en el dispositivo generar un password para el usuario root. (Esto puede parecer no necesario pero sí lo es porque si no se hace no se puede copiar posteriormente con "scp" las imágenes a flashear en el dispositivo):

root@OpenWrt:/# passwd

* 1. Preparar el dispositivo para copiarle las imágenes a flashear:

root@OpenWrt:/# mtd erase /dev/mtd1

root@OpenWrt:/# mtd erase /dev/mtd2

root@OpenWrt:/# mkdir /mnt/kernel

root@OpenWrt:/# mkdir /mnt/rootfs

root@OpenWrt:/# mount /dev/mtdblock1 /mnt/kernel

root@OpenWrt:/# mount /dev/mtdblock2 /mnt/rootfs

* 1. Copiar las imágenes a flashear para el dispositivo:

Para esto se debe salir de telnet y volver al prompt de Debian.

scp /home/<user>/mikrotik/trunk/bin/ar71xx/openwrt-ar71xx-mikrotik-vmlinux-lzma.elf root@192.168.1.1:/tmp

scp /home/<user>/mikrotik/trunk/bin/ar71xx/openwrt-ar71xx-mikrotik-Ath5k-rootfs.tar.gz root@192.168.1.1:/tmp

* 1. Volver a entrar en el dispositivo, pero esta vez por ssh (al crear el password openwrt deshabilita telnet y activa ssh):

ssh -l root 192.168.1.1

* 1. Realizar los pasos que flashean el kernel como tal:

root@OpenWrt:/# mv /tmp/openwrt-ar71xx-mikrotik-vmlinux-lzma.elf /mnt/kernel/kernel

root@OpenWrt:/# chmod +x /mnt/kernel/kernel

root@OpenWrt:/# umount /mnt/kernel

root@OpenWrt:/# cd /mnt/rootfs

root@OpenWrt:/# tar -xvzf /tmp/openwrt-ar71xx-mikrotik-Ath5k-rootfs.tar.gz

root@OpenWrt:/# cd /

root@OpenWrt:/# umount /mnt/rootfs

* 1. Reiniciar el dispositivo.

A partir de este punto ya el dispositivo tiene openwrt instalado y para conectarse a él deberá hacerse esta operación usando los puertos ethernet desde el 2 hasta el 5. El puerto 1 es el puerto para conectarse a la WAN, por lo tanto openwrt no distribuye direcciones ip en él por dhcp, sino que, al contrario, viene configurado por defecto para buscar por dhcp una dirección para ese puerto.

El password que se le configuró a openwrt en el paso 7.1 se pierde cuando este se reinicia pues openwrt estaba cargado en la RAM del dispositivo; por lo tanto cuando este se reinicia hay que volver a entrar por telnet y generar el password nuevamente. A partir de ahí ya ese password queda guardado en la flash y si es persistente entre reinicios del dispositivo.

1. Preparación de openwrt para la instalación de SiCat.

Openwrt viene configurado por defecto con las siguientes características:

* Todas las interfaces locales (ethernet de la 2 a la 5 y la interface wifi) haciendo un puente a una interface virtual llamada br-lan la cual es configurada con la dirección 192.168.1.1
* La radio wifi desactivada.
* El paquete dnsmasq configurado para distribuir direcciones ip en el rango 192.168.1.100 - 150.
* El paquete "firewall" instalado y configurado con reglas típicas para el uso del dispositivo como un AP.
* El paquete LuCI como interface web para la configuración del dispositivo.

La configuración inicial del dispositivo se puede hacer tanto por la línea de comandos habiéndose conectado previamente al router por ssh ( ssh -l root 192.168.1.1) o usando la interface web (http://192.168.1.1).

Para el uso del programa SiCat es necesario activar la interface wifi y desinstalar el paquete "firewall"

* 1. Activación de la interface wifi:
     1. Usando la línea de comandos:
* root@OpenWrt:~# nano /etc/config/wireless
* En el archivo que se abre con nano modificar la última línea que aparece en el bloque *"config wifi-device radio0"* de:

option disabled 1

a:

#option disabled 1

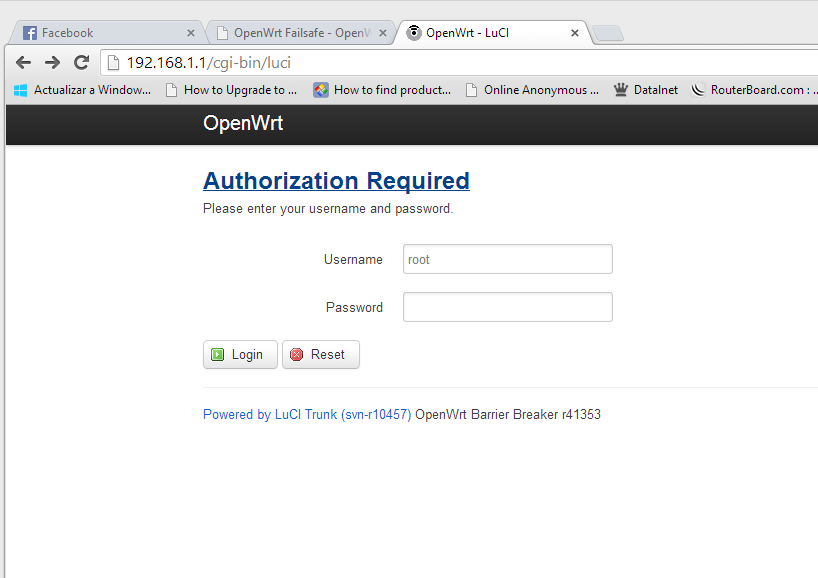
* Salvar el archivo: ctrl + O
* Salir de nano: ctrl + X
* Reiniciar la interface wifi:

root@OpenWrt:~# wifi restart

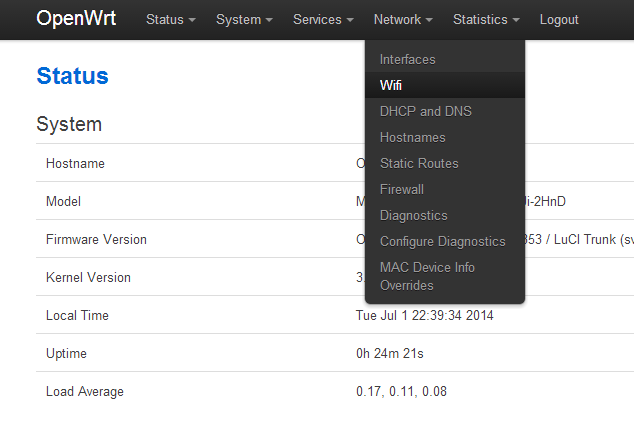
Deberá aparecer la red wifi de openwrt (SSID = OpenWrt) disponible a los dispositivos wifi que se encuentren en el entorno.

* + 1. Usando LuCI:

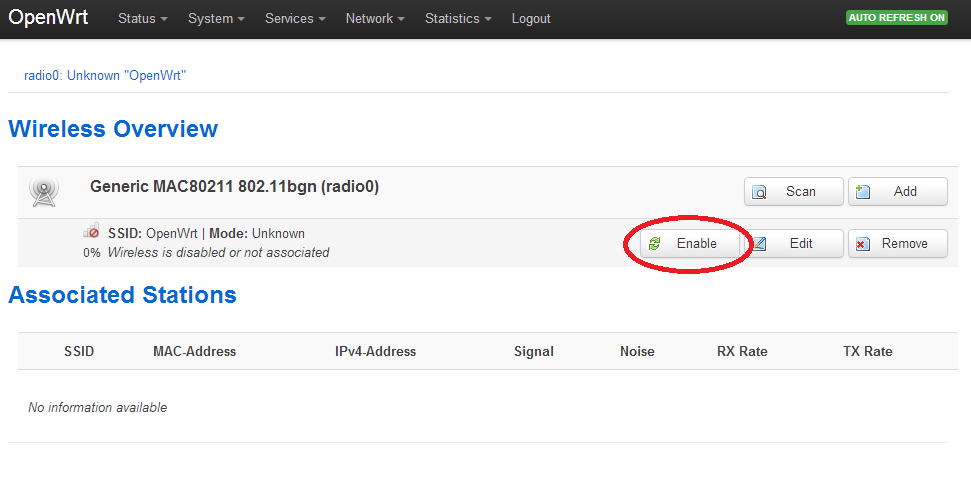
Entrar en la interface web usando nuestro browser. Deberemos poner la clave que se configuró para root del dispositivo:



En el menú principal de la página de configuración ir a la opción "Network/Wifi":



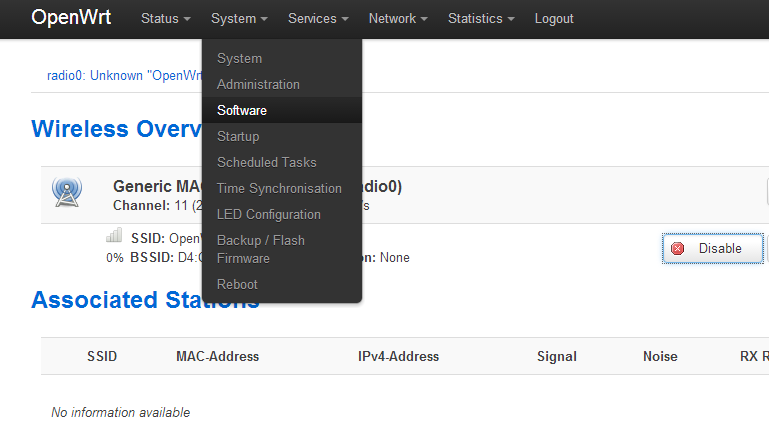
Activar el botón "Enable":



A partir de este punto ya disponemos de red wifi.

* 1. Desinstalación del paquete "Firewall"

Este paso se realiza usando la interface web LuCI.

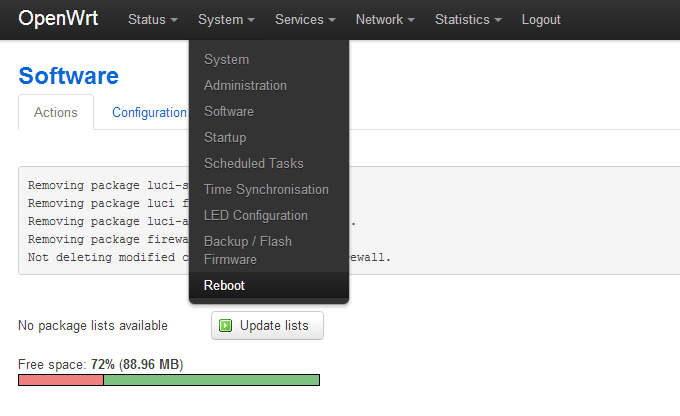


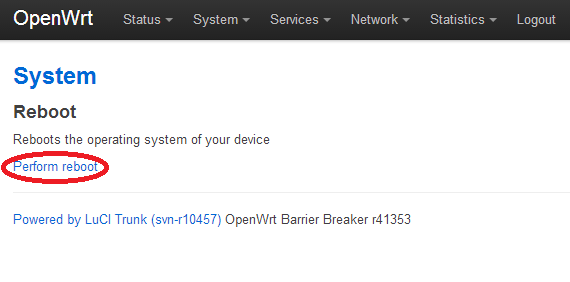
En el listado de paquetes instalados buscamos el paquete firewall:



Activamos la opción: "Remove".

Reiniciar el dispositivo:





Para comprobar que el paquete fue debidamente desinstalado desde la línea de comandos chequeamos el estado de iptables:

*root@OpenWrt:~# iptables -L -n*

*Chain INPUT (policy ACCEPT)*

*target prot opt source destination*

*Chain FORWARD (policy ACCEPT)*

*target prot opt source destination*

*Chain OUTPUT (policy ACCEPT)*

*target prot opt source destination*

No deberá haber ninguna regla definida para ninguna de las cadenas de iptables.

1. Instalación de SiCat:

Este paso se realiza desde la línea de comandos:

* 1. Copiar el paquete en cuestión para el subdirectorio /tmp del dispositivo:

scp SiCat\_1.0.0-1\_ar71xx.ipk root@192.168.1.1:/tmp/

* 1. En el dispositivo crear el subdirectorio /var/opkg-lists:

mkdir /var/opkg-lists

* 1. Copiar el archivo "Packages" para /var/opkg-lists/packages:

scp Packages root@192.168.1.1:/var/opkg-lists/packages

* 1. En el dispositivo ir al subdirectorio /tmp e instalar el paquete con opkg:

opkg install SiCat\_1.0.0-1\_ar71xx.ipk

Una vez instalado SiCat, y antes de reiniciar el dispositivo, deberemos configurar los parámetros de trabajo de openwrt para la localización donde estamos instalando el router, incluyendo los parámetros de trabajo de SiCat. De no realizarse esta configuración justo en el momento posterior de haberse instalado SiCat (que aún este no se ha cargado en memoria) nos estaremos arriesgando a que no podamos entrar al dispositivo posteriormente para configuración usando LuCI.

Debe recordarse que en esta primera versión de SiCat el único modo que se soporta es el uso del router como AP estando conectado por la interface WAN (puerto ethernet 1 del dispositivo) directamente a internet a través de un modem-router configurado como bridge, o a través de otro router que le brinde conectividad a Internet a través de una interface ethernet configurada como bridge o como router. En este último caso la interface WAN deberá configurarse manualmente poniendo como gateway por defecto la dirección IP del router al que se está conectando el mikrotik.

1. Configuración de SiCat.

El programa SiCat se configura en el archivo sicat.conf que está ubicado en el subdirectorio /etc.

Las variables que son importante configurar son:

* 1. LoginTimeout: Este es el tiempo que un usuario podrá estar conectado a Internet después de haberse autentificado correctamente en facebook. Cuando este timeout se haya cumplido, el usuario deberá volver autentificarse para poder seguir navegando. Por defecto se establece en 5 minutos.
  2. ExternalDevice: Interface de red que está conectada a la WAN (Internet). Por defecto eth0.
  3. InternalDevice: Interface de red a la que se conectarán los clientes del servicio. Por defecto se establece a br-lan , que es la interface virtual del puente que openwrt trae por defecto uniendo las interfaces ethernet y la wifi.

El posible uso de configuraciones de red en la cual se haga diferenciación entre las interfaces ethernet y la wifi (es decir se elimine el puente entre ellas), o que se establezcan diferentes redes wifi para distintos servicios, deberá esperar por experimentación a realizarse por el equipo de desarrollo de SiCat.

Cuando estas variantes estén disponibles en esta variable lo que se pondrá será una lista de las interfaces en las cuales se brindará el servicio.

* 1. LocalNetwork: Red local de la interface br-lan en la cual se brindará el servicio de SiCat. Por defecto 192.168.1.0/24
  2. DNSAddr: El servidor DNS que los usuarios de la red local usarán. En caso de usarse dnsmasq con su opción de DNS forwarding (por defecto viene activada en openwrt), deberá ponerse en esta variable la misma dirección ip del router en la red local (por defecto 192.168.1.1), sino deberá ponerse en esta variable una dirección ip de un servidor DNS en internet que usarán los usuarios para este protocolo.
  3. AllowedWebHosts: Esta variable es importante que se establezca a la propia dirección del router en la red local (192.168.1.1) para poder configurar el router usando la interface web (LuCI) una vez que se haya reiniciado y el programa SiCat se haya activado.

Todas las demás variables del archivo sicat.conf deberán dejarse en los valores por defecto que traen.

1. Reiniciar el router.