To setup olsrd using APRouter Pro

This is with ver6.1 APRouter Pro f/w and assumes everything is defaults to start off with.

Assuming a basic, two unit, mesh...

Node1 connected to internet (via any of the LAN ports). In this example the main LAN is subnet 192.168.0.0/255.255.255.0 with our internet router on 192.168.0.11.

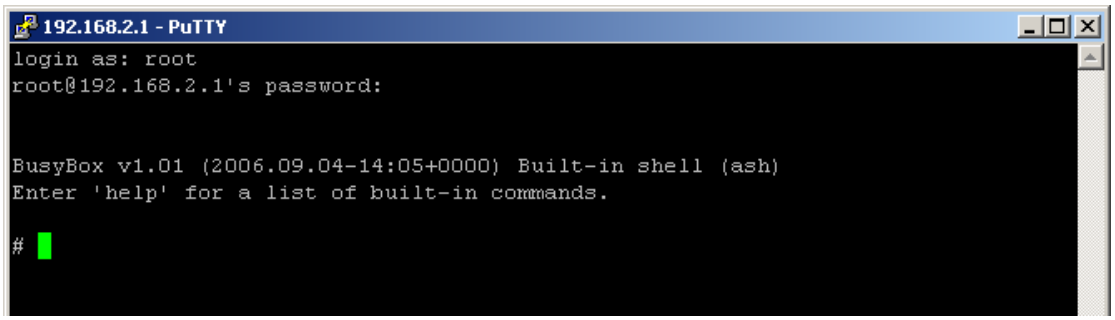
Node2 acting as repeating mesh node

The basics are:

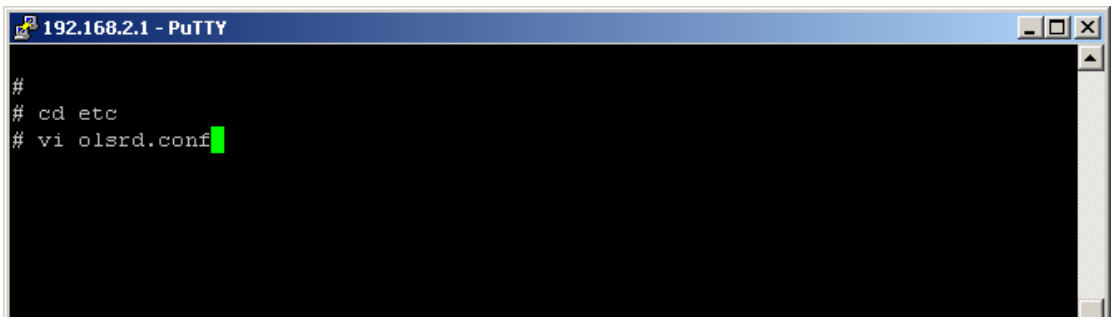
- Make sure that olsrd.conf config file for the main for main node (Node1 – the unit connected to the internet) has HNA4 entry showing that it has internet access (0.0.0.0/0.0.0.0). HNA4 for other nodes should be empty.
- Make sure that olsrd.conf config file for all units has interface “bro” and “wlan0” entry.
- wireless mode is Client/Ad Hoc.
- Set TCP mode as Bridge.
- Give each node a different LAN address (easiest way is to set the LAN ports as DHCP clients so each one will get an address from our main router).
- WAN settings don’t matter but you will need to enable Meshing (olsrd) on WAN setup page.
- Add ifconfig wlan0 entry into main script so it’s executed each time there’s reboot
- Setup Watchdog to reboot if the connection fails for any reason.
- Test...

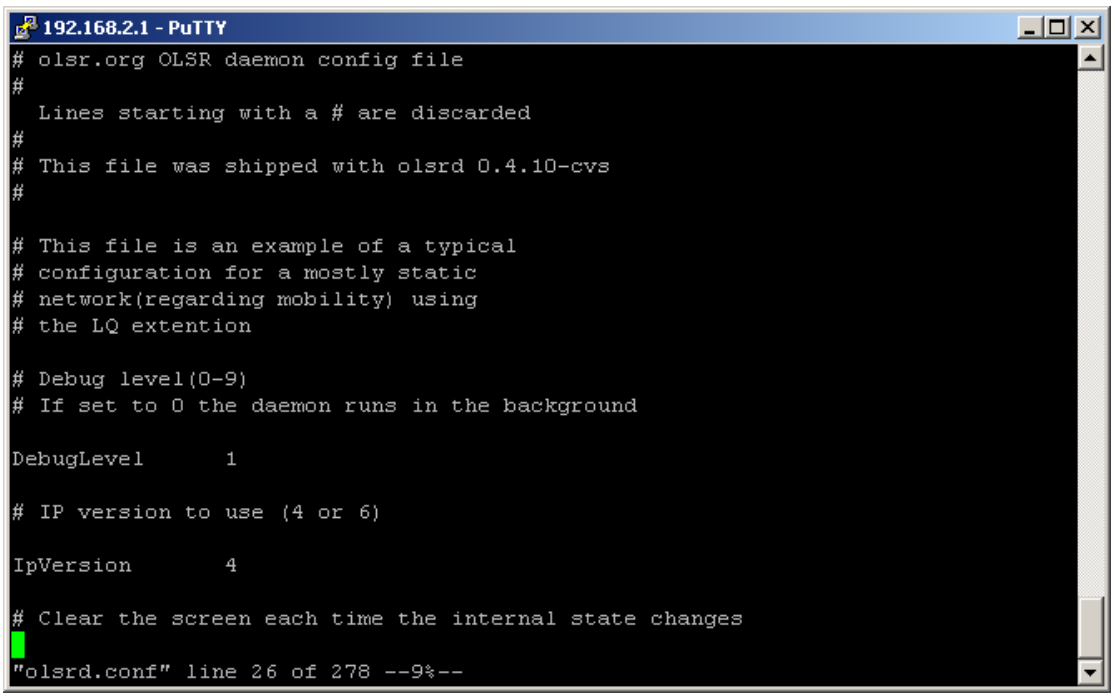
1 Configuring the olsrd.conf configuration file

First of all edit the olsrd.conf file. To do that you need to ssh into the unit using PuTTY or similar. Use the username ‘root’ and password ‘admin’



Now goto the etc directory and start the vi editor (apologies for this... the vi editor has got to be the worst line editor I’ve ever, EVER, seen but that’s all there is I’m afraid)





Scroll down to the ‘Hna4’ entry...

```
Hna4
{
#   Internet gateway:
   0.0.0.0   0.0.0.0
#   more entries can be added:
#   192.168.0.0  255.255.255.0
}

# HNA IPv6 routes
# syntax: netaddr prefix
# Example Internet gateway:
```

Now, for Node1, you need to add an entry which tells the olsr network that this is where the main internet access is (0.0.0.0 0.0.0.0). Quick instructions for vi: move curser to where you want to add. Then do ESCAPE ‘i’ – this puts you into insert mode. Now when you type then, hopefully, text should appear on the screen (though I agree that vi is REALLY cr*p!). If you do ESCAPE ‘x’ then it goes into delete mode and, each time you press ‘x’ it deletes the character under the curser. However, frequently the display gets mucked up so it’s worth doing a few Page Downs and Ups form time to time to make sure it’s going according to plan ☺ Anyway... hopefully you can manage to add a line for your network!

For Node2 all entries in the Hna4 section should be remmed out (with ‘#’ at the start of the line).

Now scroll down to the ‘Interface’ section....

192.168.2.1 - PuTTY

```
# default values. Multiple interfaces
# can be specified in the same block
# and multiple blocks can be set.

# !!CHANGE THE INTERFACE LABEL(s) TO MATCH YOUR INTERFACE(s)!!
# (eg. wlan0 or eth1):

Interface "br0"
{

    # IPv4 broadcast address to use. The
    # one usefull example would be 255.255.255.255
    # If not defined the broadcastaddress
    # every card is configured with is used

    # Ip4Broadcast              255.255.255.255

    # IPv6 address scope to use.
    # Must be 'site-local' or 'global'

    # Ip6AddrType               site-local

    # IPv6 multicast address to use when
"olsrd.conf" line 223 of 278 --80%--
```

The aim here is to change the Interface entry so it just has the br0 an wlan0 interfaces there. So use the ESCAPE ‘i’ and the ESCAPE ‘x’ commands to edit the line (I know it’s hard but persevere!). Set the same for both Node1 and Node2.

```
# default values. Multiple interfaces
# can be specified in the same block
# and multiple blocks can be set.

# !!CHANGE THE INTERFACE LABEL(s) TO MATCH YOUR INTERFACE(s)!!
# (eg. wlan0 or eth1):

Interface "wlan0" "br0"
{

    # IPv4 broadcast address to use. The
    # one usefull example would be 255.255.255.255
    # If not defined the broadcastaddress
    # every card is configured with is used

    Ip4Broadcast                255.255.255.255

    # Emission intervals.
    # If not defined, RFC proposed values will
    # be used in most cases.

    # Hello interval in seconds(float)
"olsrd.conf" line 167 of 202 --82%--
```

Now we need to write the file and quite the editor.

Do ESCAPE ‘:w’ to write the file. The ESCAPE ‘:q’ to quit vi.

Now you need to ensure the new file is saved to flash so enter the command ‘save’

```
192.168.2.1 - PuTTY
    # IPv4 broadcast address to use. The
    # one usefull example would be 255.255.255.255
    # If not defined the broadcastaddress
    # every card is configured with is used

    # Ip4Broadcast              255.255.255.255

    # IPv6 address scope to use.
    # Must be 'site-local' or 'global'

    # Ip6AddrType               site-local

    # IPv6 multicast address to use when
#
#
#
#
#
#
# save
running set mode
wrote: 21395 bytes
etc Salvo
#
```

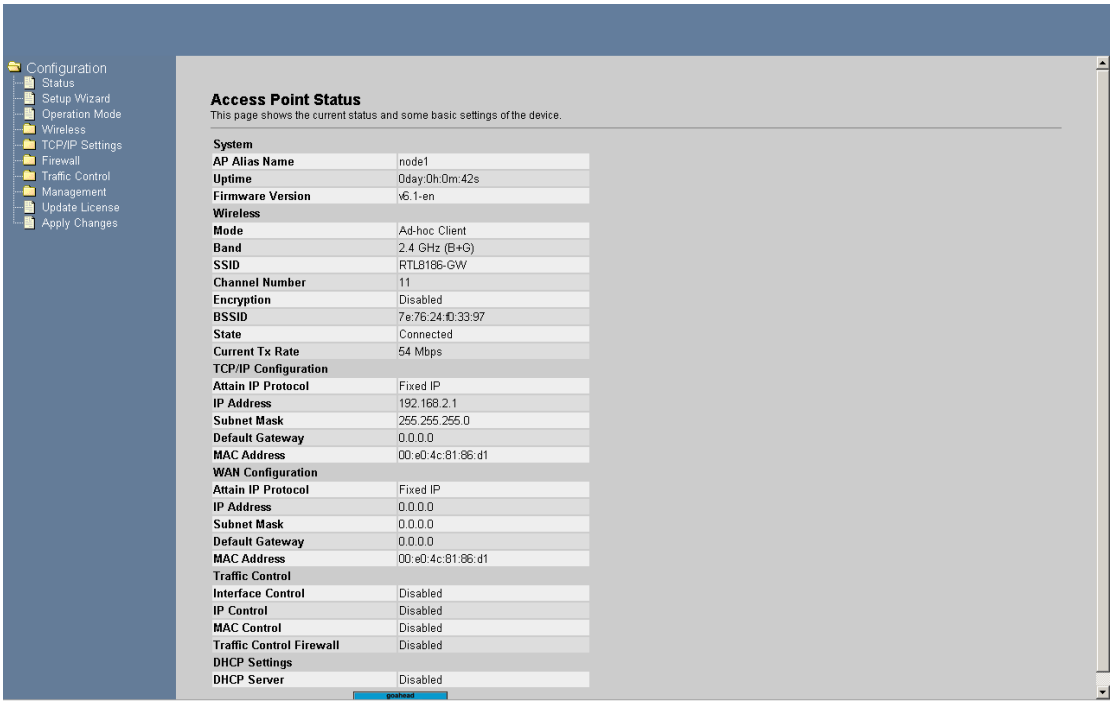
You can now exit PuTTY.

The full olsrd.conf file is listed at the end of the document.

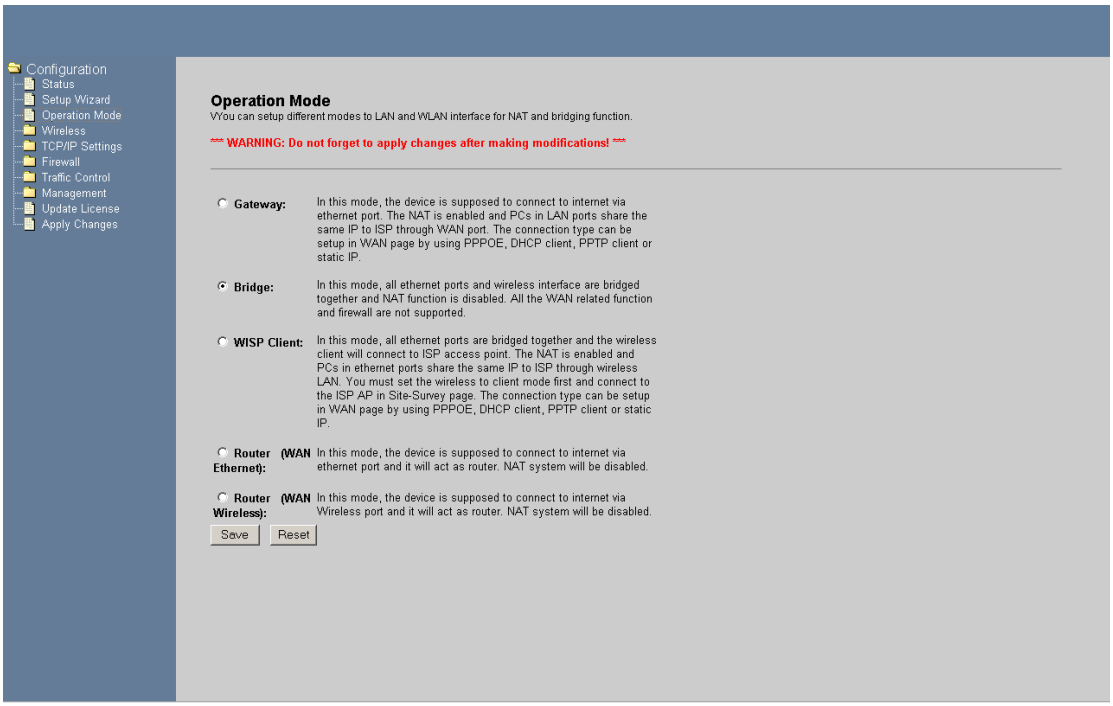
2 Configuration of the nodes via the web interface

Remember, after doing a change on each web page, you must SAVE the changes.

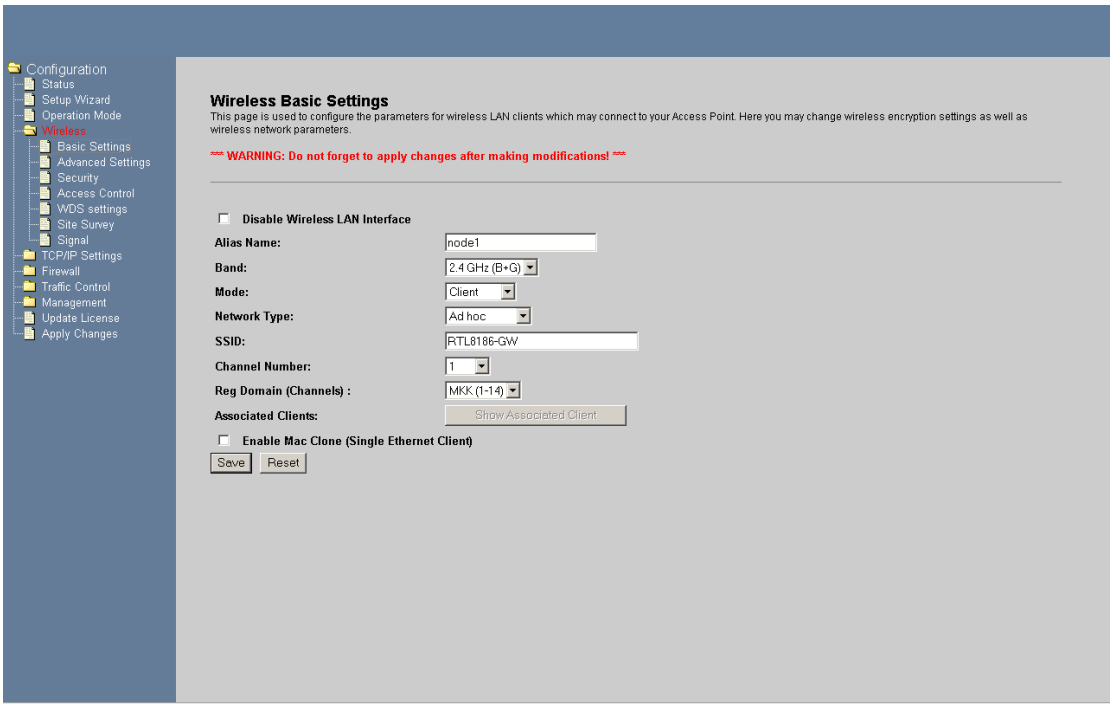
Enter the web setup via your browser (default IP 192.168.2.1)



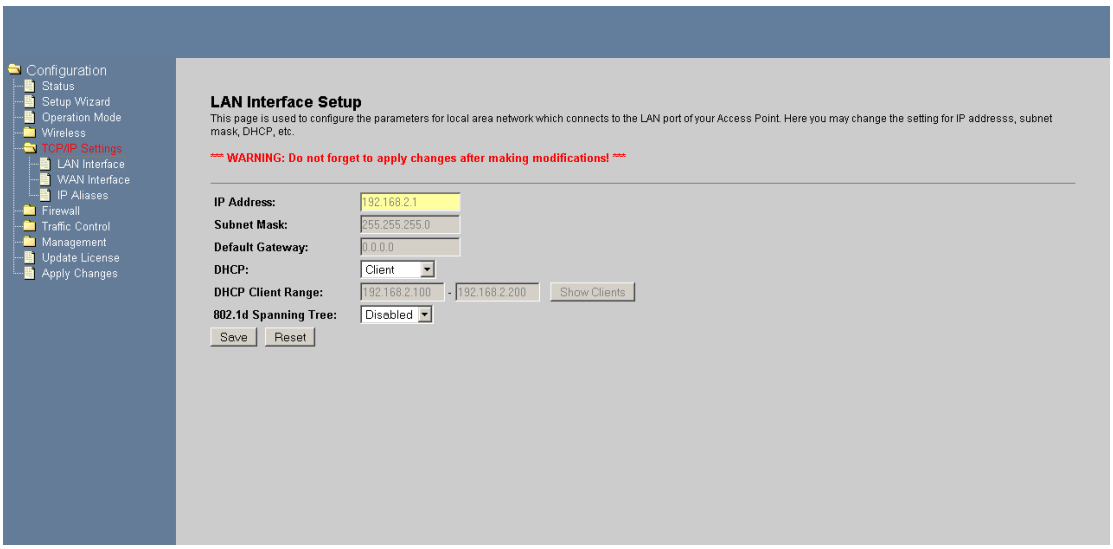
Goto Operation Mode – make sure it's Bridge



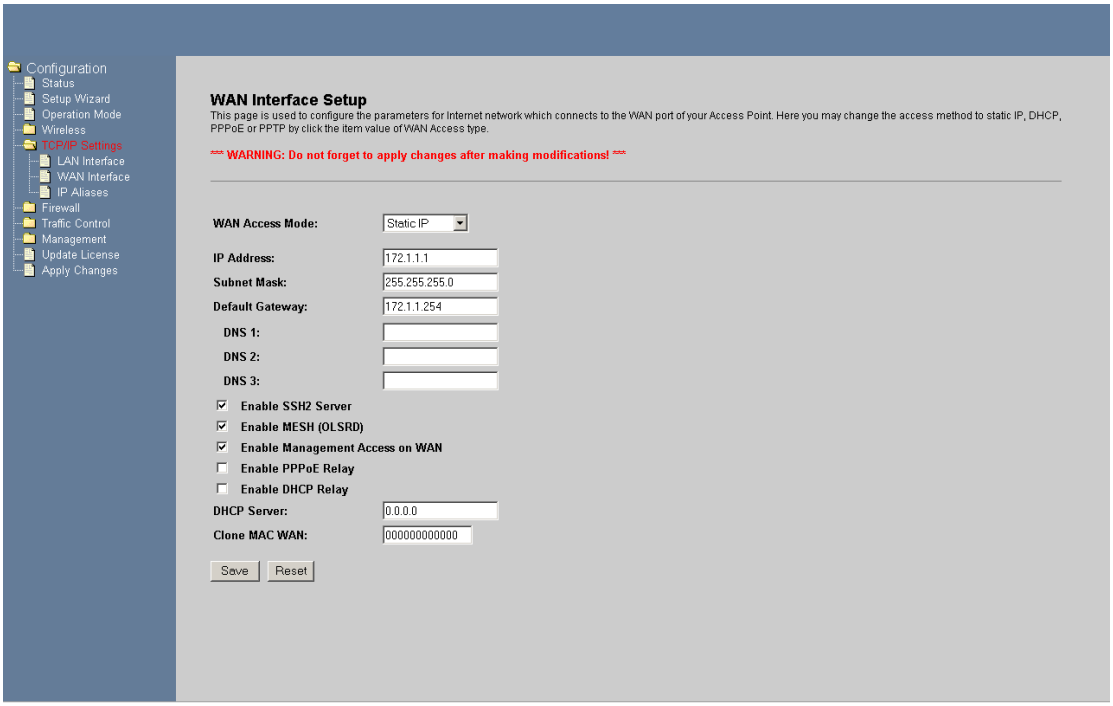
Goto Wireless/Basic – Give unit a name (e.g. Node1 for the main unit, connected to the internet. Node2 for the one acting as a repeating node) and then make sure setup for Client/Ad hoc



Goto TCP/IP Settings/LAN Interface – give each node a unique address. You can either give each node and address from your main subnet or set the LAN interface as a DHCP client (though, if you do this, you'll have to check with your DHCP server what address has been allocated)



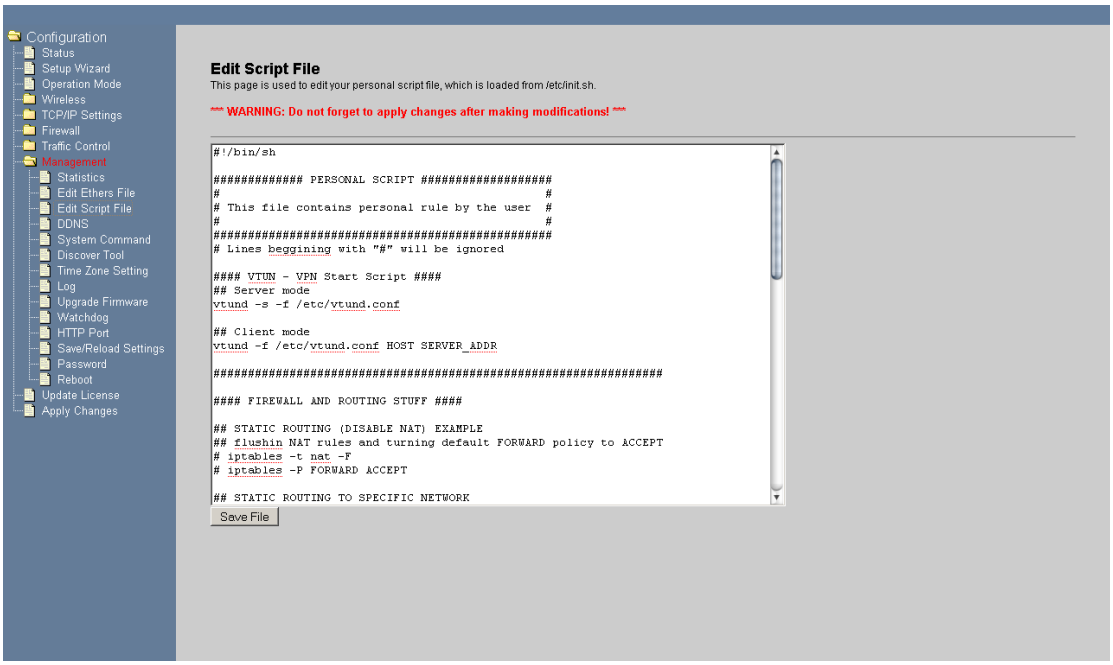
Goto WAN Interface



The WAN interface is not actually used so the IP settings etc... are unimportant. The only important thing is to make sure the box 'Enable MESH (OLSRD) is ticked.

Now, before the wlan0 interface will load, you need to give it an IP address (olsr is a routing protocol so that means the interfaces need IP addresses).

To do this you can add a simple line to the main startup script called init.sh in the etc directory. The easiest way to do this is via Web gui on the Management/Edit Script File page:

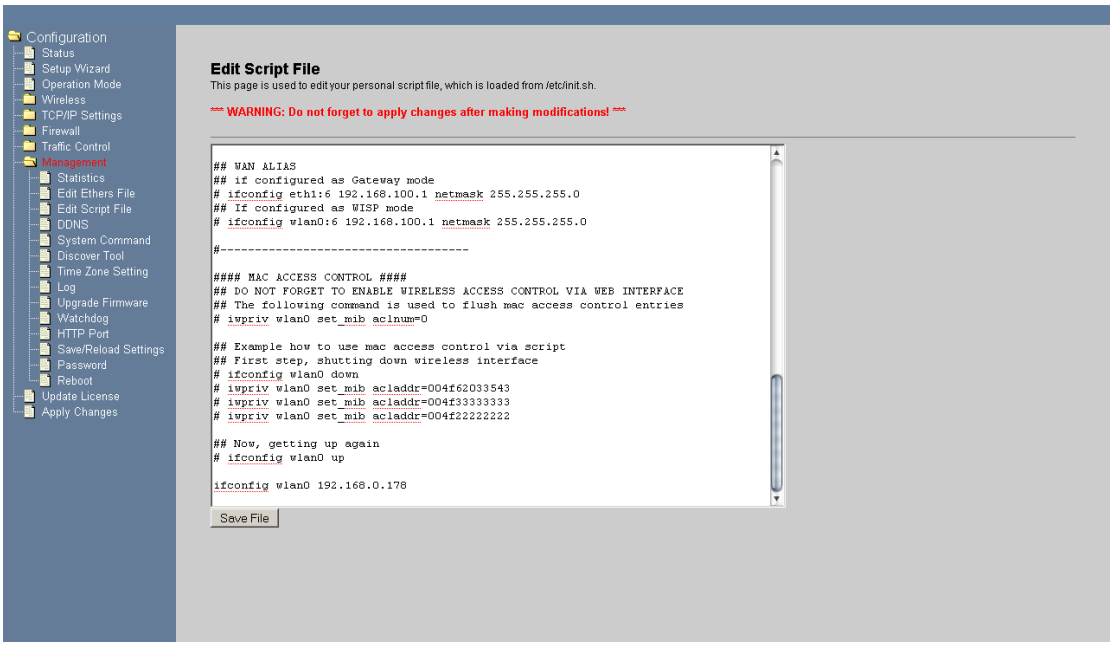


Scroll down to the bottom of the script and add an extra line of the form...

ifconfig wlan0 <ip address>

Use an unused IP address which is in your main range but outside those used by your DHCP server (e.g. our server only gives out addresses up to 100 so any address above there is free for static use – check with your system if you are unsure!):

e.g.

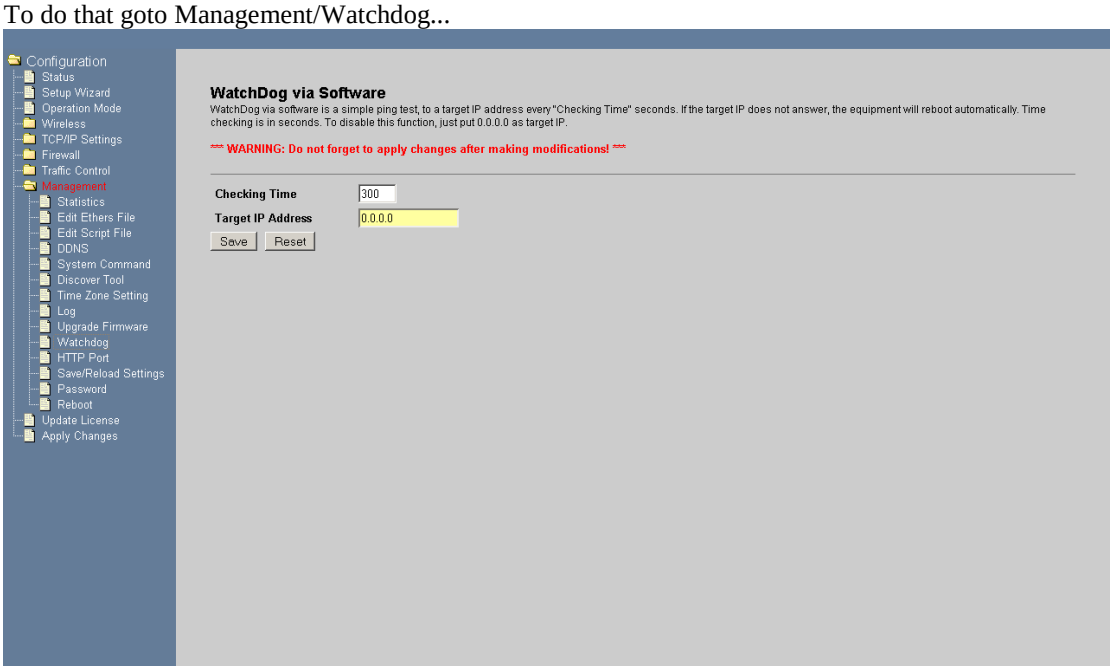


For each node use a different IP address (but still in your main range).

Remember to Save File.

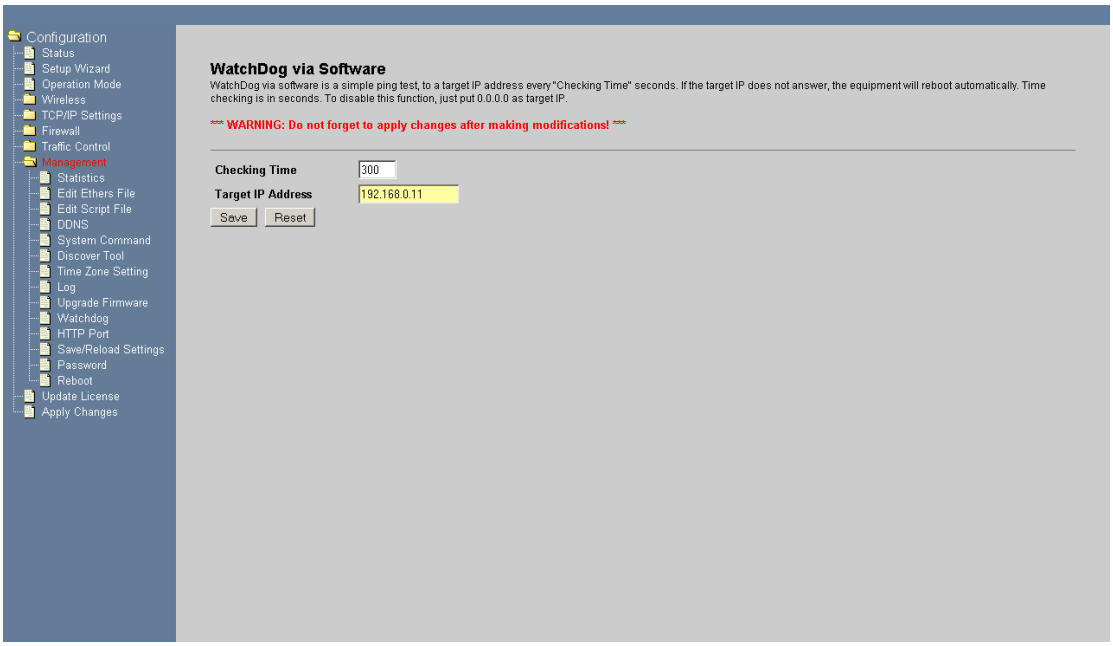
Also it's a good idea to setup the Watchdog (this will reboot the unit if ever it sees something has gone wrong with the link).

To do that goto Management/Watchdog...



You need to enter an IP address for the unit to check connectivity to. I'd suggest the address of your router (on your main LAN). Also set the Checking Time to a sensible value (not too low so that after the slightest hiccup it starts rebooting):

e.g. our router is 192.168.0.11 but use your OWN router address



Remember to save settings and then finally Apply Changes.

Now.... a word of warning, once you enable MESH mode and you have enabled debug mode in the olsrd.conf file then, when you reboot the unit, the web interface is disabled!! Below are notes of now to disable the olsrd via ssh/commands.

3 Testing

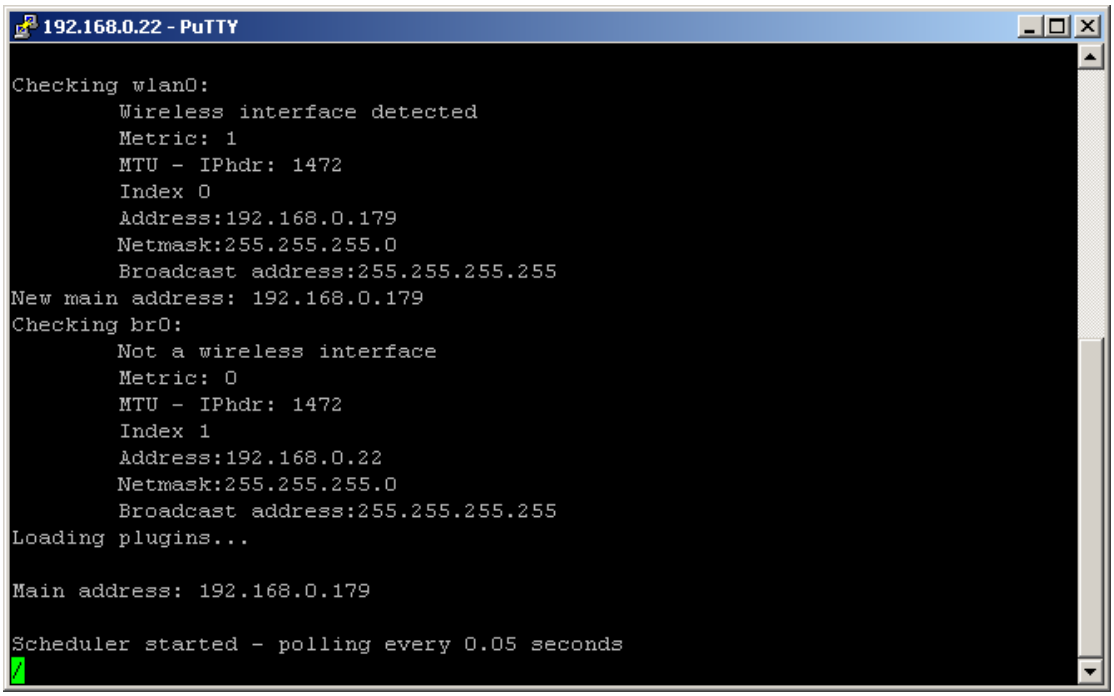
3.1 Basic connectivity – Does it work?

Make sure you've saved all the settings and then reboot the units. Now, if everything's up and running then the obvious test is to see if your wireless PC will connect to the mesh network. When it's connected then, if your PC is set to get IP settings automatically (using DHCP) then your PC should obtain a valid set of IP values and then be able to get onto the internet.

3.2 Checking OLSRD via the meshing units

It's possible to run the meshing in debug mode on the AP units which will log the mesh connectivity. To do this use PuTTY to enter command mode. Then, enter the command 'olsrd'

The olsrd will execute and print out a list of the interfaces...



Below shows the output from a few seconds of running with debug turned on in the olsrd.conf file...

--- 00:59:23.81 --- LINKS

IP address hyst LQ lost total NLQ ETX
192.168.0.178 0.000 0.200 0 2 1.000 5.00
192.168.0.24 0.000 0.200 0 2 1.000 5.00

--- 00:59:23.81 -- NEIGHBORS

IP address LQ NLQ SYM MPR MPRS will
192.168.0.178 0.200 1.000 YES NO NO 3

```
--- 00:59:23.81 ------------------------------------------------ TOPOLOGY

Source IP addr   Dest IP addr     LQ     ILQ    ETX
      *** olsr.org - 0.4.10 (Mar  8 2006) ***
(ioctl)Adding route with metric 1 to 192.168.0.24/255.255.255.255 via 192.168.0.24/br0.
(ioctl)Adding route with metric 1 to 192.168.0.178/255.255.255.255 via 192.168.0.24/br0.

--- 00:59:25.53 ------------------------------------------------- LINKS

IP address       hyst   LQ     lost   total  NLQ    ETX
192.168.0.178    0.000  0.200  0      2      1.000  5.00
192.168.0.24     0.000  0.300  0      3      1.000  3.33

--- 00:59:25.53 ---------------------------------------------- NEIGHBORS

IP address       LQ     NLQ    SYM   MPR   MPRS  will
192.168.0.178    0.300  1.000  YES   NO    NO    3

--- 00:59:25.53 ------------------------------------------------ TOPOLOGY

Source IP addr   Dest IP addr     LQ     ILQ    ETX
      *** olsr.org - 0.4.10 (Mar  8 2006) ***
(ioctl)Deleting route with metric 1 to 192.168.0.178/255.255.255.255 via 192.168.0.24/br0.
(ioctl)Deleting route with metric 1 to 192.168.0.24/255.255.255.255 via 192.168.0.24/br0.
(ioctl)Adding route with metric 1 to 192.168.0.178/255.255.255.255 via 192.168.0.178/br0.
Add route(192.168.0.178): File exists
(ioctl)Adding route with metric 1 to 192.168.0.24/255.255.255.255 via 192.168.0.178/br0.
Add route(192.168.0.24): File exists
.
.
.
.
.
```

Above you can see the SSH screens, with olsrd debug log running and node entries for the neighbouring units.

3.3 Checking OLSRD via a Windows PC

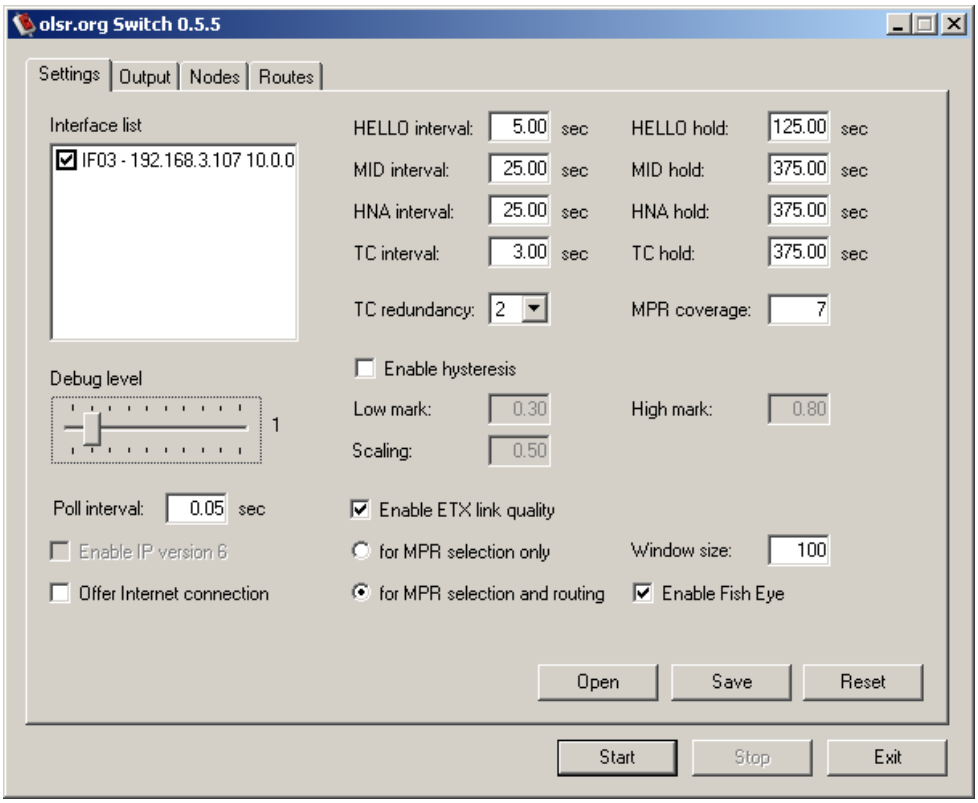
You can further test by running the olsrd switch daemon on any windows PC connected to the mesh network (via LAN port of any meshing unit or wirelessly to any meshing unit).

You can download the olsrd daemon at...

http://www.olsr.org/releases/0.5/olsrd-0-5-5-setup.exe

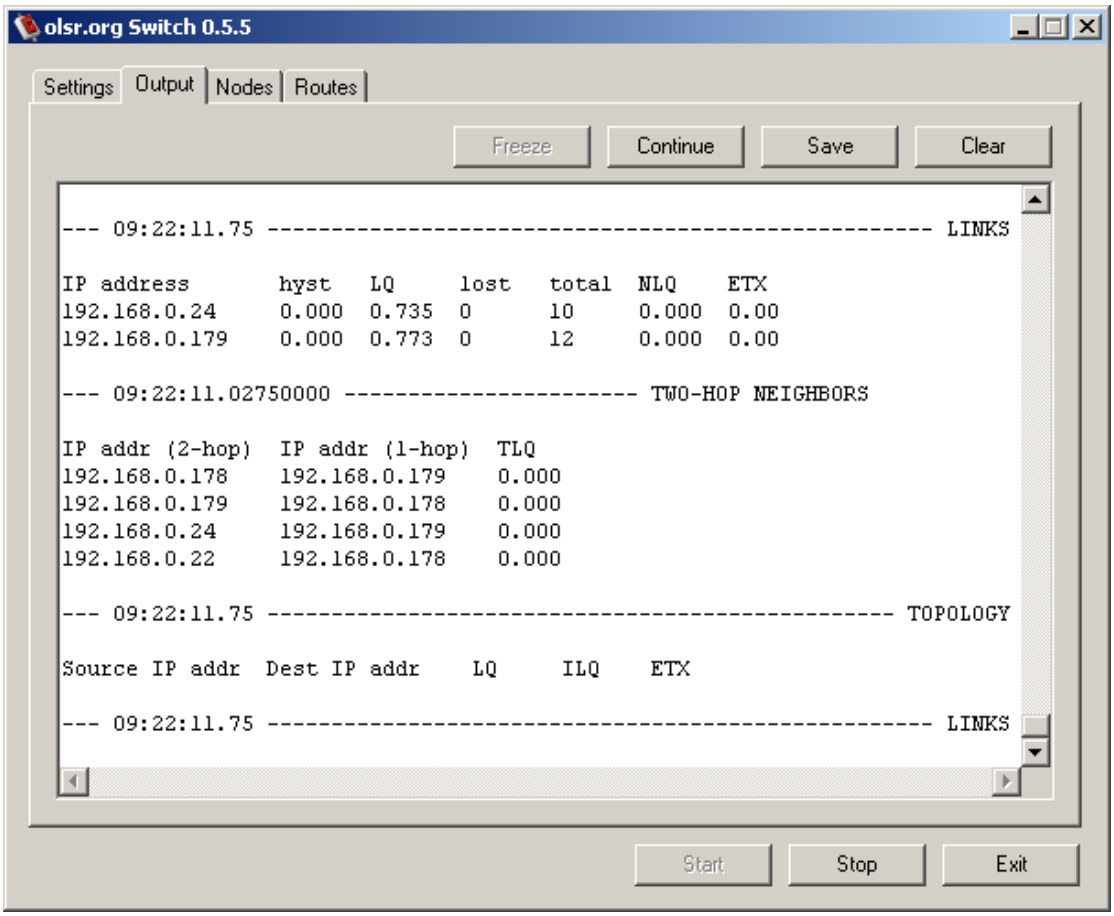
Download and install this application. Next make sure you PC has an IP address in the same subnet as that used for your olsrd nodes.

Run the olsrd switch.



In the Interfaces list you should see the IP address of the lan interface of your PC (I have lot's!). Set the Debug Level to 1 and then click on Start...

Now goto the Output screen:

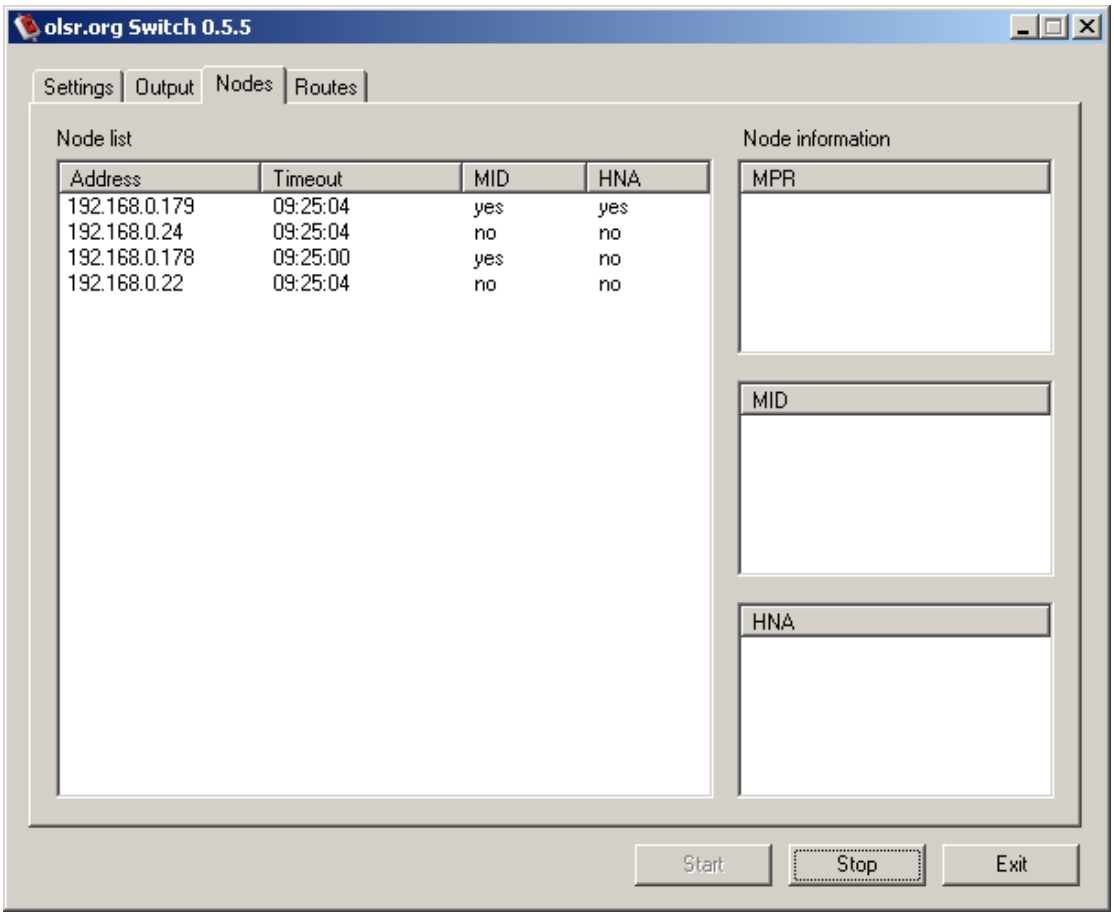


You can see the other olsr interfaces listsed.

In my case

0.22 is LAN of my main node
0.179 is the WLAN of my main node
0.24 is LAN of my main node
0.178 is the WLAN of my main node

Also, if I look at the Nodes page is see...



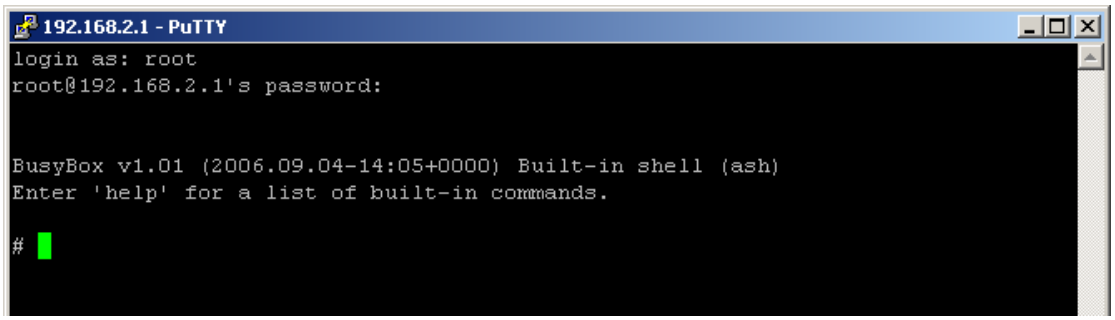
Notice the extra node addresses. Notice that HNA is showing next the WLAN of my main node. Also notice that the WLAN interfaces of the nodes are now shown as MID. A MID or Multiple interface declaration(MID) is essentially an interfaces on which a node runs OLSR.

If you can't see your nodes then something's wrong ☺

4 Disabling olsrd via SSH

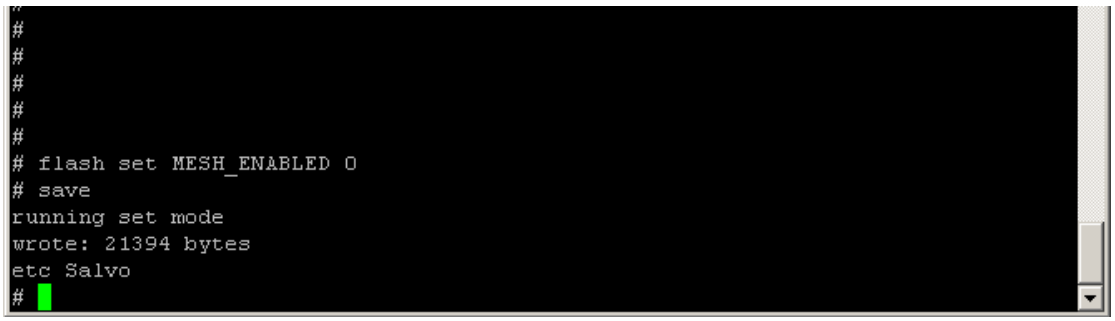
As mentioned above, when meshing is enabled with debug selected in the olsrd.conf file then, when you reboot the unit, the web interface is disabled. So these are instructions for how to use ssh (PuTTY) to disable olsrd. You can then reboot and the web interface will work again.

So, use PuTTY to enter the command screen of the device...



You can permanently cease olsrd by editing the value set in the flash i.e.

Now save the changes...



Another option, to temporarily turn olsrd off is to just kill that process using the killall linux comannd....

Now you can exit PuTTY.

If you have turned olsrd off my changing the flash value then you need to reboot to see the effect (though, remember, if you have truned the setting off then you will need to turn it back on in the WAN page of the web setup). When it comes back up the web interface should be working.

If you just used killall to temporarily stop the process then the web interface should immediately be working.

5 Olsrd.conf file

```
#
# olsr.org OLSR daemon config file
#
# Lines starting with a # are discarded
#
# This file was shipped with olsrd 0.4.10-cvs
#

# This file is an example of a typical
# configuration for a mostly static
# network(regarding mobility) using
# the LQ extention

# Debug level(0-9)
# If set to 0 the daemon runs in the background
# Leave as 0 (off) for normal running

DebugLevel      1

# IP version to use (4 or 6)

IpVersion   4

# Clear the screen each time the internal state changes

ClearScreen     yes

# HNA IPv4 routes
# syntax: netaddr netmask
# Example Internet gateway:
# 0.0.0.0 0.0.0.0

Hna4
{
#   Internet gateway: for main node ONLY
   0.0.0.0      0.0.0.0
#   more entries can be added:
#   192.168.0.0  255.255.255.0
}

# HNA IPv6 routes
# syntax: netaddr prefix
# Example Internet gateway:
Hna6
{
#   Internet gateway:
#   ::              0
#   more entries can be added:
#   fec0:2200:106:: 48
}


# Should olsrd keep on running even if there are
```

```
# no interfaces available? This is a good idea
# for a PCMCIA/USB hotswap environment.
# "yes" OR "no"

AllowNoInt  yes

# TOS(type of service) value for
# the IP header of control traffic.
# If not set it will default to 16

#TosValue   16

# The fixed willingness to use(0-7)
# If not set willingness will be calculated
# dynamically based on battery/power status
# if such information is available

Willingness       7

# Allow processes like the GUI front-end
# to connect to the daemon.

IpcConnect
{
     # Determines how many simultaneously
     # IPC connections that will be allowed
     # Setting this to 0 disables IPC

     MaxConnections 0

     # By default only 127.0.0.1 is allowed
     # to connect. Here allowed hosts can
     # be added

     Host            127.0.0.1
     #Host            10.0.0.5

     # You can also specify entire net-ranges
     # that are allowed to connect. Multiple
     # entries are allowed

#     Net             192.168.2.0 255.255.255.0
}

# Wether to use hysteresis or not
# Hysteresis adds more robustness to the
# link sensing but delays neighbor registration.
# Used by default. 'yes' or 'no'

UseHysteresis     no

# Hysteresis parameters
# Do not alter these unless you know
# what you are doing!
# Set to auto by default. Allowed
# values are floating point values
# in the interval 0,1
# THR_LOW must always be lower than
# THR_HIGH.

#HystScaling      0.50
```

```
#HystThrHigh      0.80
#HystThrLow 0.30


# Link quality level
# 0 = do not use link quality
# 1 = use link quality for MPR selection
# 2 = use link quality for MPR selection and routing
# Defaults to 0

LinkQualityLevel  2

# Link quality window size
# Defaults to 10

LinkQualityWinSize      10

# Polling rate in seconds(float).
# Default value 0.05 sec

Pollrate    0.05


# TC redundancy
# Specifies how much neighbor info should
# be sent in TC messages
# Possible values are:
# 0 - only send MPR selectors
# 1 - send MPR selectors and MPRs
# 2 - send all neighbors
#
# defaults to 0

TcRedundancy      2


#
# MPR coverage
# Specifies how many MPRs a node should
# try select to reach every 2 hop neighbor
#
# Can be set to any integer >0
#
# defaults to 1

MprCoverage 3


# Olsrd plugins to load
# This must be the absolute path to the file
# or the loader will use the following scheme:
# - Try the paths in the LD_LIBRARY_PATH
#   environment variable.
# - The list of libraries cached in /etc/ld.so.cache
# - /lib, followed by /usr/lib

# Example plugin entry with parameters:

#LoadPlugin "olsrd_dyn_gw.so.0.3"
#{
    # Here parameters are set to be sent to the
```

```
    # plugin. Theese are on the form "key" "value".
    # Parameters ofcause, differs from plugin to plugin.
    # Consult the documentation of your plugin for details.

    # Example: dyn_gw params

    # how often to check for Internet connectivity
    # defaults to 5 secs
#   PlParam     "Interval"   "40"

    # if one or more IPv4 addresses are given, do a ping on these in
    # descending order to validate that there is not only an entry in
    # routing table, but also a real internet connection. If any of
    # these addresses could be pinged successfully, the test was
    # succesful, i.e. if the ping on the 1st address was
successful,the
    # 2nd won't be pinged
#   PlParam     "Ping"       "141.1.1.1"
#   PlParam     "Ping"       "194.25.2.129"
#}



# Interfaces and their rules
# Omitted options will be set to the
# default values. Multiple interfaces
# can be specified in the same block
# and multiple blocks can be set.

# !!CHANGE THE INTERFACE LABEL(s) TO MATCH YOUR INTERFACE(s)!!
# (eg. wlan0 or eth1):

Interface "br0" "wlan0"
{

    # IPv4 broadcast address to use. The
    # one usefull example would be 255.255.255.255
    # If not defined the broadcastaddress
    # every card is configured with is used

    # Ip4Broadcast            255.255.255.255

    # IPv6 address scope to use.
    # Must be 'site-local' or 'global'

    # Ip6AddrType       site-local

    # IPv6 multicast address to use when
    # using site-local addresses.
    # If not defined, ff05::15 is used

    # Ip6MulticastSite        ff05::11

    # IPv6 multicast address to use when
    # using global addresses
    # If not defined, ff0e::1 is used

    # Ip6MulticastGlobal      ff0e::1


    # Emission intervals.
```

```
    # If not defined, RFC proposed values will
    # be used in most cases.

    # Hello interval in seconds(float)
    HelloInterval    10.0

    # HELLO validity time
    HelloValidityTime   100.0

    # TC interval in seconds(float)
    TcInterval        3.0

    # TC validity time
    TcValidityTime      30.0

    # MID interval in seconds(float)
    MidInterval   5.0

    # MID validity time
    MidValidityTime     30.0

    # HNA interval in seconds(float)
    HnaInterval   5.0

    # HNA validity time
    HnaValidityTime     30.0

    # When multiple links exist between hosts
    # the weight of interface is used to determine
    # the link to use. Normally the weight is
    # automatically calculated by olsrd based
    # on the characteristics of the interface,
    # but here you can specify a fixed value.
    # Olsrd will choose links with the lowest value.

    # Weight 0


}
```