# ICTP 2012
# Mesh Networking Lab

# Part 1

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.

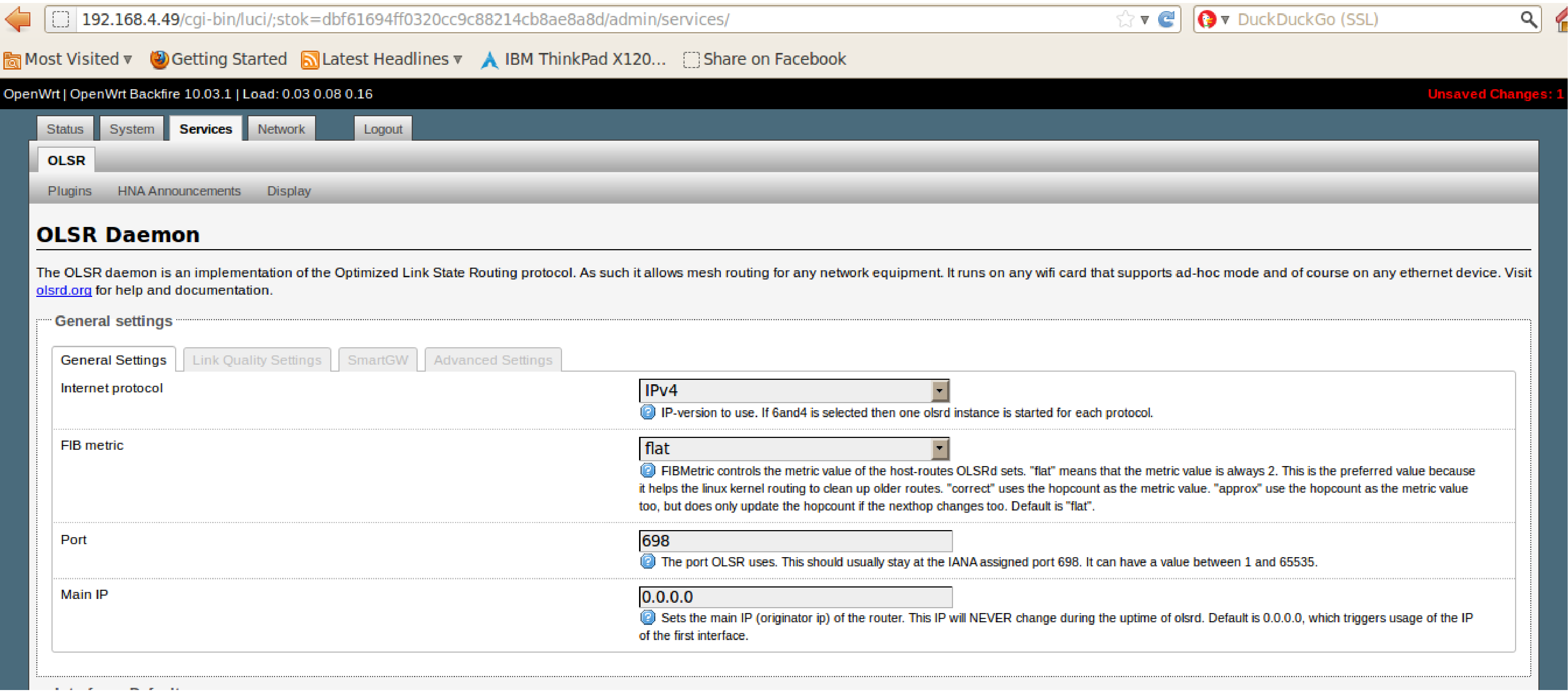
- Add packages you need, e.g.

  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



OpenWrt | OpenWrt Backfire 10.03.1 | Load: 0.03 0.08 0.16

Unsaved Changes: 1

## OLSR Daemon

The OLSR daemon is an implementation of the Optimized Link State Routing protocol. As such it allows mesh routing for any network equipment. It runs on any wifi card that supports ad-hoc mode and of course on any ethernet device. Visit olsrd.org for help and documentation.

General settings

| | |
|---|---|
| Internet protocol | IPv4<br>IP-version to use. If 6and4 is selected then one olsrd instance is started for each protocol. |
| FIB metric | flat<br>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<br>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<br>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

-

## Configuring OLSR in OpenWRT

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

IP settings: need fixed IP … again we need to agree!

192.168.4.40 Sebastian

SSID = LabMesh

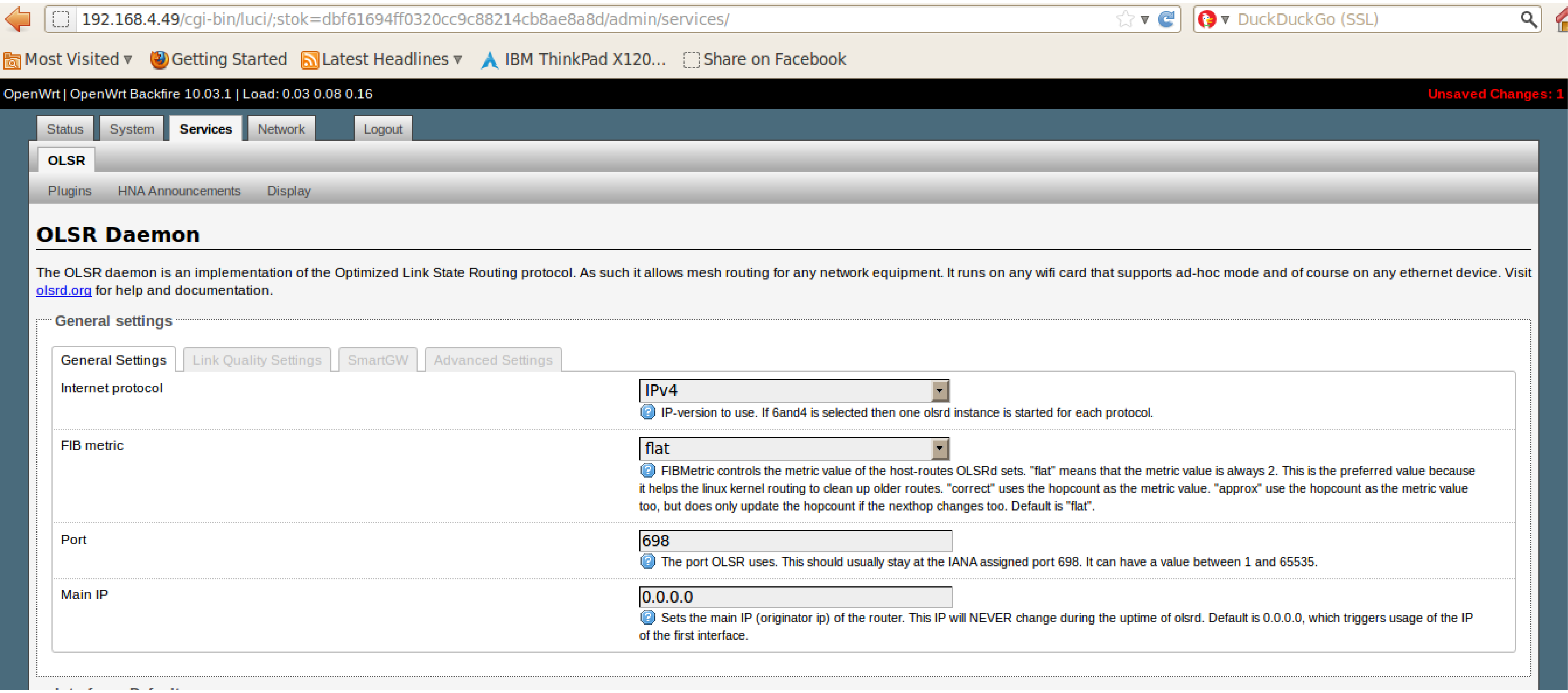
channel 1

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

Remember to enable OLSR on the right interface

## Configuring OLSR in OpenWRT

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



192.168.4.49/cgi-bin/luci/;stok=dbf61694ff0320cc9c88214cb8ae8a8d/admin/services/

OpenWrt | OpenWrt Backfire 10.03.1 | Load: 0.03 0.08 0.16

Unsaved Changes: 1

OLSR Daemon

The OLSR daemon is an implementation of the Optimized Link State Routing protocol. As such it allows mesh routing for any network equipment. It runs on any wifi card that supports ad-hoc mode and of course on any ethernet device. Visit olsrd.org for help and documentation.

General settings

| | |
|---|---|
| Internet protocol | IPv4<br>IP-version to use. If 6and4 is selected then one olsrd instance is started for each protocol. |
| FIB metric | flat<br>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<br>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<br>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

-

## Old slides for reference:

Freifunk, debricking,
olsrd on linux laptop,
etc

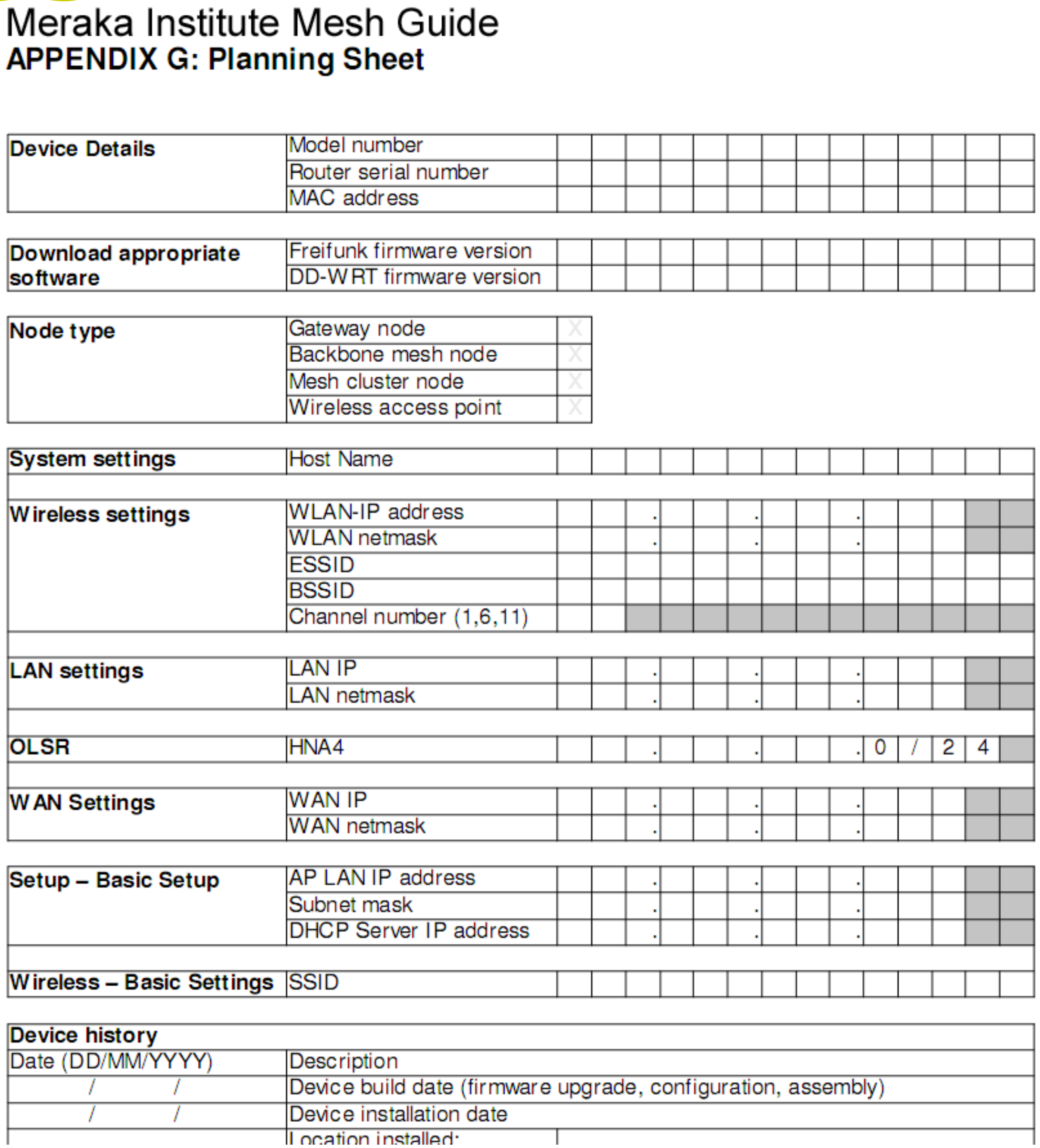
# Lab: Configuring a Freifunk Mesh

## 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: Configuring a Freifunk Mesh Guide

- **For the planning for each device: e.g. Meraka Mesh Guide Form**



Meraka Institute Mesh Guide
**APPENDIX G: Planning Sheet**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Device Details** | Model number | | | | | | | | | | | | | | |
| | Router serial number | | | | | | | | | | | | | | |
| | MAC address | | | | | | | | | | | | | | |
| **Download appropriate software** | Freifunk firmware version | | | | | | | | | | | | | | |
| | DD-WRT firmware version | | | | | | | | | | | | | | |
| **Node type** | Gateway node | X | | | | | | | | | | | | | |
| | Backbone mesh node | X | | | | | | | | | | | | | |
| | Mesh cluster node | X | | | | | | | | | | | | | |
| | Wireless access point | X | | | | | | | | | | | | | |
| **System settings** | Host Name | | | | | | | | | | | | | | |
| **Wireless settings** | WLAN-IP address | | | . | | | . | | | . | | | | ▨ | ▨ |
| | WLAN netmask | | | . | | | . | | | . | | | | ▨ | ▨ |
| | ESSID | | | | | | | | | | | | | | |
| | BSSID | | | | | | | | | | | | | | |
| | Channel number (1,6,11) | | | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ |
| **LAN settings** | LAN IP | | | . | | | . | | | . | | | | ▨ | ▨ |
| | LAN netmask | | | . | | | . | | | . | | | | ▨ | ▨ |
| **OLSR** | HNA4 | | | . | | | . | | | . | 0 | / | 2 | 4 | |
| **WAN Settings** | WAN IP | | | . | | | . | | | . | | | | ▨ | ▨ |
| | WAN netmask | | | . | | | . | | | . | | | | ▨ | ▨ |
| **Setup – Basic Setup** | AP LAN IP address | | | . | | | . | | | . | | | | ▨ | ▨ |
| | Subnet mask | | | . | | | . | | | . | | | | ▨ | ▨ |
| | DHCP Server IP address | | | . | | | . | | | . | | | | ▨ | ▨ |
| **Wireless – Basic Settings** | SSID | | | | | | | | | | | | | | |

| **Device history** | |
|---|---|
| Date (DD/MM/YYYY) | Description |
| / / | Device build date (firmware upgrade, configuration, assembly) |
| / / | Device installation date |
| | Location installed: |

## 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-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
```