SAPIENZA
UNIVERSITÀ DI ROMA

Network Infrastructures

Wireless Community Mesh Networks

with a practical node configuration

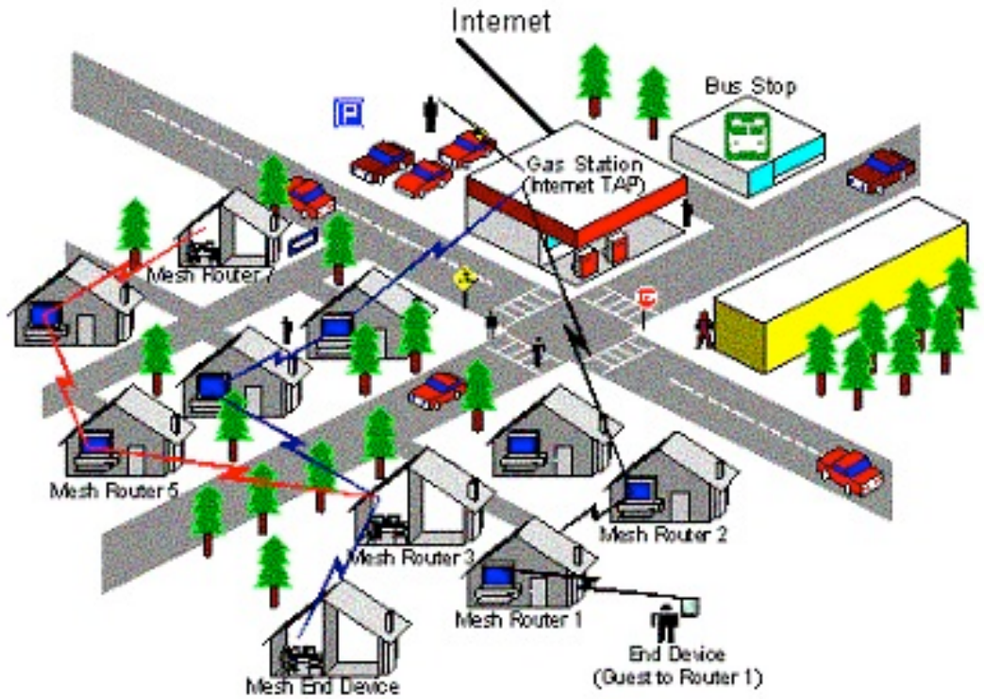
Fabio Petroni
Raffaele Petteruti
Mara Sorella

NINUX.ORG

Wireless Community Networks

•A Wireless community network is an organization of people attempting to provide a network infrastructure based on **WiFi** (802.11abgn) technology

•Many community networks cover metropolitan areas, in general covered areas can correspond to cities, districts but even entire states



freifunk.net

FUNKFEUER
FREE NET

guifi·net (((ComunsSenseFils)))
Talla els fils que et lliguen.
Una xarxa lliure per a tots
és possible.

SEATTLE WIRELESS.NET

NINUX.ORG

Ninux.org is a wireless community network born in Rome in 2001.

It mainly spreads over **Rome** and its province but small ninux communities are also starting to grow up in **Pisa** (Tuscany), **Vittoria** and **Mistretta** (Sicily) cities.

- **~80** active nodes
- **~170** potential nodes

Services

file sharing
VoIP asterisk exchange
VPN
Internet access
Bonjour services
...

TuscoloMesh

Wireless Community Networks (II)

Properties

- Neutrality

  no single subject owns the whole infrastructure: **everyone is responsible for his own node**

  everyone is both **user** and **provider**

- Common Service Level Agreement

  no service contract, **no guarantees** (i.e. minimum bandwith amount)

  Wireless Common Licence
  PicoPeering Agreement

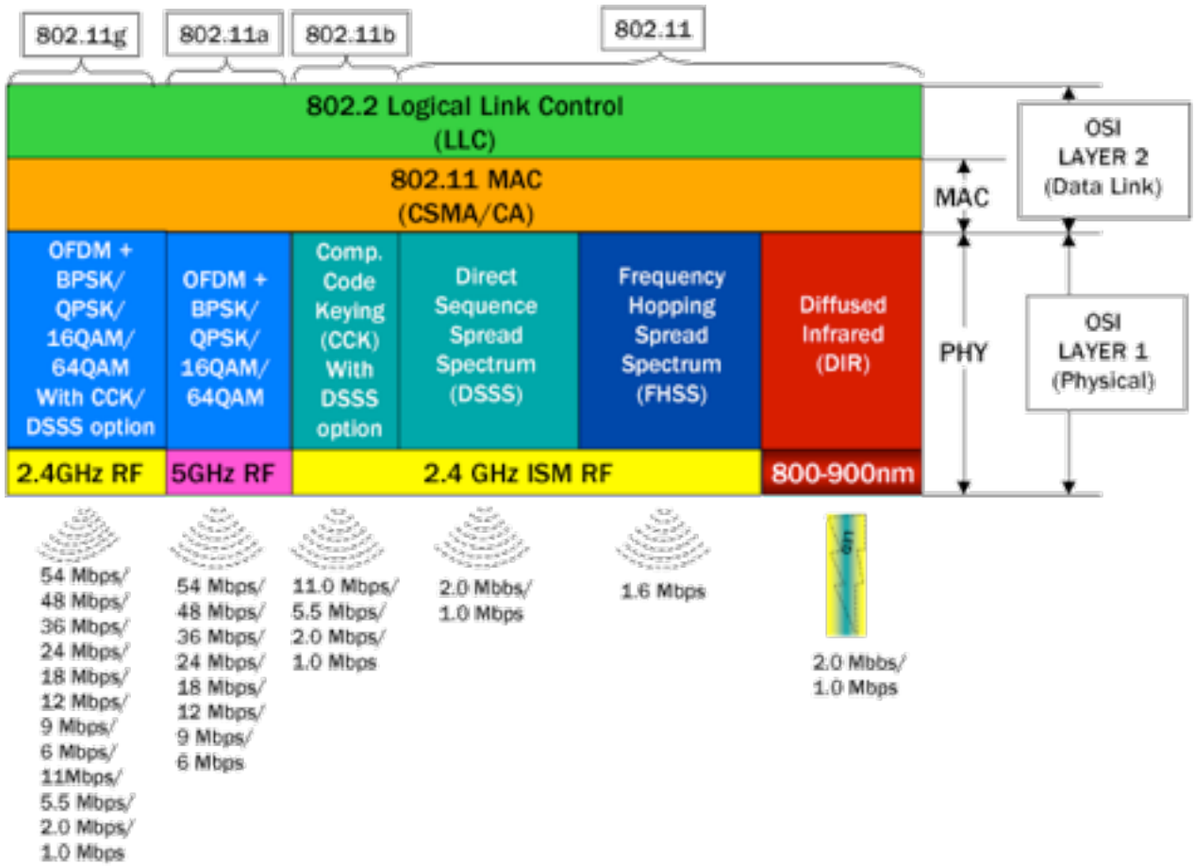
- Low manteinance costs

- Communication locality and Symmetry

  mesh structure of the core network determines that when two geographically near nodes start a communication, their packets follow **geographically coherent paths**

  symmetric links: users provide services too (YouTube, MySpace..)

The WiFi Standard: 802.11



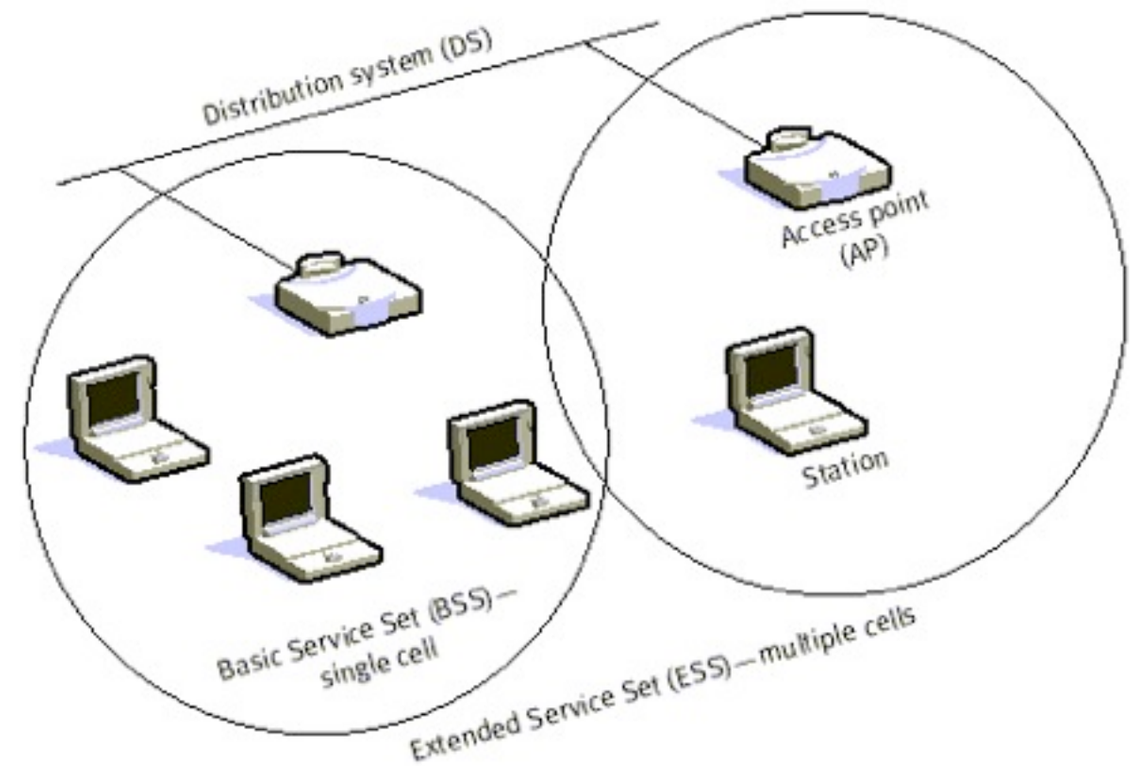
The IEEE802.11 standard describes both **physical and medium access** layers for wireless communication among local and metropolitan area networks using unlicensed bands of the electromagnetic spectrum.

First version of the standard provided 1-2 Mbps rates using two complementary modulation techniques for physical layer: **Direct Sequence Spread Spectrum (DSSS)** and **Frequency Hopping Spread Spectrum (FHSS)**.

Later revisions yelded to a net achievable throughput **up to 54Mbps** (802.11g), making use of the **Orthogonal Frequency Division Multiplexing** which allows higher transmission rates to the detriment of noise resistance and a consequent shortening of the actual communication radius.
Last revision is dated 2009 and introduced the standard **802.11n** which provides rates up to **300Mbps** and operates **both in 2.4 an 5 GHz frequency band**.

802.11 MAC Layer (I)

802.11 Medium Access Control has two basic modes of operation called **Infrastructure** and **Ad-hoc**.



Infrastructure mode:

In this configuration every station user connects to an access point via a wireless link.

The set-up formed by the access point and the stations located within its coverage area are called the **Basic Service Set**, or **BSS**. Each BSS is identified by a **BSSID**, a 6-byte (48-bit) identifier.

In infrastructure mode, the BSSID corresponds to the access point's MAC address which announces his presence by periodically broadcasting **beacon frames** in the covered area

It is possible to link several access points together using a connection called a **Distribution System** (**DS**) in order to form an **Extended Service Set** (**ESS**), identified by an alphanumeric sequence of up to 32 characters, the **ESSID**.

The connection between different BSS can be made either with a wired or wireless connection, it allows the transmission of packets **between different stations even if they are associated to different APs**

The **Wireless Distribution System** (**WDS**) is trying to provide a standard both for AP to AP connectivity via a wireless link and for allowing the **roaming** of stations between different BSS in a transparent way

802.11 MAC Layer (II)

Ad-hoc mode:

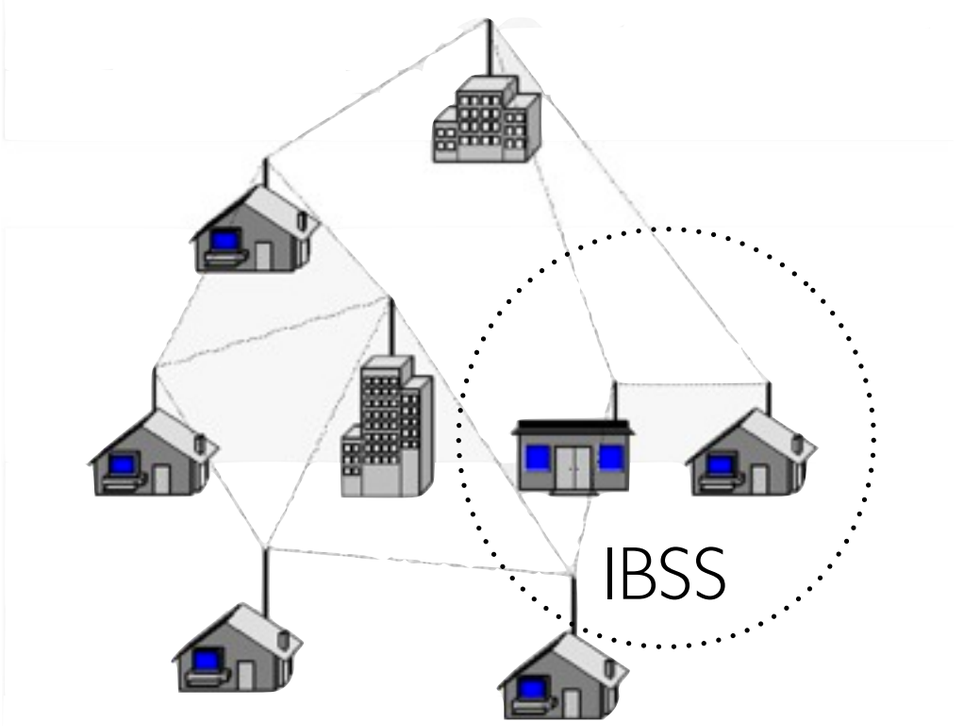
In ad hoc mode, wireless nodes connect one to another in order to form a **peer-to-peer** network, i.e. a network in which every node acts as both a client and as access point at the same time.

The set-up formed by the stations is called the **Independent Basic Service Set**, or **IBSS**.

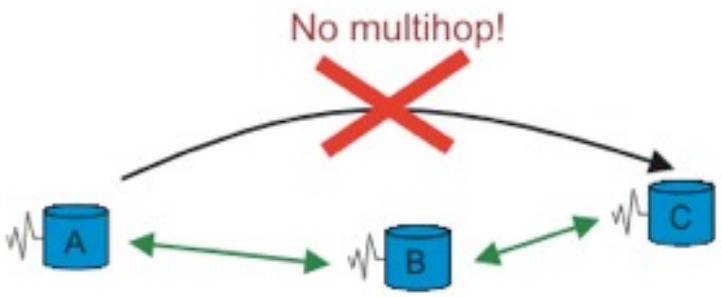
An IBSS is a wireless network which has at least two stations and uses no access point. The IBSS therefore forms a temporary network that is identified by an **SSID**, just like an ESSID in infrastructure mode.

In an ad hoc network, the range of the independent BSS is determined by each station's range.

That means that if two nodes on the network are outside each other's range, *they will not be able to communicate*, even if they can "see" other intermediate stations.



This comes from the fact that in the standard-defined 802.11 ad-hoc mode there is **no multi-hop support**

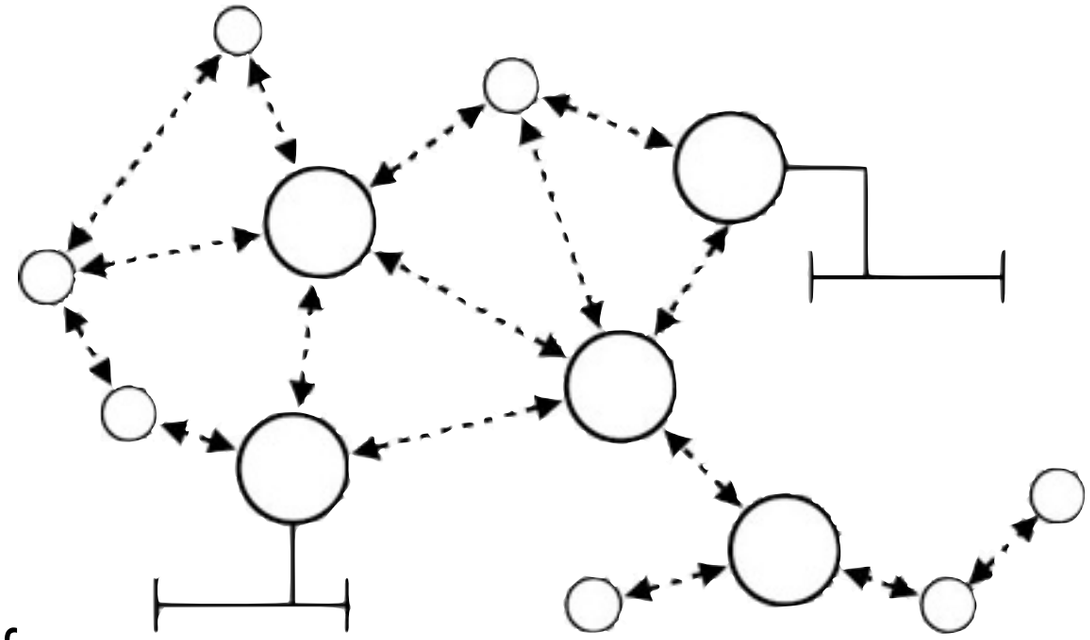


From Ad-hoc to Mesh

Two nodes not directly connected at Layer 2 can communicate by using Layer 3 routing protocols that must:

- **Discover a path** from source to destination
- **Maintain that path** (e.g., if an intermediate node goes down and breaks the path)
- Define mechanisms to exchange routing information

A wireless **ad-hoc mesh network** is a network infrastructure in which nodes are interconnected in a way form a **mesh**, and are able to send messages to other nodes either by using direct links or having them being relayed by intermediate ones.



With the aid of dynamic routing protocols, mesh networks are flexible:

self configuring

self healing

spontaneous

Every node participates in the routing process thus every decision is based upon specific connectivity requirements of the whole structure

OLSR

Optimized Link State Protocol, defined in RFC 3626, originally devised for MANETs

Both **link-state** and **proactive**

Very widespread also in wireless mesh networks, as they are easy to parallelize with mobile context due to the tipical problems that affect links stability and quality:

- weather conditions
- temporary obstacles
- power faults

Node addressing

Uses both **IPv4** and **IPv6**

Each node can have multiple interfaces, one *main address* is choosed, others are referred as *aliases*

routing process based on **shortest-path** algorithm

metric: **hop count**

use of TTLs

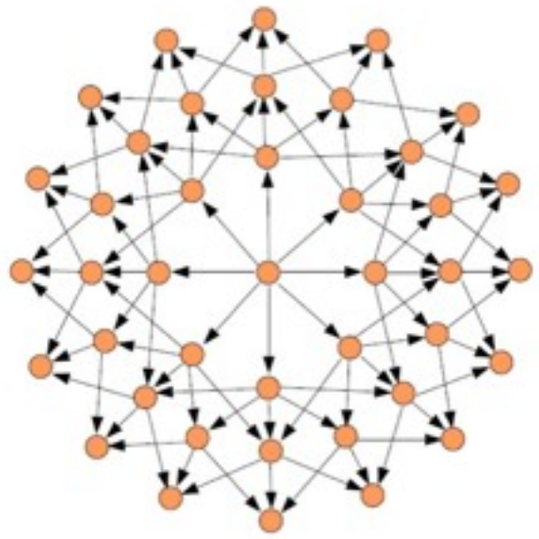
An optimization to address network flooding issue: the **Multi-Point-Relaying tecnique** (**MPR**)

OLSR - Multipoint Relaying

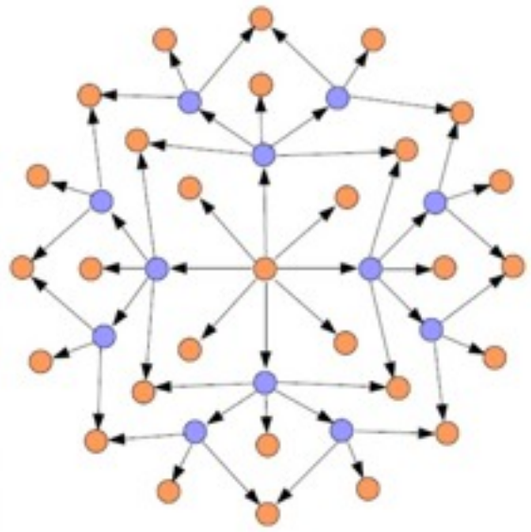
The concept of multipoint relaying is to reduce the number of duplicate retransmissions while forwarding a broadcast packet.

MPR restricts the set of nodes retransmitting a packet from all nodes, to a subset of all nodes

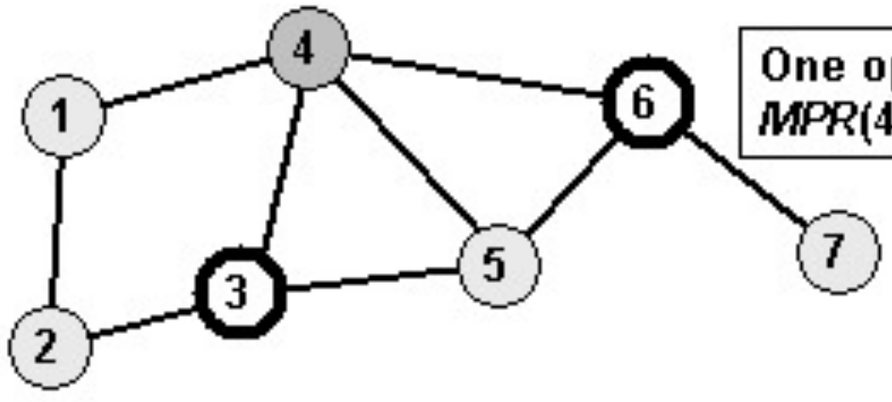
- Each node N in the network selects a set of neighbor nodes as multipoint relays, **MPR(N)**, that retransmit control packets from N
  - Neighbors not in MPR(N) process control packets from N, but they do not forward the packets
- MPR(N) is selected such that **all two-hop neighbors of N are covered by (one-hop neighbors) of MPR(N)**



flooding



MPR selective flooding

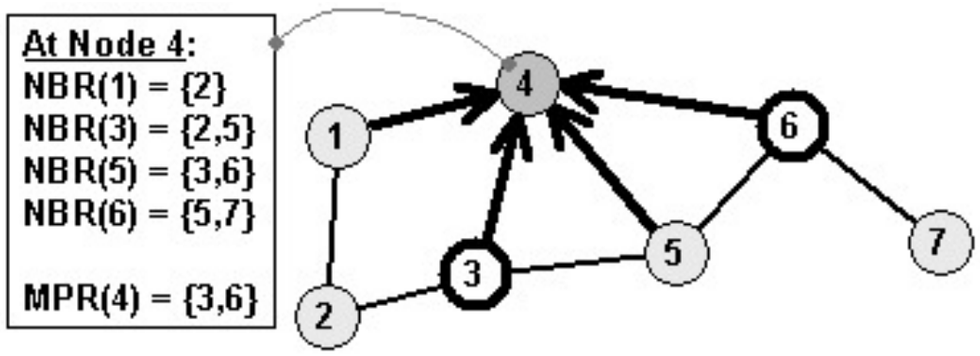


One optimal set for Node 4:
MPR(4) = { 3, 6 }

OLSR - Messages (I)

Control traffic uses UDP packets on port 698

HELLO



- Each node uses HELLO messages to **determine its MPR set**
- All nodes periodically broadcast HELLO messages to their one-hop neighbors (bidirectional links)
- Using the neighbor list in received HELLO messages, nodes can determine their two-hop neighborhood and near optimal MPR set

Topology Control (TC)

- Nodes send **topology information** in Topology Control (TC) messages
  - List of advertised neighbors (link information)
- A node **generates** TC messages only for those neighbors who choosed it as a MPR
  - Only MPR nodes generate TC messages
  - Not all links are advertised
- A node processes all received TC messages, but only **forwards** TC messages if the sender is in its MPR selector set
  - Only MPR nodes propagate TC messages

OLSR - Messages (II)

Other message types

- **MID** (**Multiple Interface Declaration**)
  used by nodes to declare the existence and the address of secondary interfaces than the radio one
- **HNA** (**Host and Network Association**)
  used by nodes to announce the presence of a gateway towards a specific subnet

OLSR Community Extensions

Many wireless network communities around the world introduced different **modifications** into OLSR to better fit issues arising from their specific context

The main problem that OLSR (as from its original RFC version) causes comes from the **hop count metric** used in the routing process

In particular it doesn't take into account **physical disomogeneity** of links:

- cover radius and signal strength (dB)
- number of links involving a node (throughput fractioning)

Different metrics were proposed in order to find out a solution:

per-hop RTT

per-hop Packet Pair Delay

...

ETX

OLSR Community Extensions - ETX

An acronym for *Expected Transmission Count*

Original idea

packet loss extimation using extra UDP packets

Freifunk WCN had the intuition to monitor the **existing** HELLO packets transmissions instead of introducing further control overhead

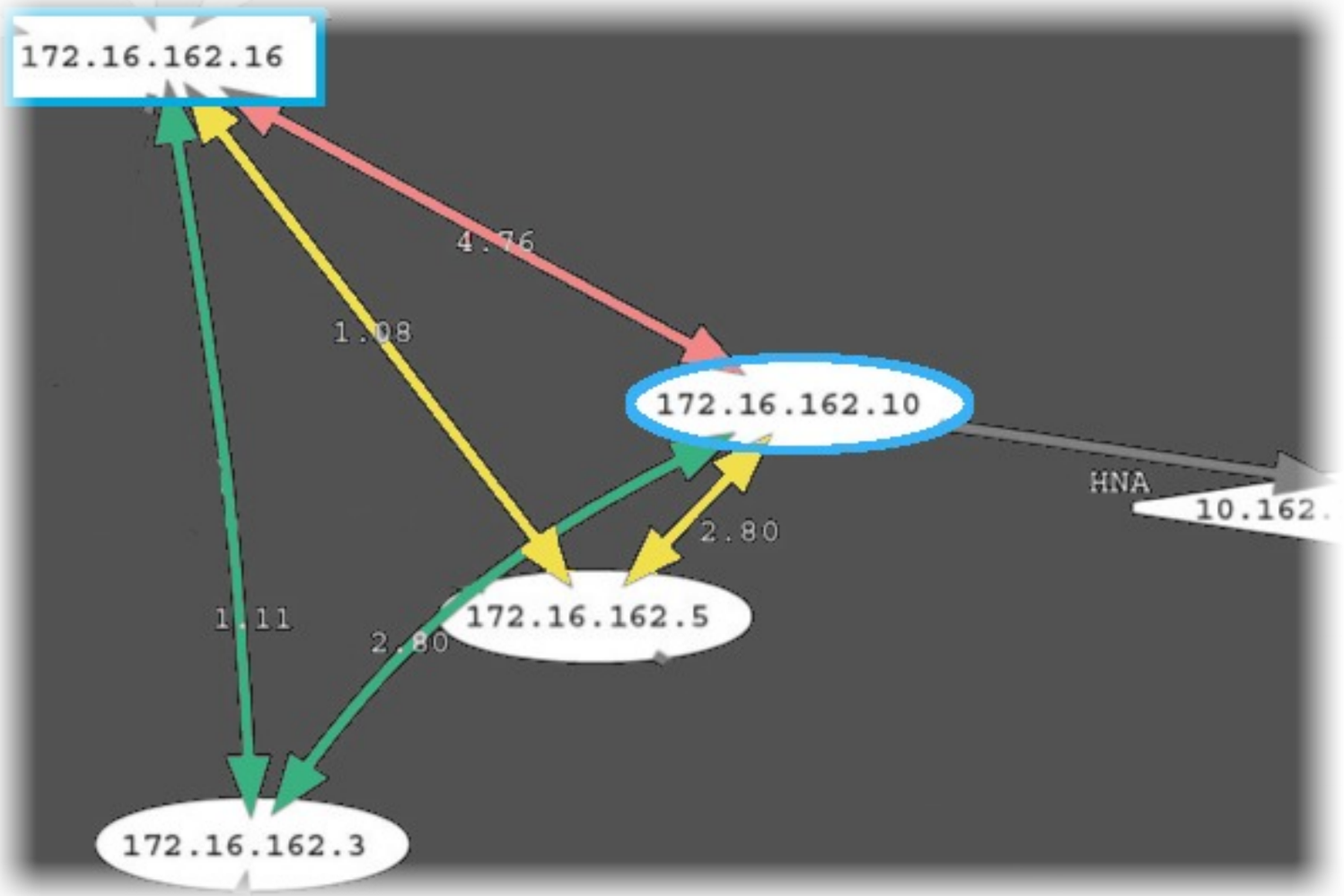
$$ETX = \frac{1}{(LQ \cdot NLQ)}$$

LQ=Link Quality [0,1]
NLQ=Neighbour Link Quality [0,1]

The ETX of a link is the estimated number of transmissions required to successfully send a packet

a perfect link has ETX=1

ETX metric: an example



In this example (took from the actual ninux.org topology) the node with main address *172.16.162.16* can choose **3 different paths** to reach the node *172.16.162.10*

The path with the **lowest ETX sum** will be chosen

- 172.16.162.16 – 171.16.162.10 with ETX= 4.76
- 172.16.162.16 – 172.16.162.3 – 172.16.162.10 with ETX= 1.08 + 2.80 = 3.88
- 172.16.162.16 – 172.16.162.5 – 172.16.162.10 with ETX= 1.11 + 2.80 = 3.91

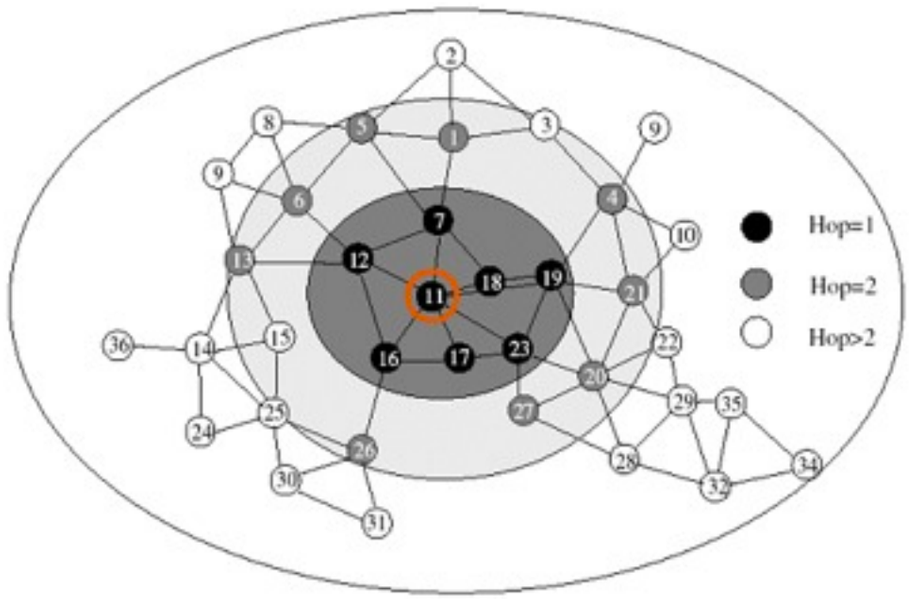
OLSR Community Extensions - Fish Eye Algorithm

Another OLSR optimization devised to reduce OLSR overhead in routing process is the Fish Eye Algorithm.

This simple technique is based upon the **reduction of the scope** (TLL) of some TC messages.

This leads to the fact that every node has a **precise** (up to date) **view of its neighborhood** and this view becomes more and more **approximated** when moving farther

This is done because there is no need for any node to have a precise information about the entire topology as its specific work is only making a **local estimation**



Results

Less overhead

Problems in big networks: can determine routing loops

olsrd.conf in practice

ETX metric: the more close to 1, the more this node has chance to be choosed for a path

```
# 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
LinkQualityLevel 2
```

TC Redundancy: specifies how much neighbour info should be sent in TC messages

```
# 0 - only send MPR selectors
# 1 - send MPR selectors and MPRs
# 2 - send all neighbors
TcRedundancy 2
```

MPR Coverage: how many MPRs a node should try select to reach every 2 hop neighbour

```
# Can be set to any integer >0
MprCoverage 3
```

Willingness: expressess the willingness for a node to be chosen as MPR

```
# 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 3
```

Devices

Devices

2.4 Ghz

2.4 Ghz

5 Ghz

NINUX.ORG

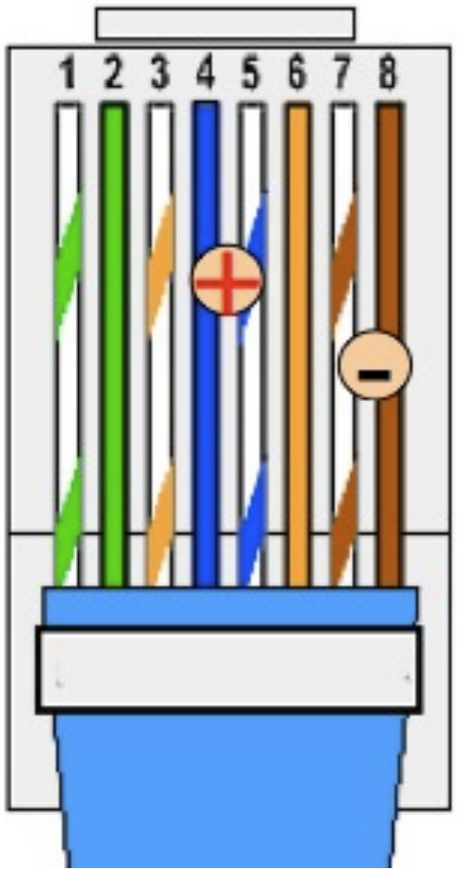
Power over Ethernet

Power over Ethernet

- Is a technology that describes a system to pass electrical power safely, along with data, on Ethernet cabling.

Power over Ethernet

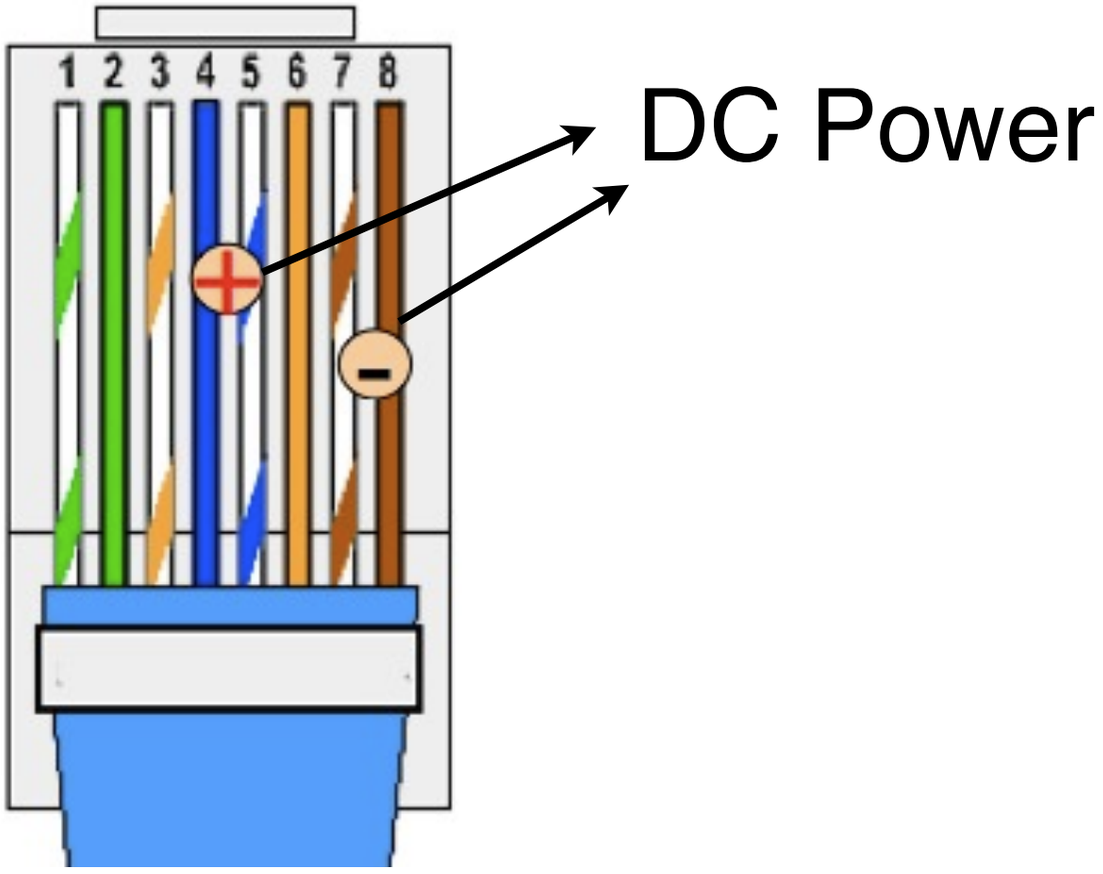
- Is a technology that describes a system to pass electrical power safely, along with data, on Ethernet cabling.



100 Mbit plug

Power over Ethernet

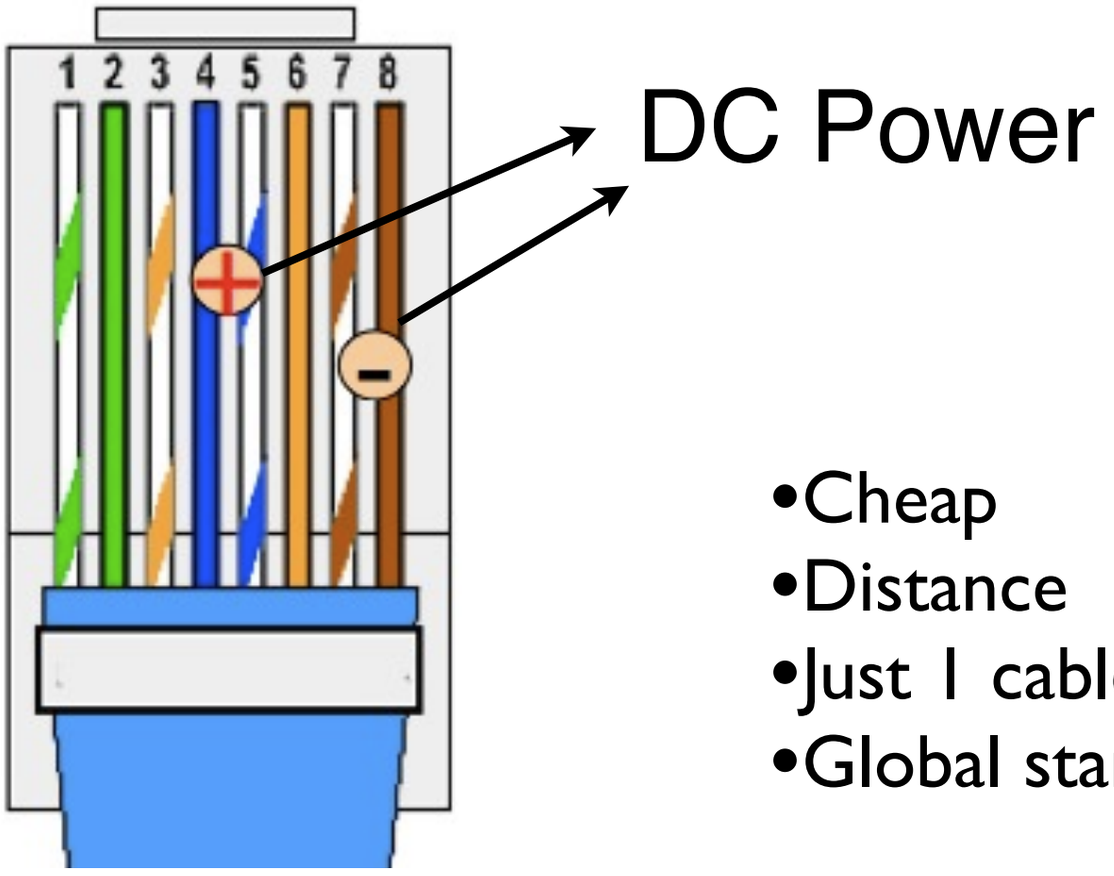
- Is a technology that describes a system to pass electrical power safely, along with data, on Ethernet cabling.



100 Mbit plug

Power over Ethernet

- Is a technology that describes a system to pass electrical power safely, along with data, on Ethernet cabling.

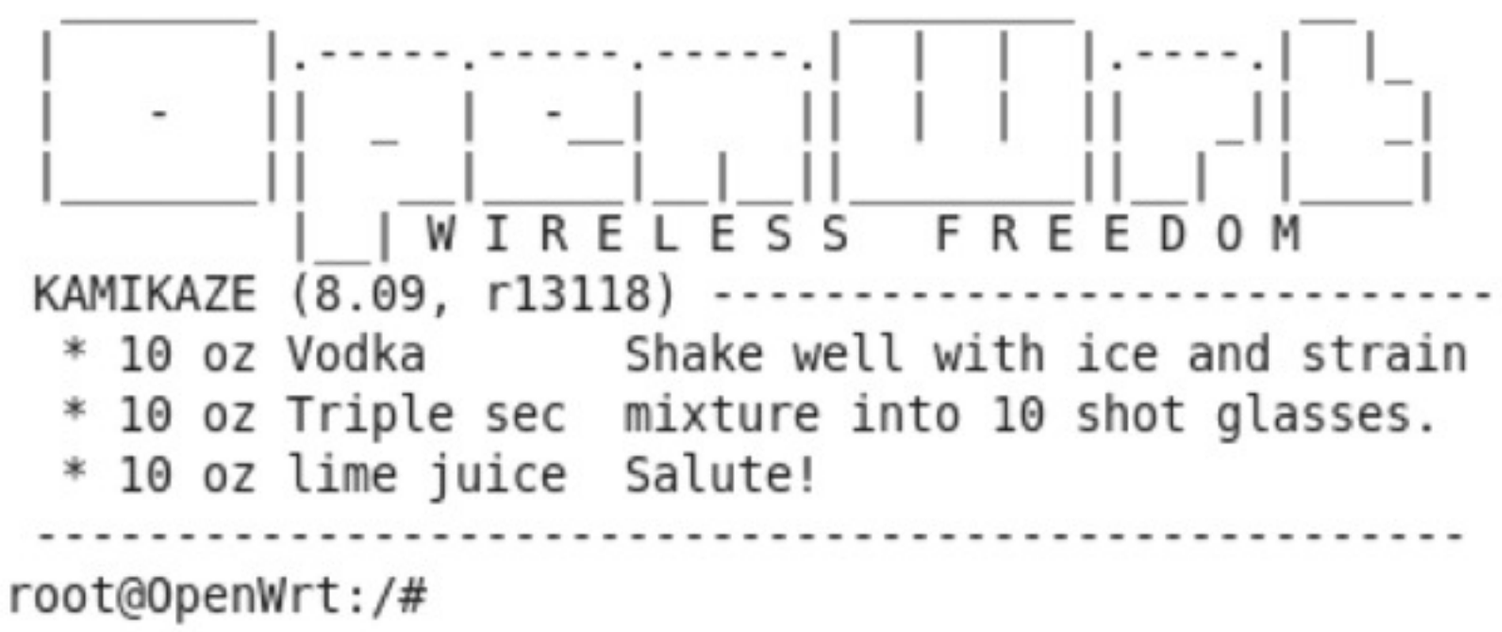


100 Mbit plug

Why?

- Cheap
- Distance
- Just 1 cable for data and power
- Global standard

OpenWRT

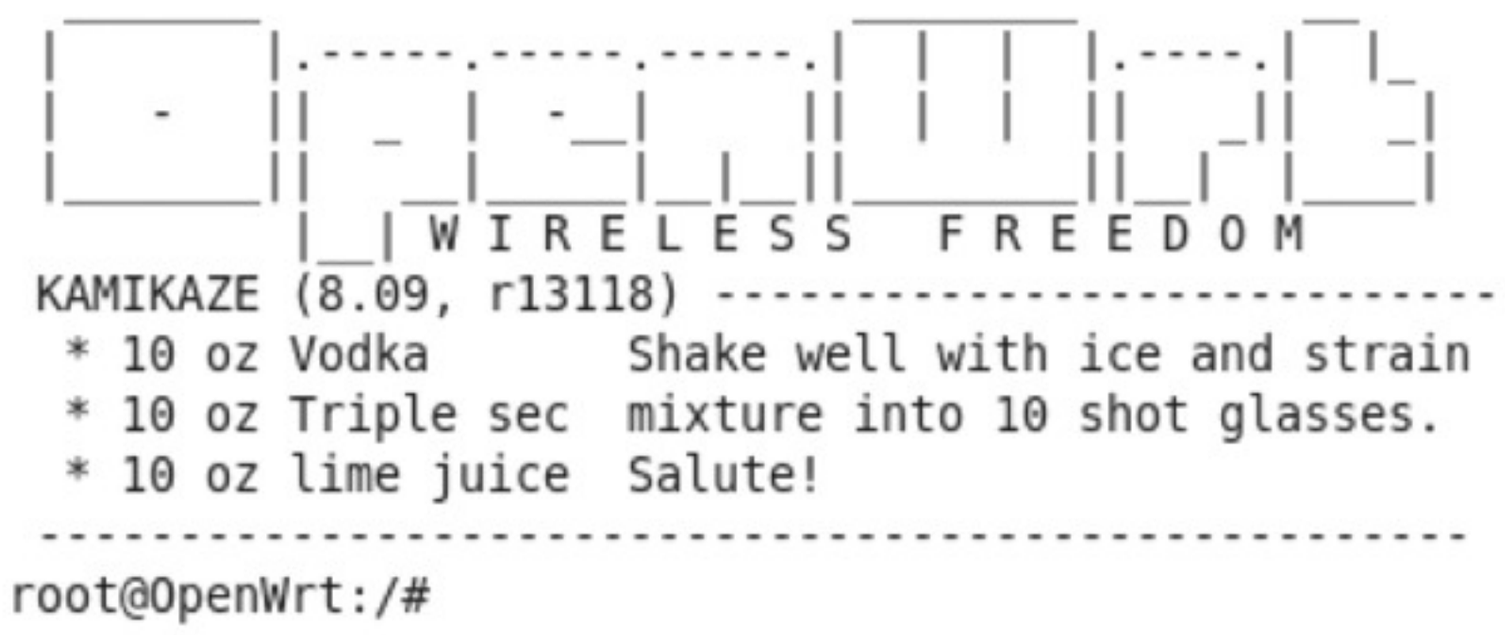

W I R E L E S S F R E E D O M
KAMIKAZE (8.09, r13118)
* 10 oz Vodka Shake well with ice and strain
* 10 oz Triple sec mixture into 10 shot glasses.
* 10 oz lime juice Salute!
root@OpenWrt:/#

OpenWRT

```
 _______                     ________        __
|       |.-----.-----.-----.|  |  |  |.----.|  |_
|   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
|_______||   __|_____|__|__||________||__|  |____|
         |__| W I R E L E S S   F R E E D O M
KAMIKAZE (8.09, r13118) ------------------------------
 * 10 oz Vodka       Shake well with ice and strain
 * 10 oz Triple sec  mixture into 10 shot glasses.
 * 10 oz lime juice  Salute!
 ------------------------------------------------------
root@OpenWrt:/#
```

Features:

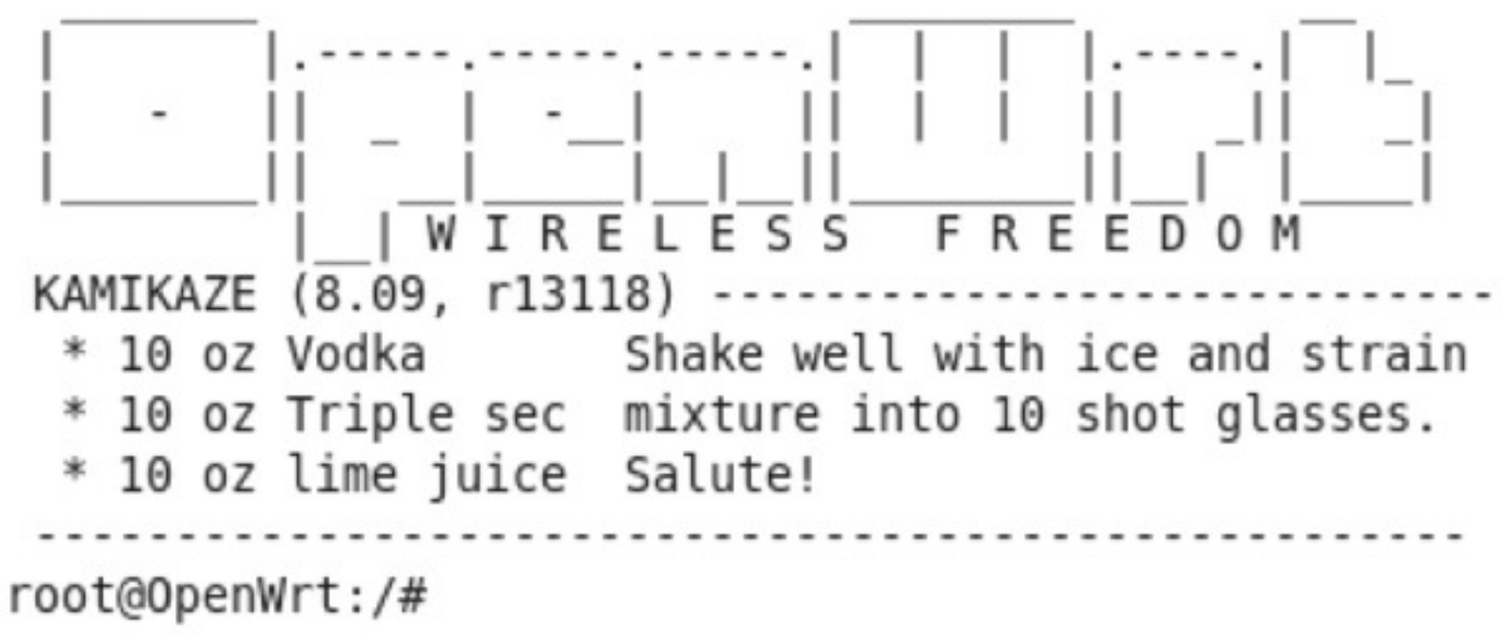
OpenWRT



Features:

- Opensource

OpenWRT



Features:

- Opensource
- Ready for Olsr

OpenWRT

```
 _______                     ________        __
|       |.-----.-----.-----.|  |  |  |.----.|  |_
|   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
|_______||   __|_____|__|__||________||__|  |____|
         |__| W I R E L E S S   F R E E D O M
KAMIKAZE (8.09, r13118) ------------------------------
 * 10 oz Vodka       Shake well with ice and strain
 * 10 oz Triple sec  mixture into 10 shot glasses.
 * 10 oz lime juice  Salute!
 ---------------------------------------------------
root@OpenWrt:/#
```

Features:

- Opensource
- Ready for Olsr
- Easy to install

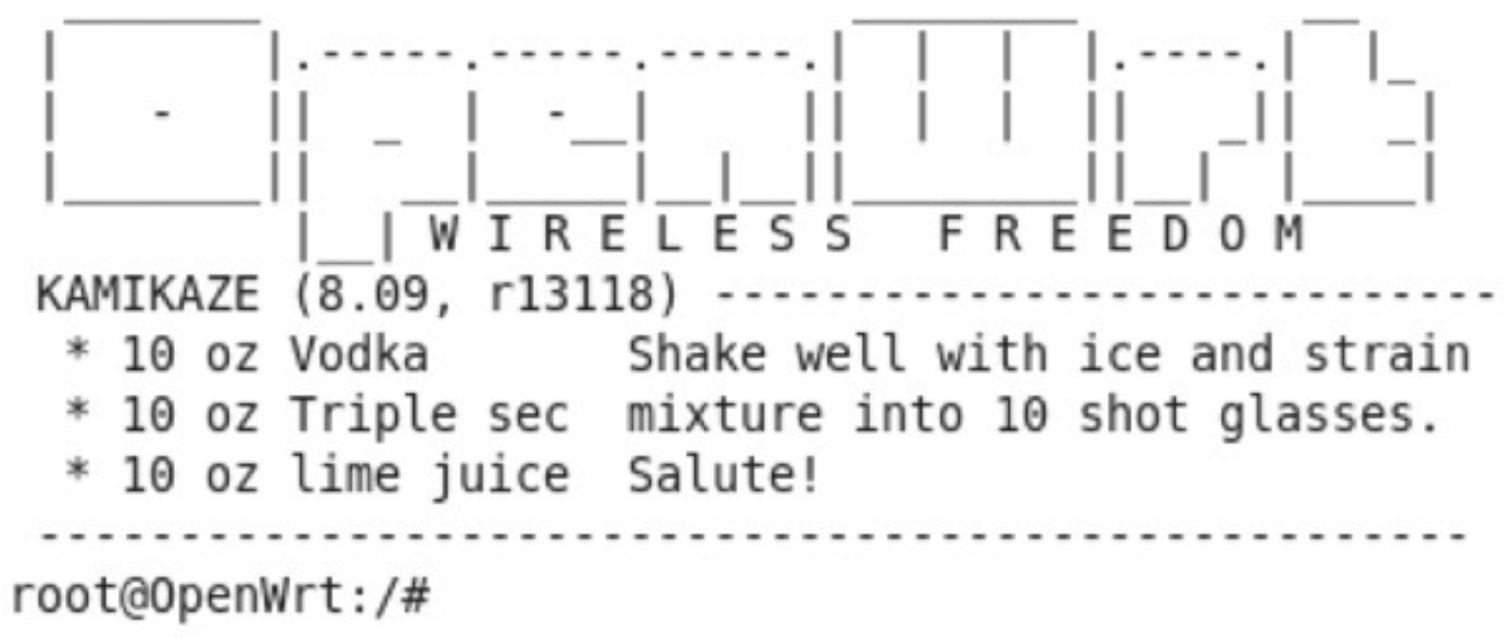
OpenWRT

```
 _______                     ________        __
|       |.-----.-----.-----.|  |  |  |.----.|  |_
|   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
|_______||   __|_____|__|__||________||__|  |____|
         |__| W I R E L E S S   F R E E D O M
KAMIKAZE (8.09, r13118) ------------------------------
 * 10 oz Vodka       Shake well with ice and strain
 * 10 oz Triple sec  mixture into 10 shot glasses.
 * 10 oz lime juice  Salute!
 ---------------------------------------------------
root@OpenWrt:/#
```

Features:

- Opensource
- Ready for Olsr
- Easy to install
- Simple GUI

OpenWRT

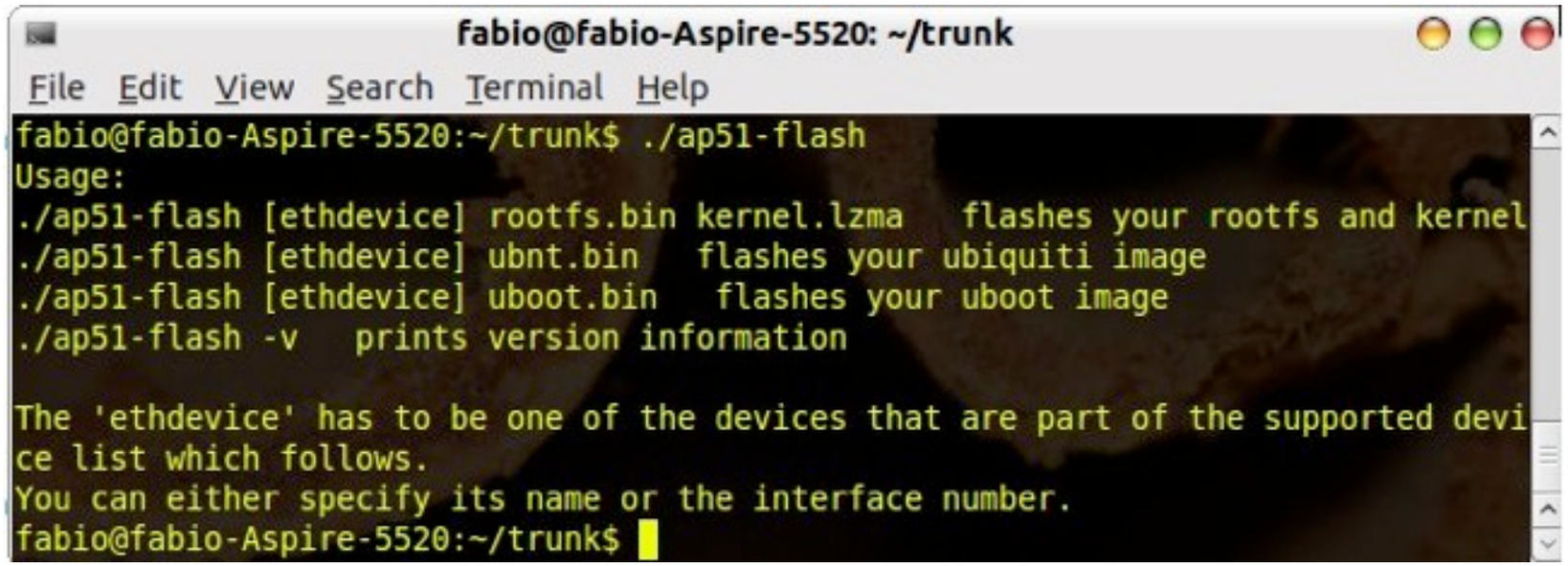


Features:

- Opensource
- Ready for Olsr
- Easy to install
- Simple GUI
- Large device support

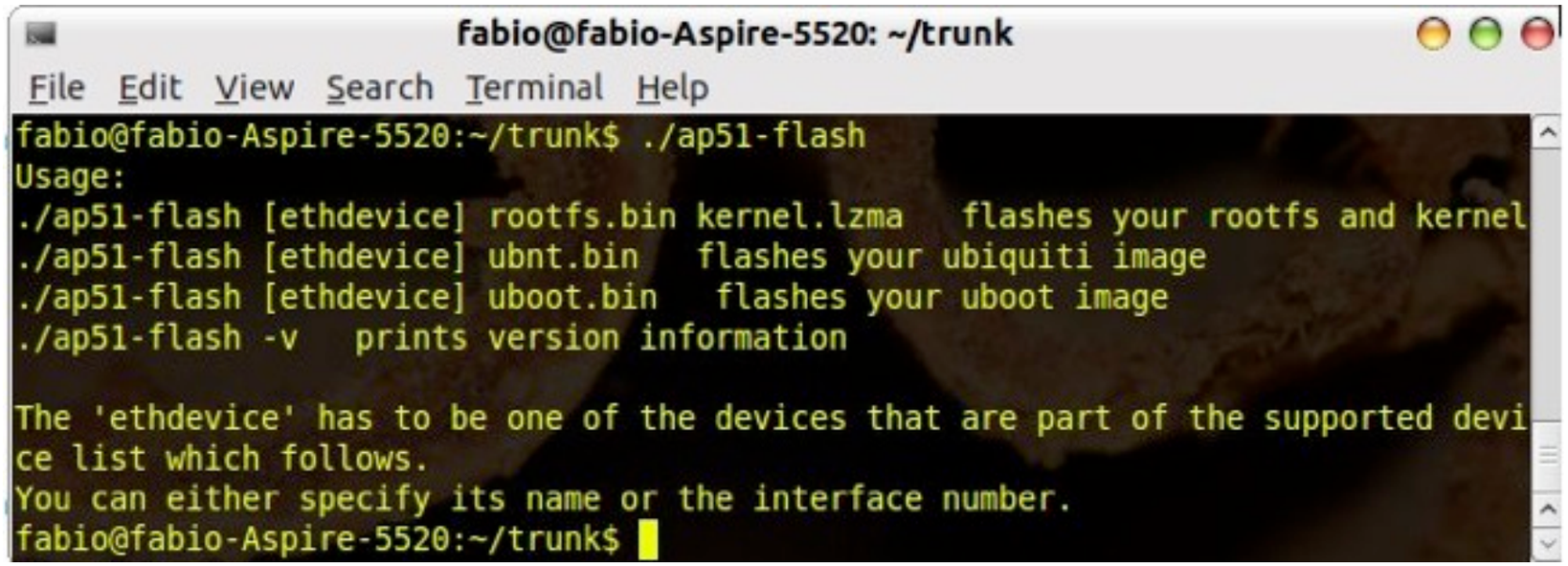
Device Flash

Device Flash



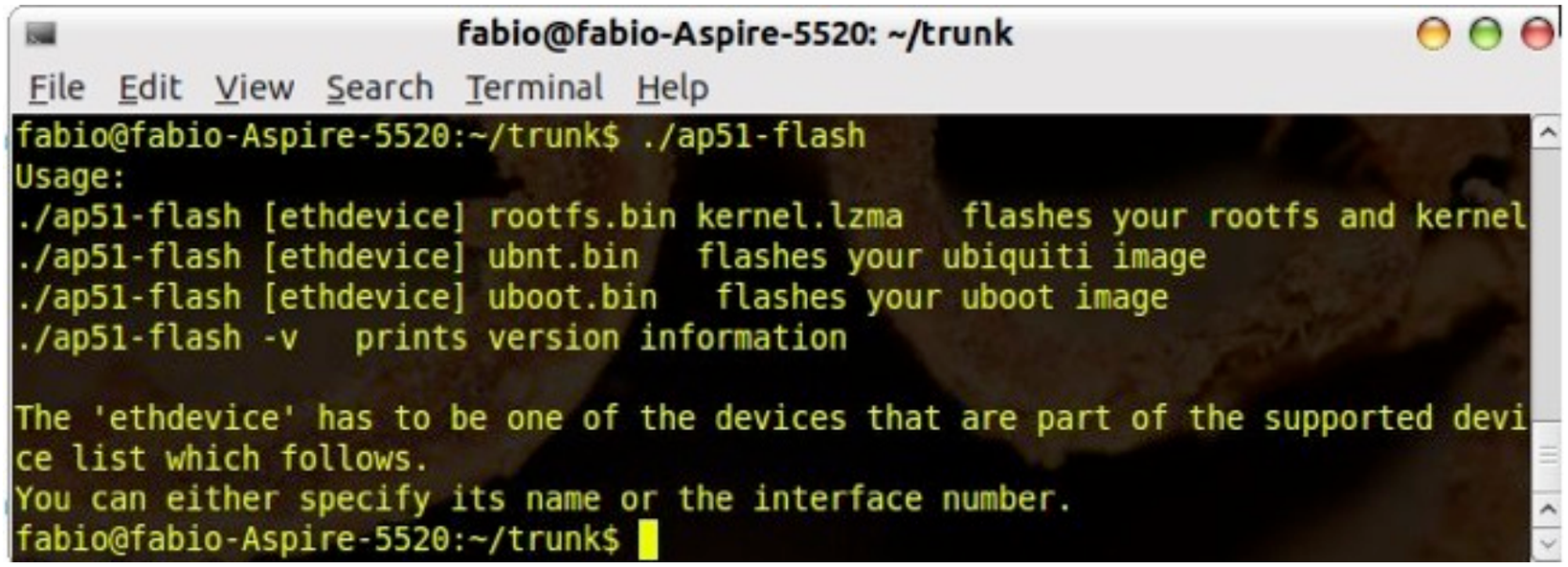
You need:

Device Flash



You need: •root filesystem

Device Flash



You need:
- root filesystem
- kernel

Device Flash

fabio@fabio-Aspire-5520: ~/trunk

File Edit View Search Terminal Help

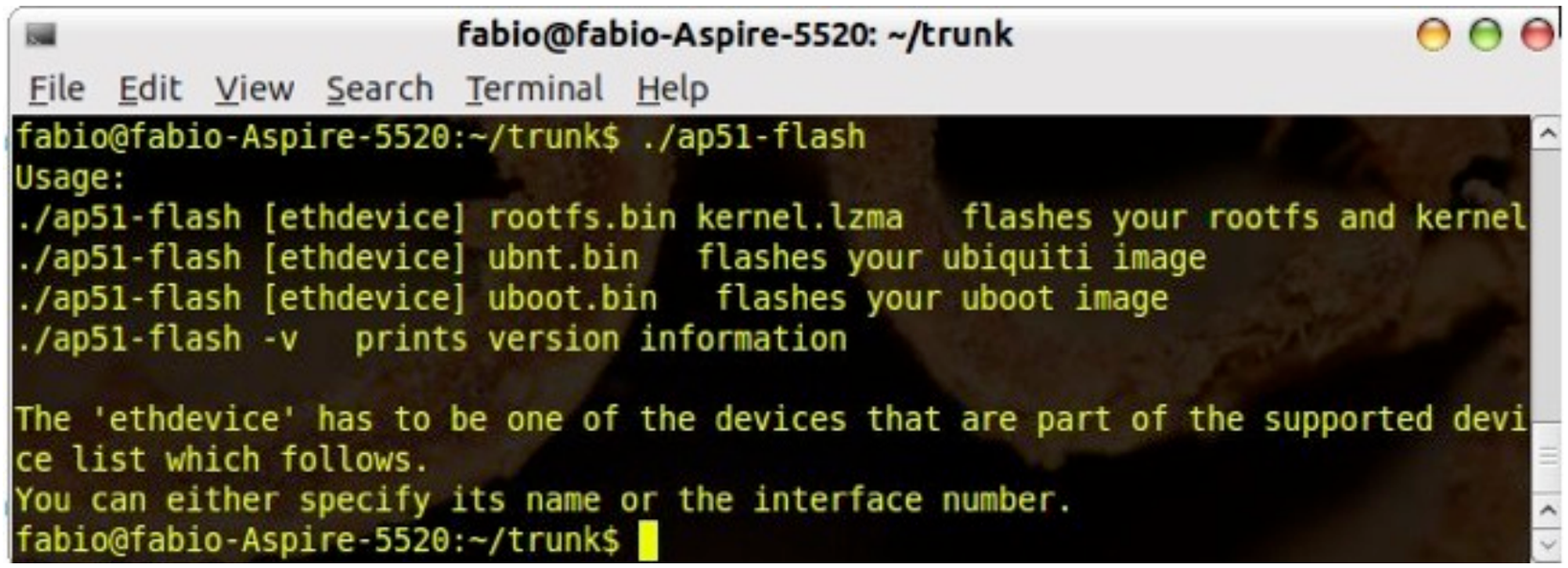
```
fabio@fabio-Aspire-5520:~/trunk$ ./ap51-flash
Usage:
./ap51-flash [ethdevice] rootfs.bin kernel.lzma   flashes your rootfs and kernel
./ap51-flash [ethdevice] ubnt.bin   flashes your ubiquiti image
./ap51-flash [ethdevice] uboot.bin   flashes your uboot image
./ap51-flash -v   prints version information

The 'ethdevice' has to be one of the devices that are part of the supported devi
ce list which follows.
You can either specify its name or the interface number.
fabio@fabio-Aspire-5520:~/trunk$
```

You need:
- root filesystem
- kernel

Flash!

Device Flash



You need:
- root filesystem
- kernel

Flash!

sudo ./ap51flash eth0 openwrt-atheros-root.squashfs openwrt-atheros-vmlinux.lzma

Getting Involved

‘Nerd’ Thursday

every thursday at FusoLab

LABORATORIO CREATIVO FUSOLAB
VIA G. PITACCO, 29

http://wiki.ninux.org
http://blog.ninux.org