

ICTP 2012

Mesh Networking Lab

Sebastian Buettrich sebastian@less.dk

last updated: Feb 2012

Lab

- Flashing a wireless device
- OpenWRT based mesh nodes – on mixed hardware (Linksys WRT54G, Ubiquiti Pico, NanoStation & Bullet)
- Routing protocol: OLSR

- Optional: Trying B.A.T.M.A.N., 802.11s
- Optional: Running olsrd on a PC/laptop
- Optional: Running batmand on a PC/laptop
- Optional: Android phone mesh / servalproject.org

Flashing

- **Flashing** a wireless device means **replacing its firmware with an alternative firmware**
- Methods for flashing:
 - 1) Through web GUI – often works for devices in default state.
 - 2) via tftp – when the web GUI does not work
 - 3) if all that fails – via serial ports, soldering etc
 - 4) if nothing works: you have a *brick* :)

Choosing firmware

- Find out exactly what hardware model and hardware revision you have
- Find the right firmware accordingly and verify that it will run

Examples of firmware:

OpenWRT – list of supported hardware:

<http://wiki.openwrt.org/toh/start>

We will use a variety of Ubiquiti devices, so check

<http://wiki.openwrt.org/toh/start#ubiquiti>

- Freifunk http://wiki.freifunk.net/Freifunk_Firmware_%28English%29

Classical Freifunk firmware was targetted at Linksys WRT54G and similar devices

Newest Freifunk software comes as packages to be installed under OpenWRT. Kamikaze/Builder allows for creation of packages for all hardware platforms supported by OpenWRT Kamikaze

OpenWRT versions for Ubiquiti

- Look at

<http://wiki.openwrt.org/toh/start#ubiquiti>

to find

Ubiquiti Nanostation2, Bullet2 -

<http://downloads.openwrt.org/kamikaze/8.09.2/atheros/openwrt-atheros-ubnt2-squashfs.bin>

Ubiquiti Picostation2 -

<http://downloads.openwrt.org/backfire/10.03.1/atheros/openwrt-atheros-ubnt2-pico2-squashfs.bin>

Flashing a wireless device via Web interface

- Have firmware ready on your computer
- Find out the device's default IP
- Set your own computers IP accordingly (same network, different IP)
- Connect to the device's original web interface (see: Access Point configuration) and find the "Firmware Update" button, or similar, if there is any
- Use this to upload and install new firmware
- **NOW: REMEMBER TO WAIT!
DO NOT PRESS CONTINUE WHEN IT SHOWS!!! DONT!!!
WAIT UNTIL THE LEDS STOP BLINKING!
TYPICALLY 4-6 MINUTES!**
- **MAKE SURE YOU HAVE STABLE POWER WHILE DOING THIS!
IF YOU DO NOT TRUST GRID POWER, USE A BATTERY!**

Flashing a wireless device via tftp

- Connect the device to your computer via ethernet cable.
- Set your computer to be 192.168.1.x, but not 192.168.1.20 (which is the Ubiquiti default IP)
- Power down the device.
- Power it back up, while at the same time holding the reset button, for about 10 seconds.
- By now, the second LED should be flashing green, and the device be pingable on 192.168.1.20
- The device is now waiting for a tftp transfer. It might look like this:

```
sb@sb-laptop:~$ tftp
tftp> connect 192.168.1.20
tftp> binary
tftp> put <firmware-filename>.bin
Sent 2989541 bytes in 13.2 seconds
```

- **Remember to be patient – Dont interrupt it.**
- **MAKE SURE YOU HAVE STABLE POWER WHILE DOING THIS!**
IF YOU DO NOT TRUST GRID POWER, USE A BATTERY!

Working with OpenWRT

- OpenWRT comes with the LuCi webGUI
 - Go to webgui, default is `http://192.168.1.1`
 - Set password!
-
- Alternative: instead of using the GUI, use ssh & command line

If not, set up a root password in the device:

```
telnet 192.168.1.1
```

```
passwd
```

- ssh into the device using the root password you've just entered

Adding software packages to OpenWRT

- Note you will need to have your device connected to the internet, so you may need a switch, and good idea how to connect :)

Hint: you could set the device to receive a DHCP lease, or set a static IP – while you do this, you might wanna connect to the device wirelessly.

Once you are done, you can use the newly configured ethernet connection to connect to it.

But this also means, we need to start coordinating within our group:

192.168.4.2-50
is free for static addresses

192.168.4.49	= Sebastian
192.168.4.48	=
192.168.4.47	=
192.168.4.46	=
192.168.4.45	=
192.168.4.44	=
192.168.4.43	=

Adding software packages to OpenWRT

- Update packages
- Note you will need to have your device connected to the internet, so you may need a switch, and good idea how to connect :)

Hint: you could set the device to receive a DHCP lease, or set a static IP – while you do this, you might wanna connect to the device wirelessly.

Once you are done, you can use the newly configured ethernet connection to connect to it.

- Add packages you need, e.g.

for OLSR:

```
opkg install luci-app-olsr
```

Adding software packages to OpenWRT

- Update packages
- Note you will need to have your device connected to the internet, so you may need a switch, and good idea how to connect :)

Hint: you could set the device to receive a DHCP lease, or set a static IP – while you do this, you might wanna connect to the device wirelessly.

Once you are done, you can use the newly configured ethernet connection to connect to it.

- Add packages you need, e.g.

for OLSR:

```
opkg install luci-app-olsr
```

Configuring OLSR in OpenWRT

- When `olsr` is installed, in the web gui, go to **Services**, and configure OLSR

The screenshot shows the OpenWRT web GUI interface. The browser address bar displays `192.168.4.49/cgi-bin/luci/stok=dbf61694ff0320cc9c88214cb8ae8a8d/admin/services/`. The page title is "OLSR" and the navigation menu includes "Status", "System", "Services", "Network", and "Logout". The "Services" tab is active, and the "OLSR Daemon" configuration page is displayed. The page includes a description of the OLSR daemon and a "General settings" section with the following fields:

Field	Value	Description
Internet protocol	IPv4	IP-version to use. If 6and4 is selected then one olsrd instance is started for each protocol.
FIB metric	flat	FIBMetric controls the metric value of the host-routes OLSRd sets. "flat" means that the metric value is always 2. This is the preferred value because it helps the linux kernel routing to clean up older routes. "correct" uses the hopcount as the metric value. "approx" use the hopcount as the metric value too, but does only update the hopcount if the nexthop changes too. Default is "flat".
Port	698	The port OLSR uses. This should usually stay at the IANA assigned port 698. It can have a value between 1 and 65535.
Main IP	0.0.0.0	Sets the main IP (originator ip) of the router. This IP will NEVER change during the uptime of olsrd. Default is 0.0.0.0, which triggers usage of the IP of the first interface.

http://wiki.ubnt.com/OLSR_on_OpenWrt
<http://wiki.openwrt.org/inbox/mesh.olsr>

- You can also do all this without the GUI! In `/etc/olsrd.conf`

Configuring OLSR in OpenWRT - agreements

- We need to agree on some settings:

IP settings: wired side: e.g.

192.168.4.3x with x = number of group

wireless side: e.g.

10.0.0.x with x = number of group

SSID = LabMesh

channel 1

BSSID = 00:CA:FF:EE:BA:BE

Remember to enable OLSR on the right interface!

Configuring OLSR in OpenWRT - step 1

- > Network > Interfaces

Create a logical interface, configure IP settings, name it as you like. Mine is called "MESH".

The screenshot shows the OpenWRT web interface. At the top, there are navigation tabs: Status, System, Services, Network (selected), and Logout. Below these are sub-tabs for the Network section: Interfaces (selected), Wifi, DHCP and DNS, Hostnames, Static Routes, Firewall, and Diagnostics. The main content area is titled 'Interfaces' and shows a list of interfaces. Two interfaces are visible: 'LAN' (br-lan) and 'MESH' (Ad-Hoc "LabMesh"). The 'MESH' interface is highlighted in blue. To the right of the interface list are 'Actions' buttons: Connect, Stop, Edit, and Delete. At the bottom left, there is a button labeled 'Add new interface...'. The 'MESH' interface details are as follows:

Network	Status	Actions
LAN br-lan	Uptime: 0h 7m 39s MAC Address: 00:15:6D:F7:13:E3 RX: 309.56 KB (2980 Pkts.) TX: 309.13 KB (1651 Pkts.) IPv4: 192.168.4.39/24	Connect Stop Edit Delete
MESH Ad-Hoc "LabMesh"	Uptime: 0h 7m 26s MAC Address: 00:15:6D:F6:13:E3 RX: 166.05 KB (1451 Pkts.) TX: 41.56 KB (247 Pkts.) IPv4: 10.0.0.1/24	Connect Stop Edit Delete

Configuring OLSR in OpenWRT - step 2

- > Network > Wifi

Configure the wireless interface -

in my case (Ubiquiti Picostation) it is a Atheros 802.11bg Wireless Controller (wifi0) -

to use right SSID, Channel, Output power, etc

- **Dont forget to turn it on! It is off per default!**

The screenshot shows the OpenWRT web interface for configuring a wireless network. The top navigation bar includes 'Interfaces', 'Wifi', 'DHCP and DNS', 'Hostnames', 'Static Routes', 'Firewall', and 'Diagnostics'. The current page is titled 'wif0: Ad-Hoc "LabMesh"'. Below the title, there is a section for 'Wireless Network: Ad-Hoc "LabMesh" (ath0)'. A note explains that the 'Device Configuration' section covers physical settings like channel, transmit power, and antenna selection, while 'Interface Configuration' covers per-network settings like encryption and operation mode.

Device Configuration

General Setup | Advanced Settings

Status

Mode: Ad-Hoc | SSID: LabMesh
BSSID: 00:CA:FF:EE:BA:BE | Encryption: None
Channel: 1 (2.412 GHz) | Tx-Power: 20 dBm
Signal: -46 dBm | Noise: -96 dBm
Bit Rate: 26.0 MBit/s | Country: 00

Wireless network is enabled Disable

Channel: 1 (2.412 GHz)

Transmit Power: 20 dBm (100 mW)

Interface Configuration

General Setup | Wireless Security | MAC-Filter | Advanced Settings

ESSID: LabMesh

Mode: Ad-Hoc

BSSID: 00:CA:FF:EE:BA:BE


Network: lan: mesh:


Configuring OLSR in OpenWRT - step 3




- > Services > OLSR

Most importantly, activate OLSR on the right interface - in our case, the wireless mesh interface we have created above.

Interfaces

Enable	Network	Mode	Hello	TC	MID	HNA
<input checked="" type="checkbox"/>	mesh: 	mesh	5.0s / 40.0s	2.0s / 256.0s	18.0s / 324.0s	18.0s / 108.0s

 Add

 Reset  Save  Save & Apply

Configuring OLSR in OpenWRT - step 4

- > Services > OLSR > HNA

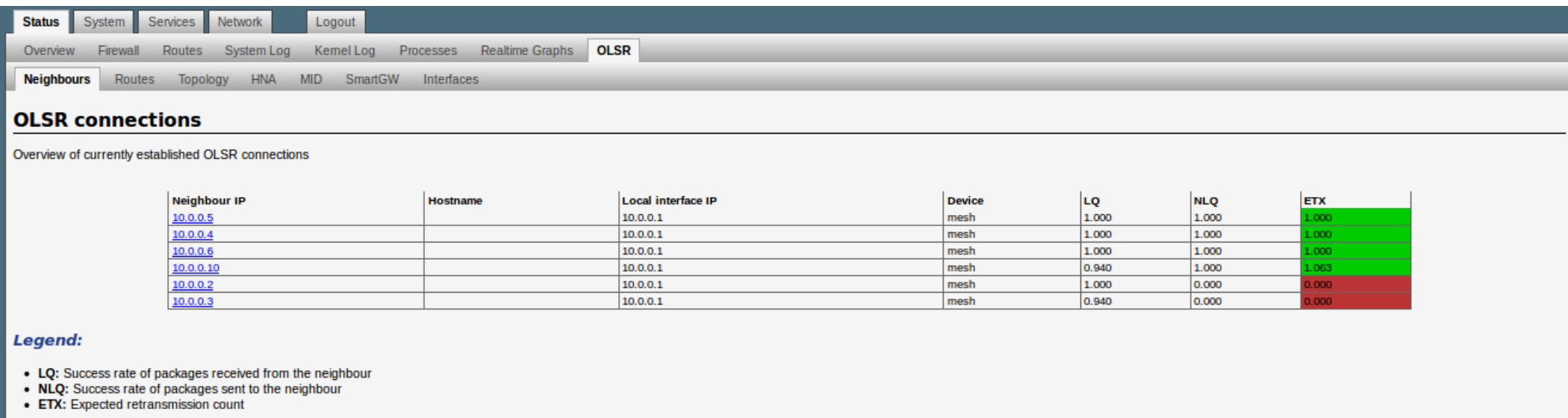
If our node offers access to a network (e.g. an internet uplink), we need to announce that.

The screenshot shows the OpenWRT web interface. At the top, there is a navigation bar with tabs for Status, System, Services, Network, and Logout. Below this, the OLSR configuration page is displayed, with sub-tabs for Plugins, HNA Announcements, and Display. The main heading is "OLSR - HNA-Announcements". Below the heading, there is a descriptive text: "Hosts in a OLSR routed network can announce connectivity to external networks using HNA messages." Underneath, there is a section titled "Hna4" with a note: "Both values must use the dotted decimal notation." This section contains two input fields: "Network address" with the value "192.168.4.0" and "Netmask" with the value "255.255.255.0". There is an "Add" button below the "Network address" field. At the bottom right of the form, there are three buttons: "Reset", "Save", and "Save & Apply".

Configuring OLSR in OpenWRT - step 5

- > Status > OLSR

Check our success ... check neighbours, LQ, ETX ...



The screenshot shows the OpenWRT web interface for OLSR connections. The navigation menu includes Status, System, Services, Network, and Logout. The OLSR section is active, showing a table of connections with columns for Neighbour IP, Hostname, Local interface IP, Device, LQ, NLQ, and ETX. A legend explains the metrics: LQ (Success rate of packages received), NLQ (Success rate of packages sent), and ETX (Expected retransmission count).

Neighbour IP	Hostname	Local interface IP	Device	LQ	NLQ	ETX
10.0.0.5		10.0.0.1	mesh	1.000	1.000	1.000
10.0.0.4		10.0.0.1	mesh	1.000	1.000	1.000
10.0.0.6		10.0.0.1	mesh	1.000	1.000	1.000
10.0.0.10		10.0.0.1	mesh	0.940	1.000	1.063
10.0.0.2		10.0.0.1	mesh	1.000	0.000	0.000
10.0.0.3		10.0.0.1	mesh	0.940	0.000	0.000

Legend:

- **LQ:** Success rate of packages received from the neighbour
- **NLQ:** Success rate of packages sent to the neighbour
- **ETX:** Expected retransmission count

Old slides for reference:

Freifunk, debricking,
olsrd on linux laptop,
etc

Lab: Configuring a Freifunk Mesh Mesh planning

- **Planning is about people! Consider the social dynamics, ownership, support, ...**
- **Map / Site Survey**
- **Select network topology**
- **Channel allocation (mesh, backbone, local hotspots)**
- **IP address allocation**
- **Draw the network diagram**

Lab: Freifunk Firmware on Linksys WRT54G

Debricking a WRT54G

- power down the WRT54GL
- Example of a tftp transmission:
on a linux command line, do
[root@samsarix /]# ifconfig eth1 192.168.1.99 up
[root@samsarix /]# tftp -v -m binary 192.168.1.1
mode set to octet
Connected to 192.168.1.1 (192.168.1.1), port 69
tftp> put openwrt-g-freifunk-1.6.25-en.bin
<NOW POWER UP THE WRT54GL! and if you are lucky... it will say>
putting openwrt-g-freifunk-1.6.25-en.bin to 192.168.1.1:openwrt-g-freifunk-1.6.25-en.bin [octet]
Sent 1303552 bytes in 7.9 seconds [1326539 bit/s]
tftp>
- If debricking via tftp fails, you will have to open the device and do some pin magic – read more here:

http://www.notsecure.us/debrick_wrt54g_without_void_warrenty.html

[http://www.ranvik.net/prosjekter-](http://www.ranvik.net/prosjekter-privat/jtag_for_wrt54g_og_wrt54gs/HairyDairyMaid_WRT54G_v22.pdf)

[privat/jtag_for_wrt54g_og_wrt54gs/HairyDairyMaid_WRT54G_v22.pdf](http://www.ranvik.net/prosjekter-privat/jtag_for_wrt54g_og_wrt54gs/HairyDairyMaid_WRT54G_v22.pdf)

<http://www.freewebs.com/wrt54grevival/wrt54grevial.htm.html>

Lab: Running olsrd / batmand on a laptop

- Olsr demons are available for Linux, Mac OS X, BSD, Windows: <http://www.olsr.org/?q=download>
- Also: Ubuntu plugins, .debs, Nokia, iPhone
- Linux: install via the normal make routine
Do this in the lib directories too to activate libs!
- Windows: GUI OLSR-Switch might be out of date!
- Biggest obstacle in all of this: card and driver issues (ad-hoc mode often badly buggy)

Lab: Running olsrd / batmand on a laptop olsrd on Linux

- Download sources, build

```
(in olsr dir)
```

```
# make
```

```
# make install
```

```
# cd lib
```

```
# make install
```

```
(...)
```


Lab: Running olsrd / batmand on a laptop olsrd on Linux

- Prepare settings and start the demon
- example session:

```
# killall NetworkManager
# killall NetworkManagerDispatcher
# killall olsrd
# ifconfig ath0 172.31.1.19 netmask
255.255.255.0
# iwconfig ath0 essid roadshow-mesh mode ad-hoc
channel 1
# olsrd -i ath0 -d 5
# netstat -nr
```