



# Mobile Ad-Hoc Networks

Andreas Tønnesen  
andreto@olsr.org

[www.olsr.org](http://www.olsr.org)

[www.unik.no](http://www.unik.no)



# Agenda

- A introduction to MANET
- Routing in MANETs
- Optimized LinkState Routing
- The UniK OLSR implementation
- Extensions to UniK olsrd
- MANET and free community networks



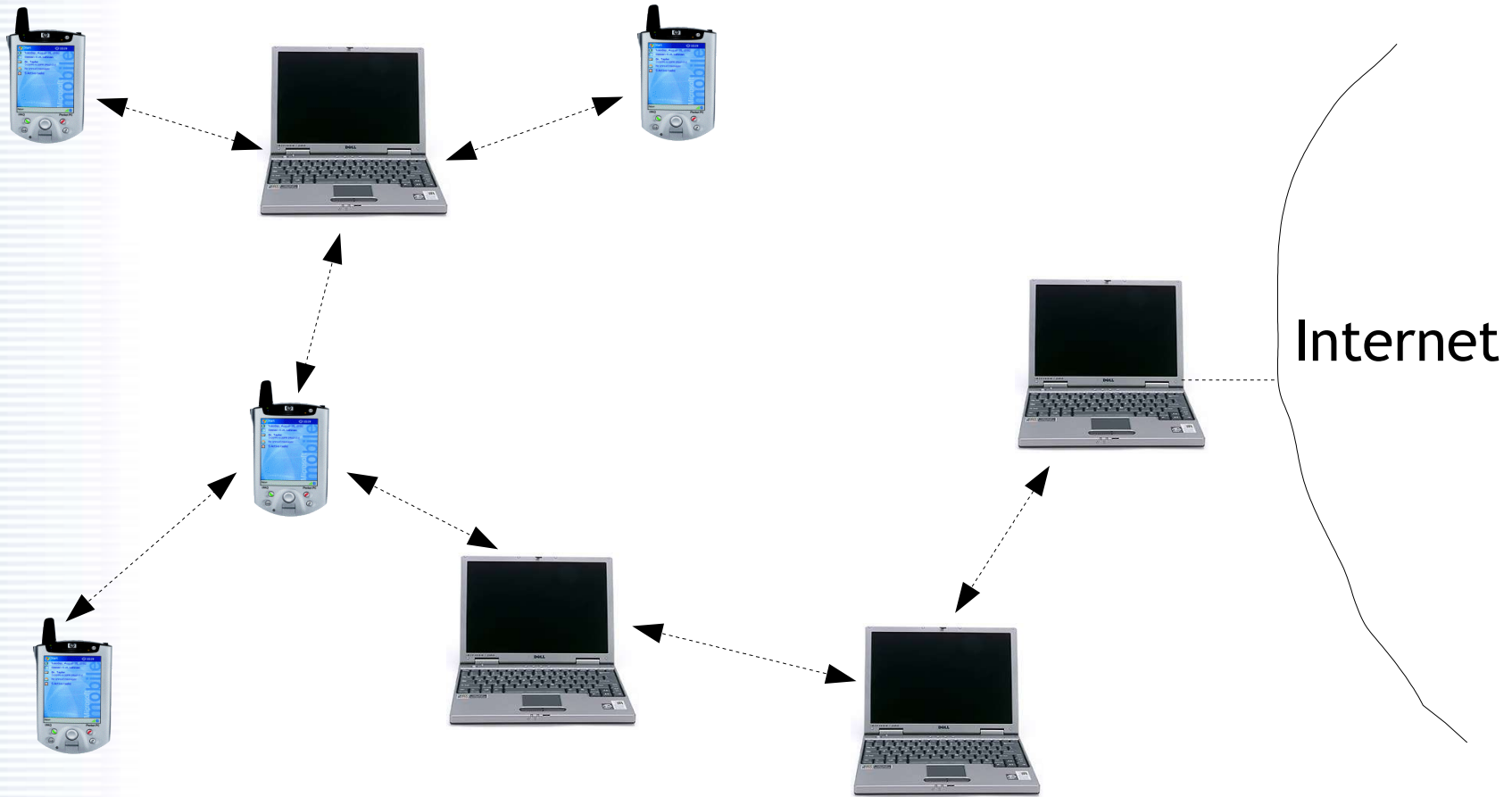
# Mobile Ad-Hoc Networks - MANET

---

- Rapidly deployable, self configuring.
- No need for existing infrastructure.
- Wireless links.
- Nodes are *mobile*, topology can be very dynamic.
- Nodes must be able to *relay traffic* since communicating nodes might be out of range.
- A MANET can be a standalone network or it can be connected to external networks(Internet).



# Example of a MANET





# MANET usage areas

- Military scenarios
- Sensor networks
- Rescue operations
- Students on campus
- Free Internet connection sharing
- Conferences

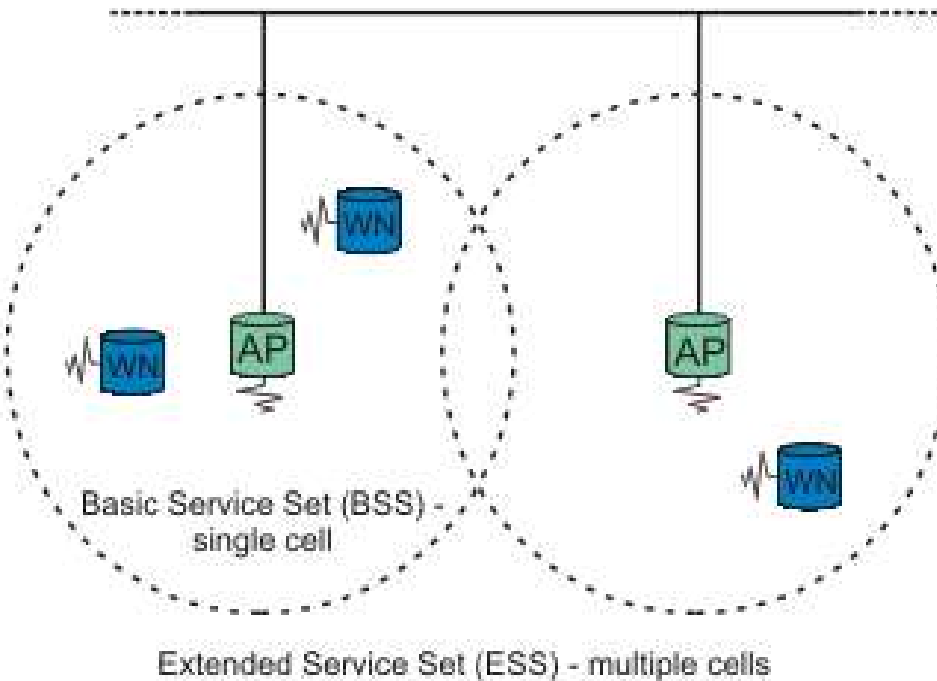
The main two characteristics are *mobility* and *multihop*.



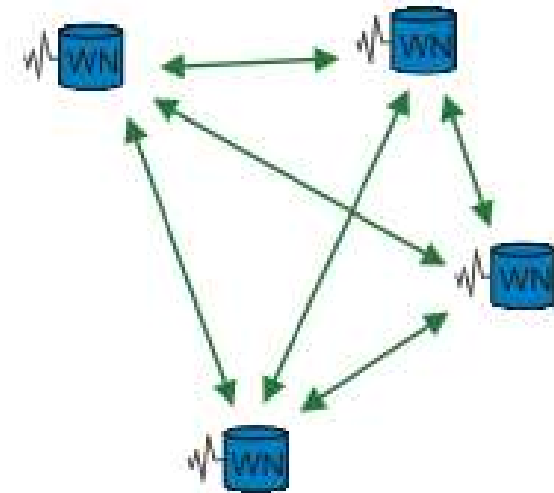
# IEEE 802.11b and MANET(1)

802.11b specifies two main operating modes.

Infrastructure Mode:



IEEE Ad-hoc Mode:

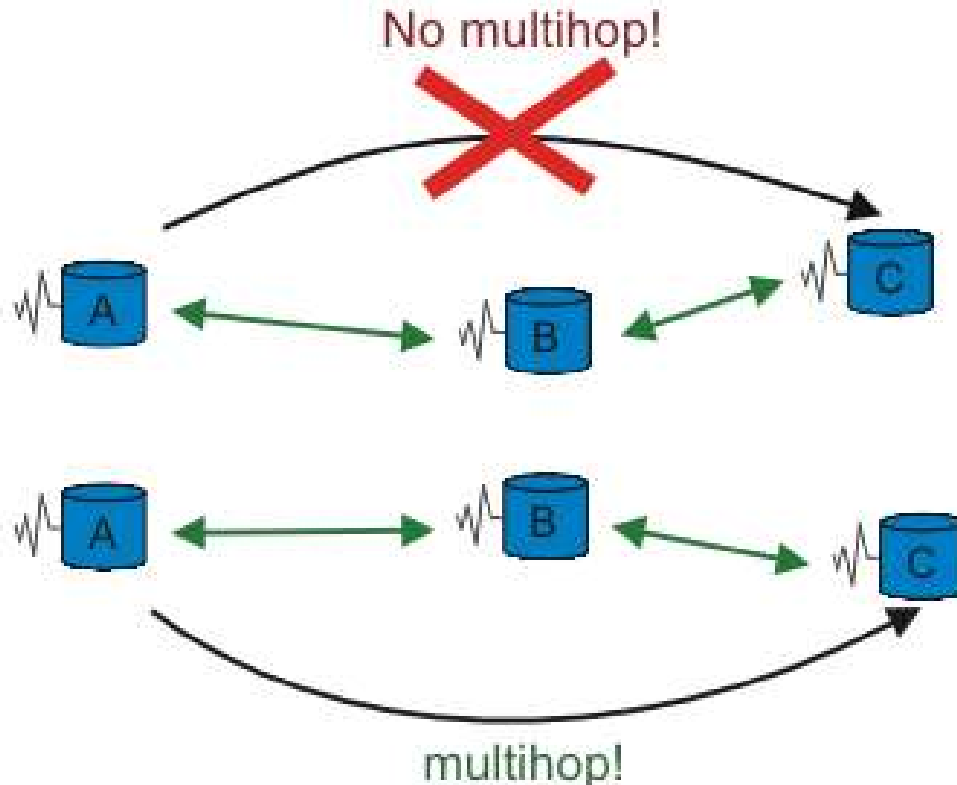




# IEEE 802.11b and MANET(2)

So is 802.11 Ad-hoc mode enough?

**IEEE Ad-hoc Mode:**  
A can NOT talk to  
C through B!



**OLSR Ad-Hoc  
("real ad-hoc"):**  
A can talk to  
C through B!



# Mechanisms required in a MANET

---

- Multihop operation requires a routing mechanism designed for mobile nodes.
- Internet access mechanisms.
- Self configuring networks requires an address allocation mechanism.
- Mechanism to detect and act on, merging of existing networks.
- Security mechanisms.



# Routing protocol requirements

---

- Self starting and self organizing
- Multi-hop, loop-free paths
- Dynamic topology maintenance
- Rapid convergence
- Minimal network traffic overhead
- Scalable to large networks



# The IETF MANET working group

Main purpose:

To standarize IP routing in mobile ad-hoc networks.

Three routing protocols accepted as experimental RFCs, and a fourth one coming up.

These protocols fall into two categories:

- Re-active
- Pro-active



# Re-active routing protocols

---

- Does not take initiative for finding routes
- Establishes routes “on demand” by flooding a query

## Pros and cons:

- Does not use bandwidth except when needed (when finding a route)
- Much network overhead in the flooding process when querying for routes
- Initial delay in traffic



# Re-active routing - AODV(RFC3561)

A wants to communicate with B



B



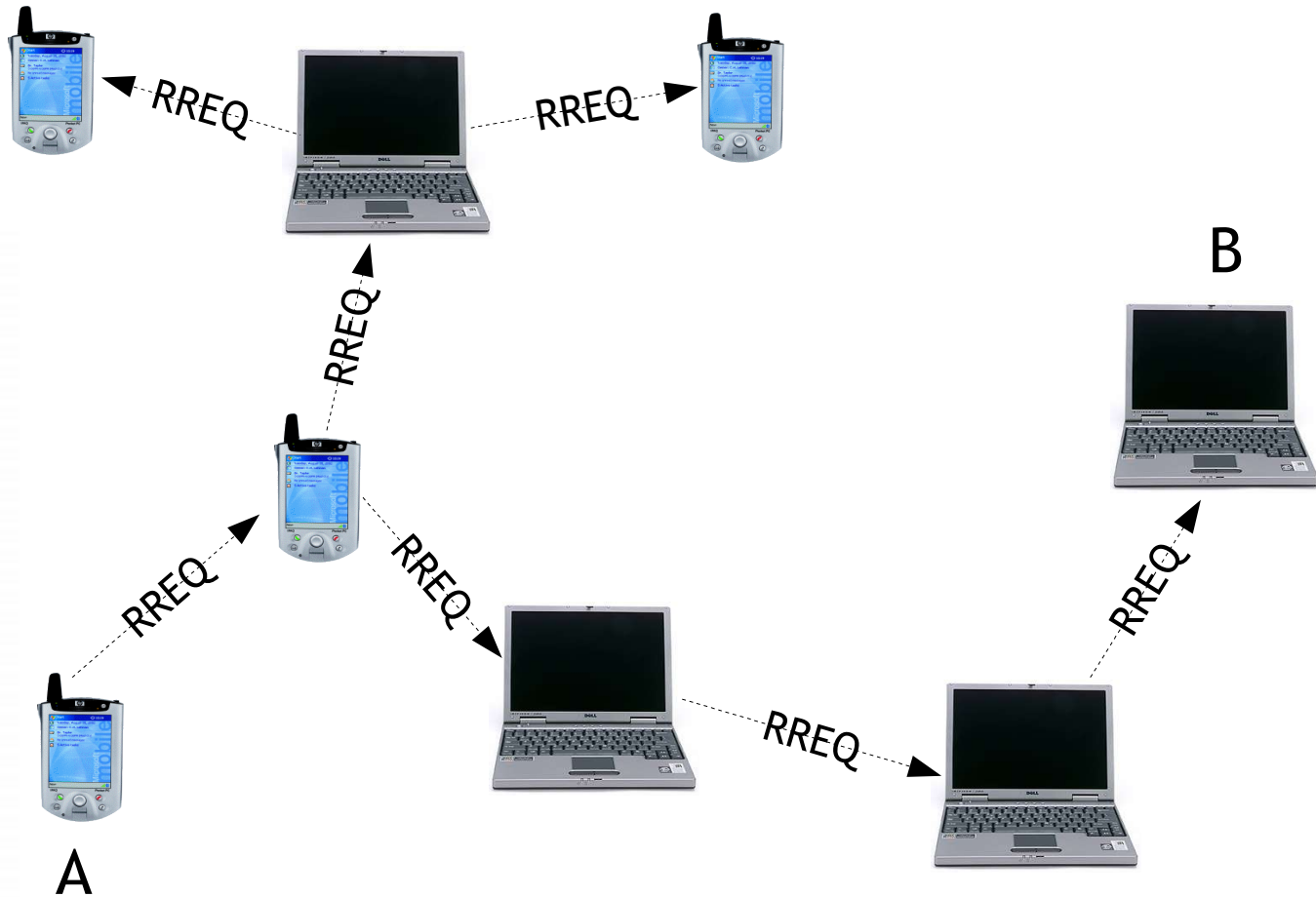
A





# Re-active routing - AODV(RFC3561)

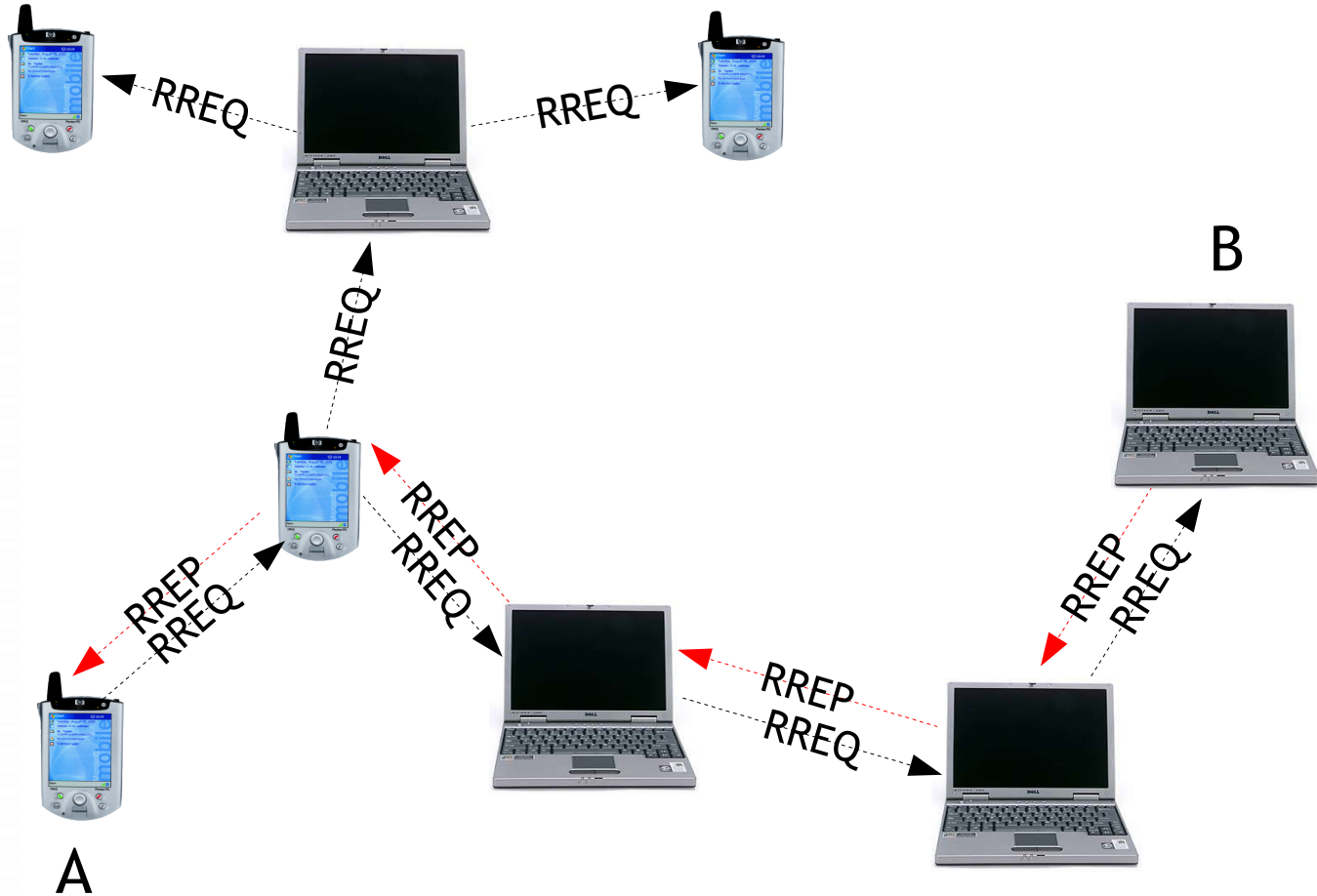
A floods a route request





# Re-active routing - AODV(RFC3561)

A route reply is unicasted back





# Pro-active routing protocols

---

- Routes are set up based on continuous control traffic
- *All* routes are maintained all the time

Pros and cons:

- Constant overhead created by control traffic
- Routes are always available



# Pro-active routing - OLSR(RFC3626)

---

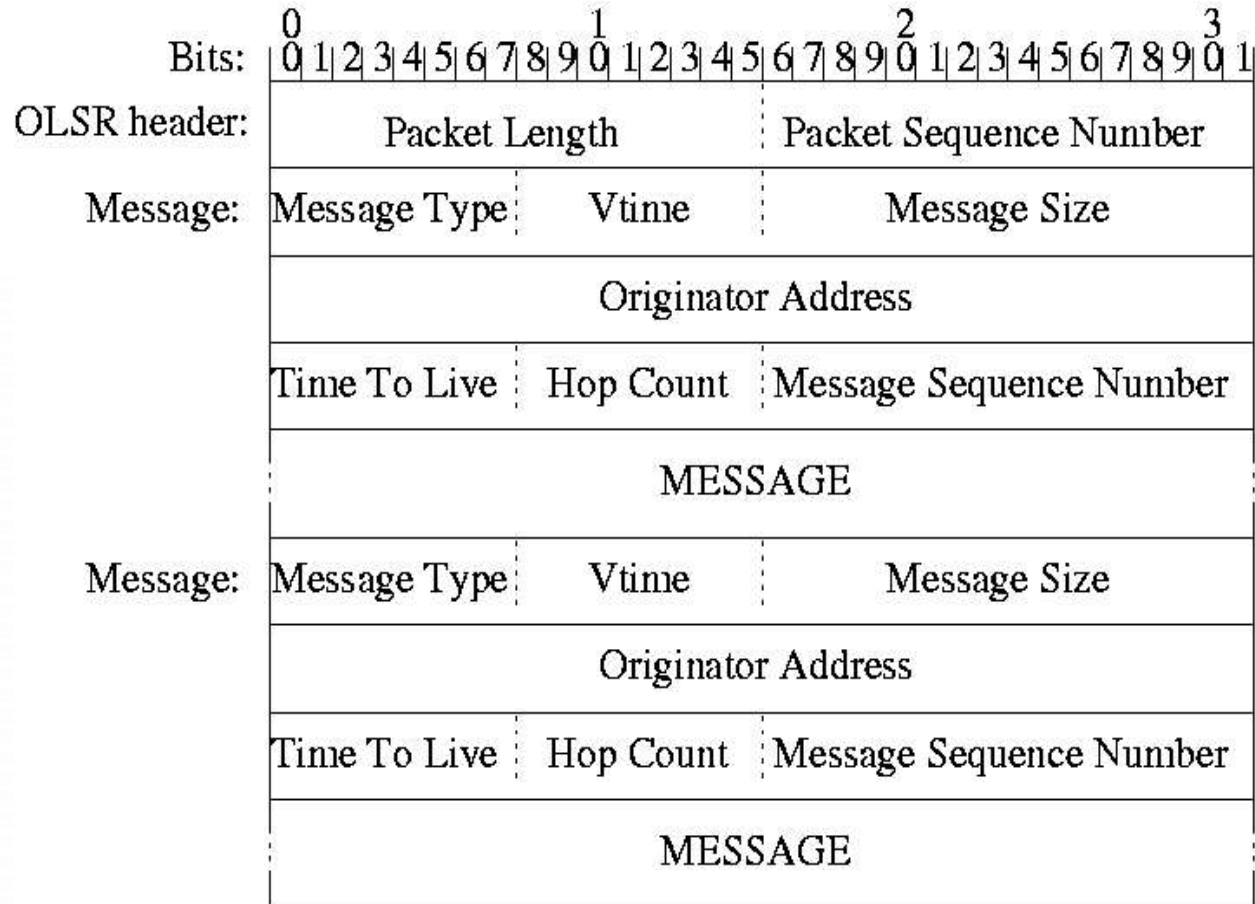
Developed by INRIA(France).

The Optimized Link-State Routing protocol can be divided in to three main modules:

- Neighbor/link sensing
- Optimized flooding/forwarding(MultiPoint Relaying)
- Link-State messaging and route calculation



# OLSR packet format



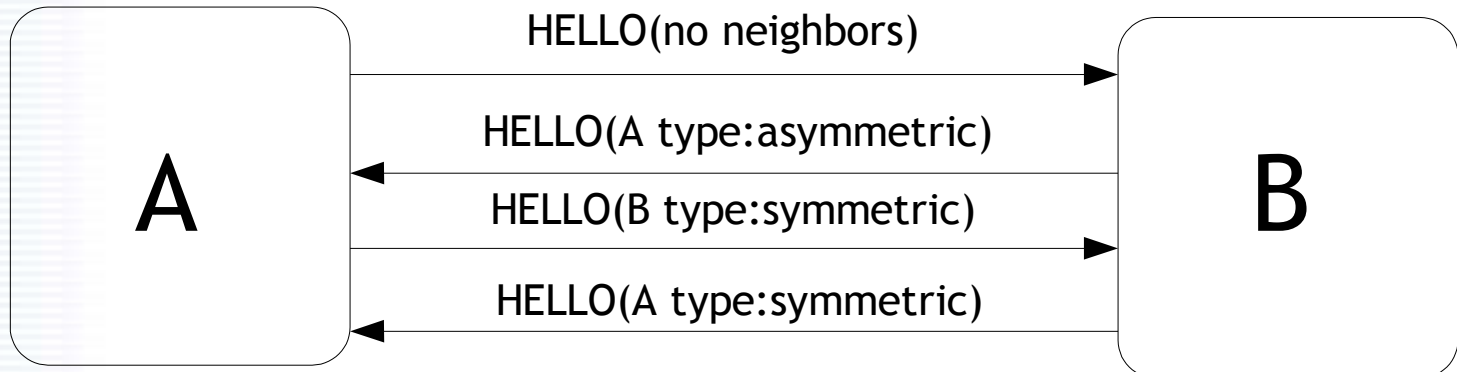


# Link and neighbor sensing

Neighbors and links are detected by HELLO messages.

All nodes transmit HELLO messages on a given interval. These contain all heard-of neighbors grouped by status.

A simplified neighbor detection scenario:





# Multipoint Relaying

---

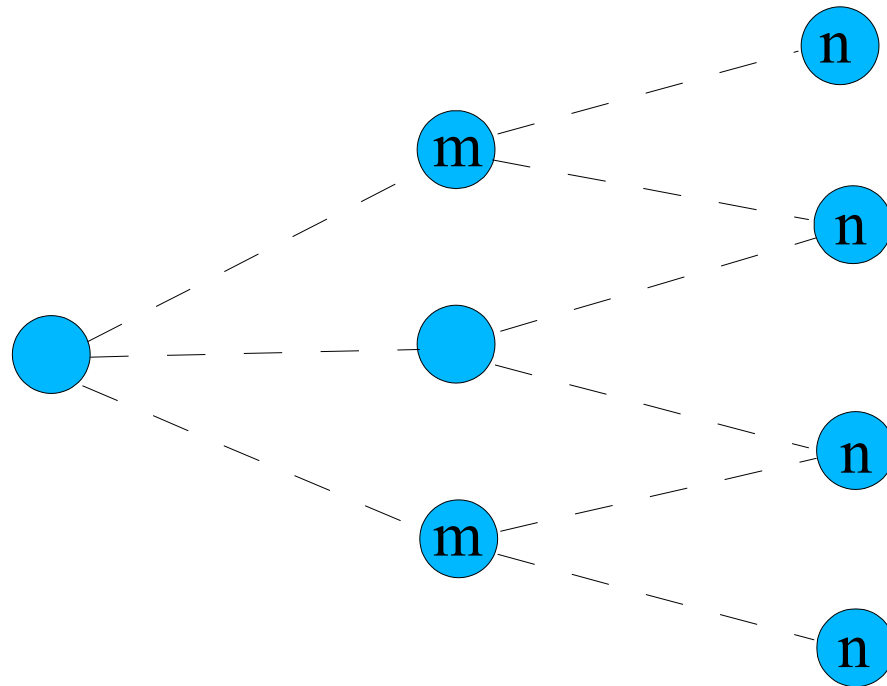
- Reduce the number of duplicate retransmissions while forwarding a broadcast packet.
- Restricts the set of nodes retransmitting a packet from all nodes (regular flooding) to a subset of all nodes.
- The size of this subset depends on the topology of the network.



# Multipoint Relay selection

All nodes select and maintain their own MPRs.

Rule: *“For all 2 hop neighbors n there must exist a MPR m so that n can be contacted via m.”*



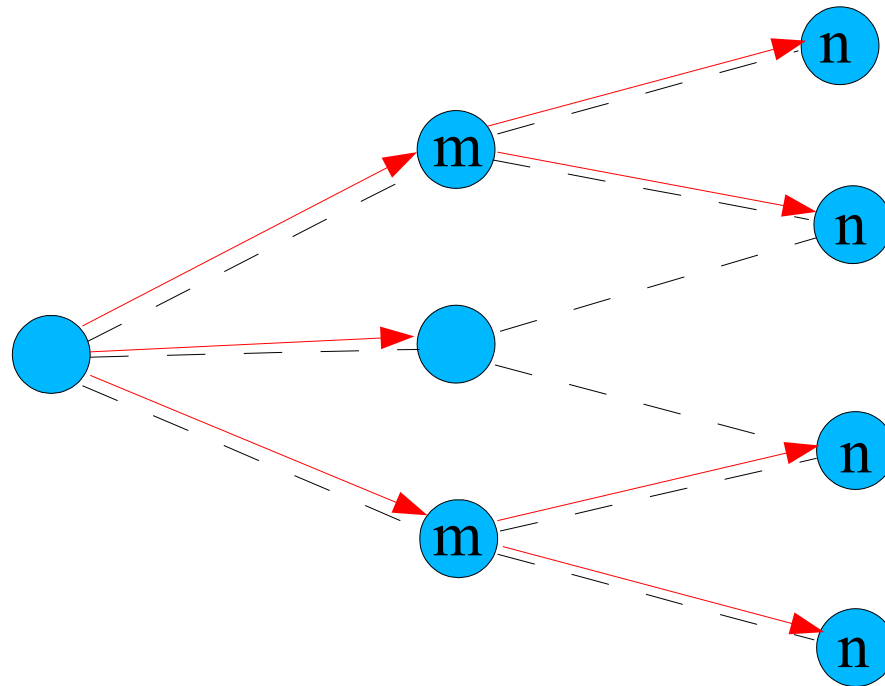


# Forwarding of traffic

All nodes registers and maintains their MPR selectors.

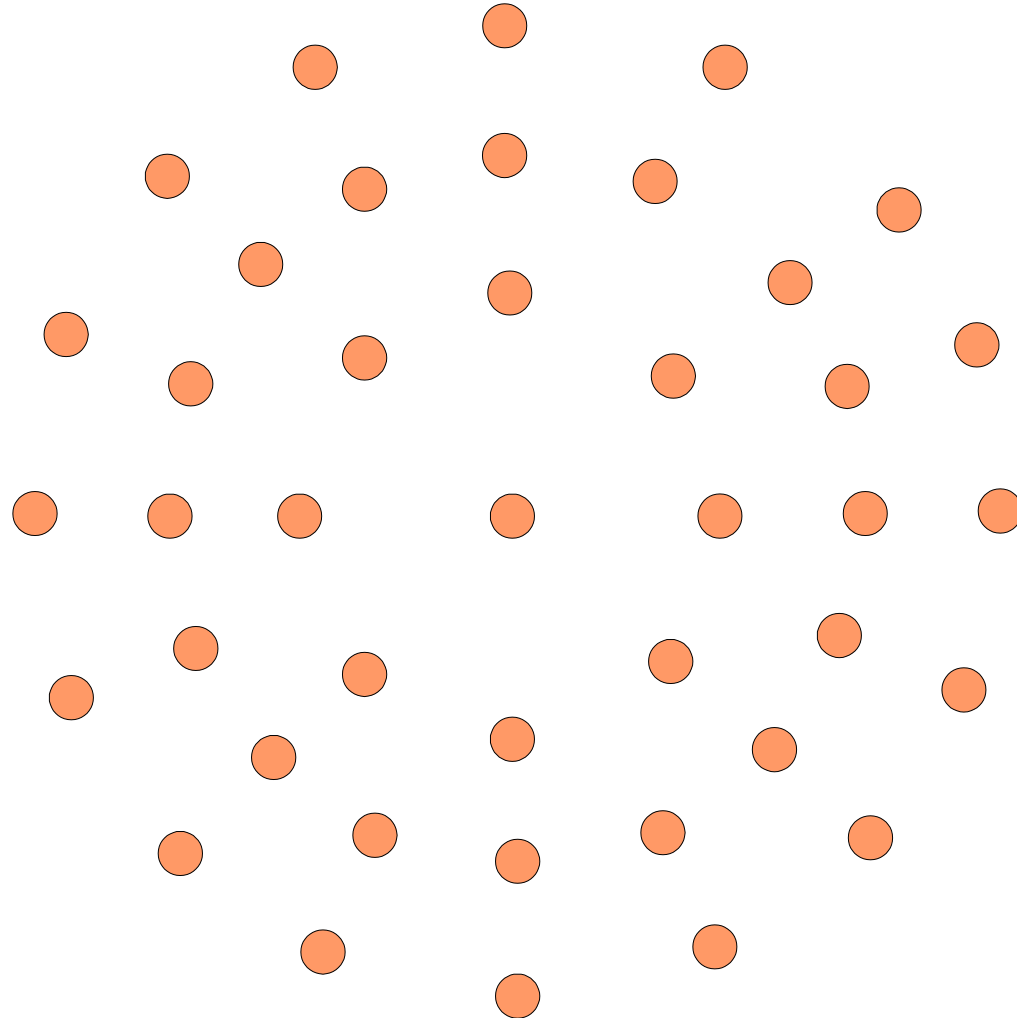
Rule: *“If OLSR-packet is received from a MPR selector, then all messages contained in that packet are to be forwarded if  $TTL > 0$ .”*

This goes for *both known and unknown* message types.





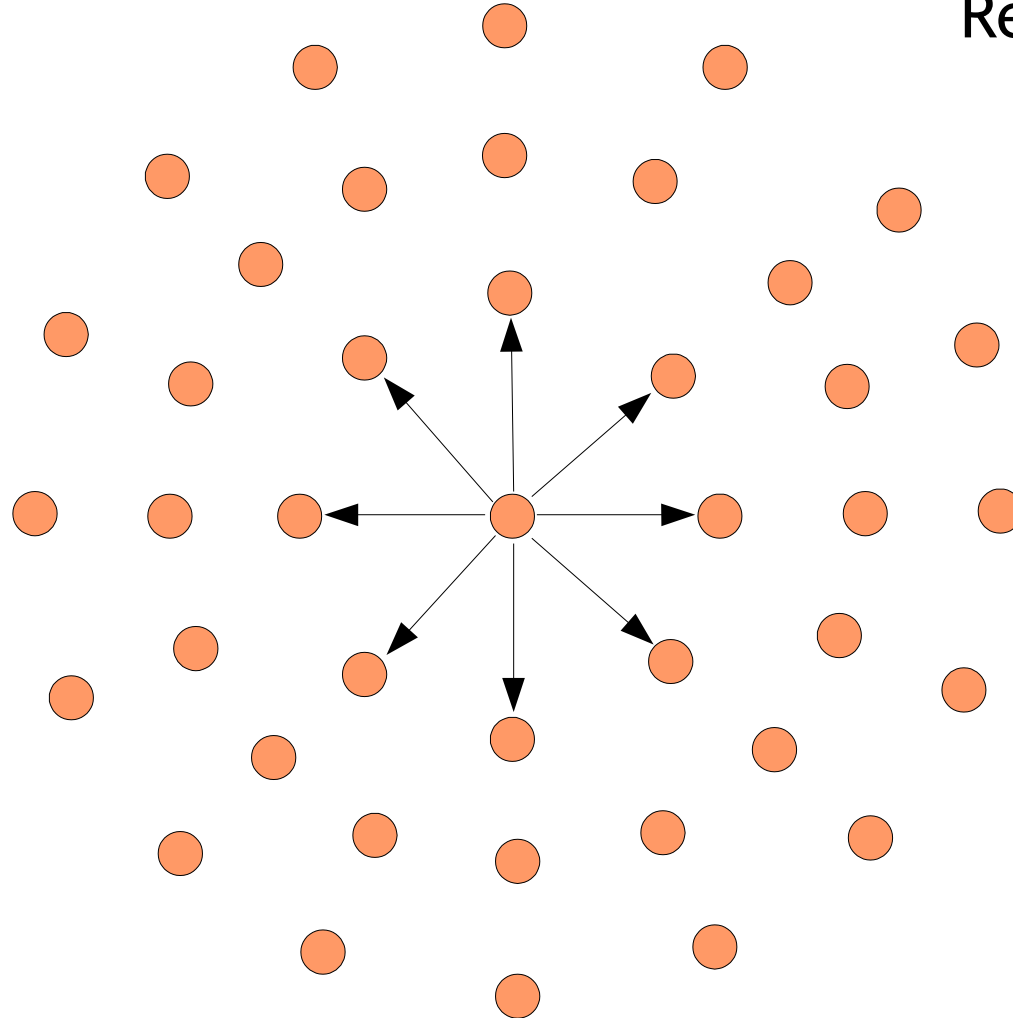
# Multipoint Relaying - example





# Multipoint Relaying - example

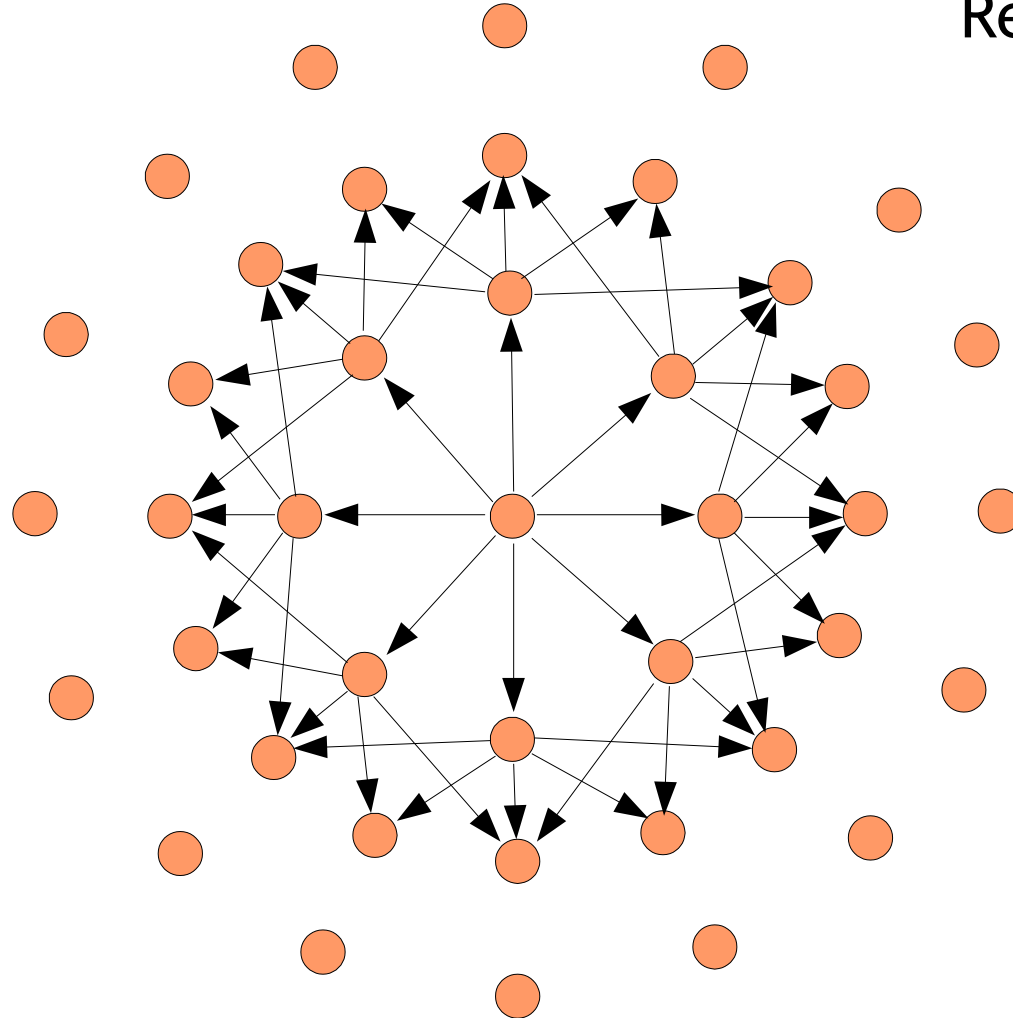
Regular flooding 1





# Multipoint Relaying - example

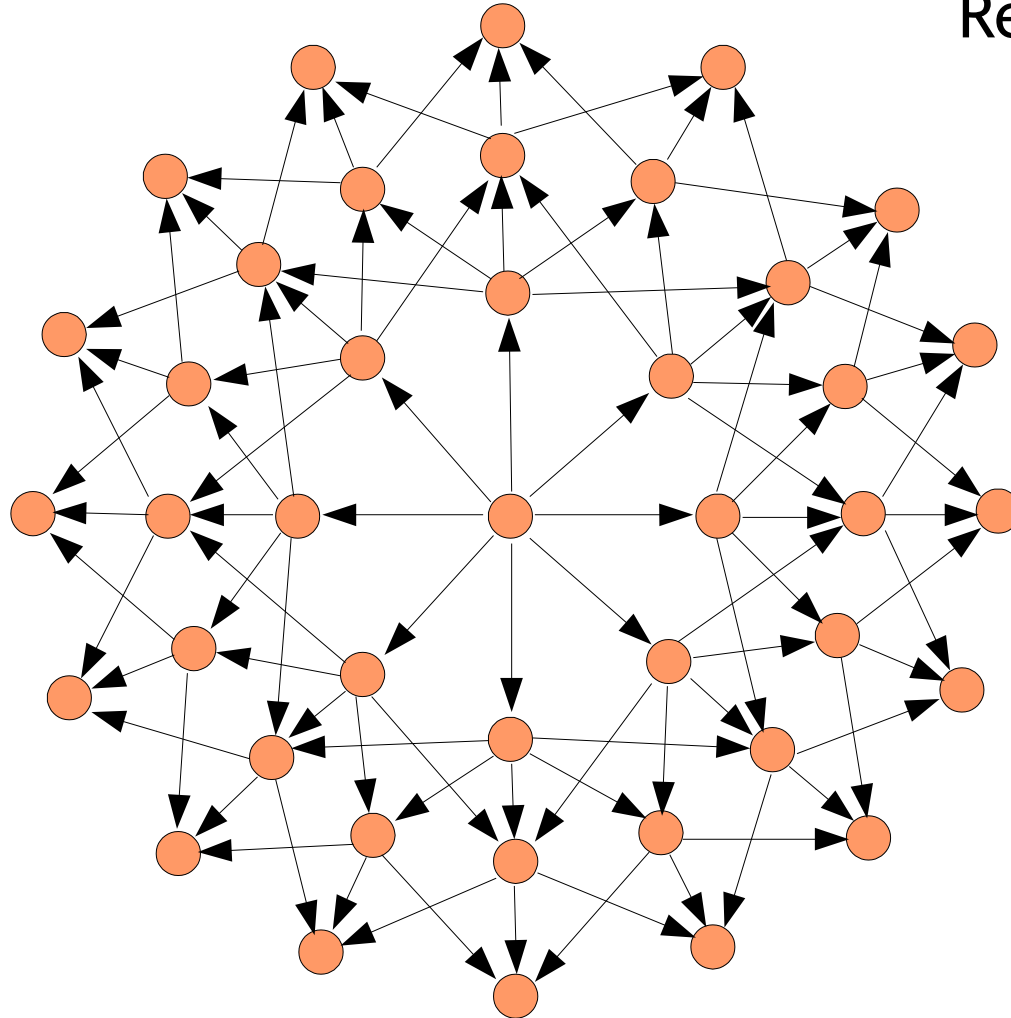
Regular flooding 2





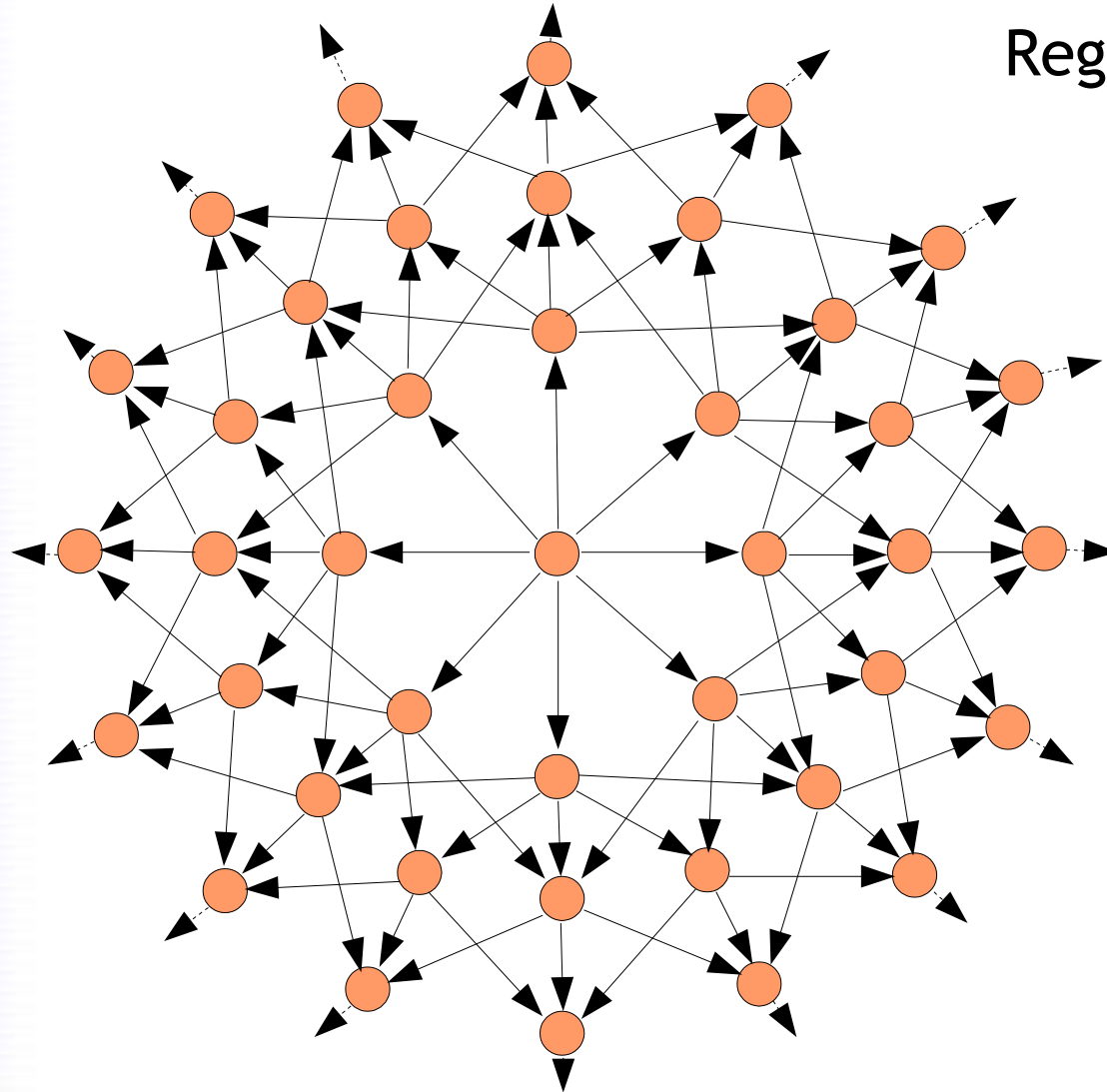
# Multipoint Relaying - example

Regular flooding 3





# Multipoint Relaying - example

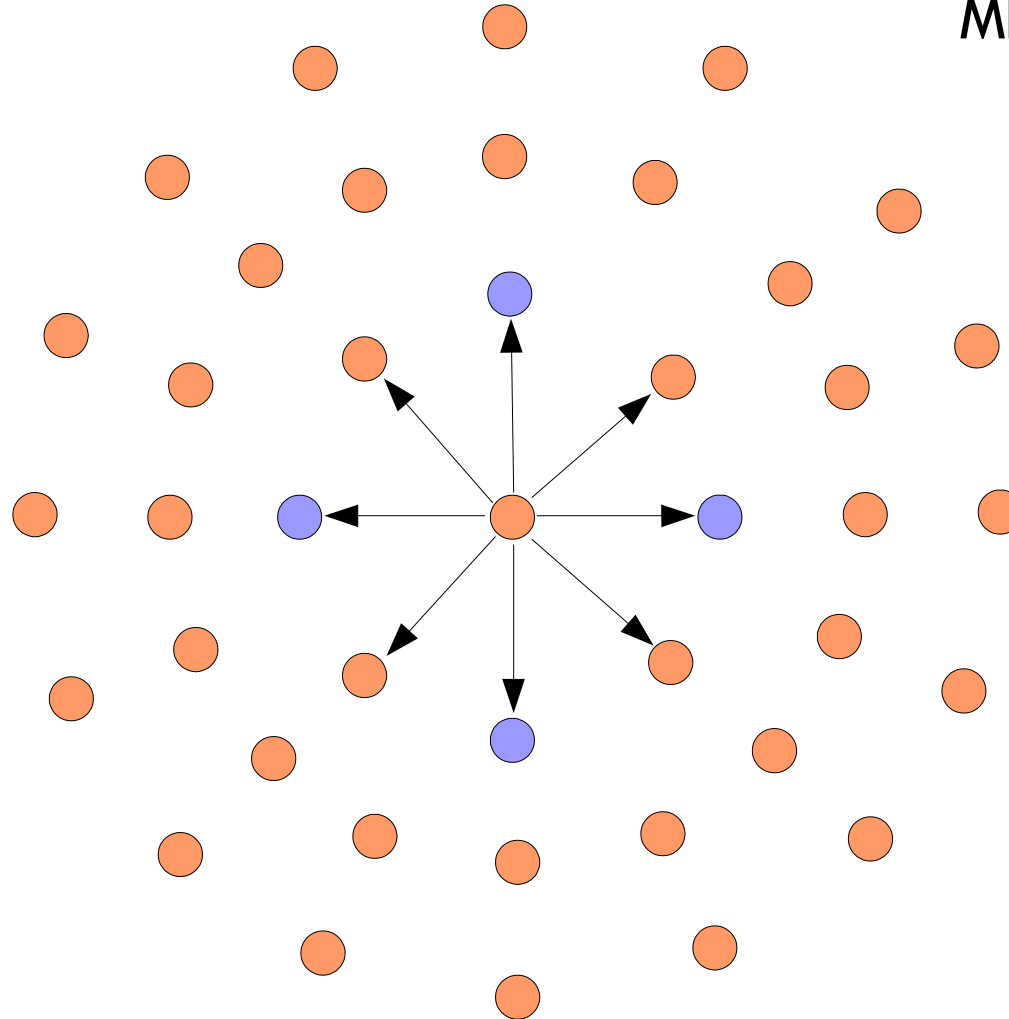


Regular flooding 4



# Multipoint Relaying - example

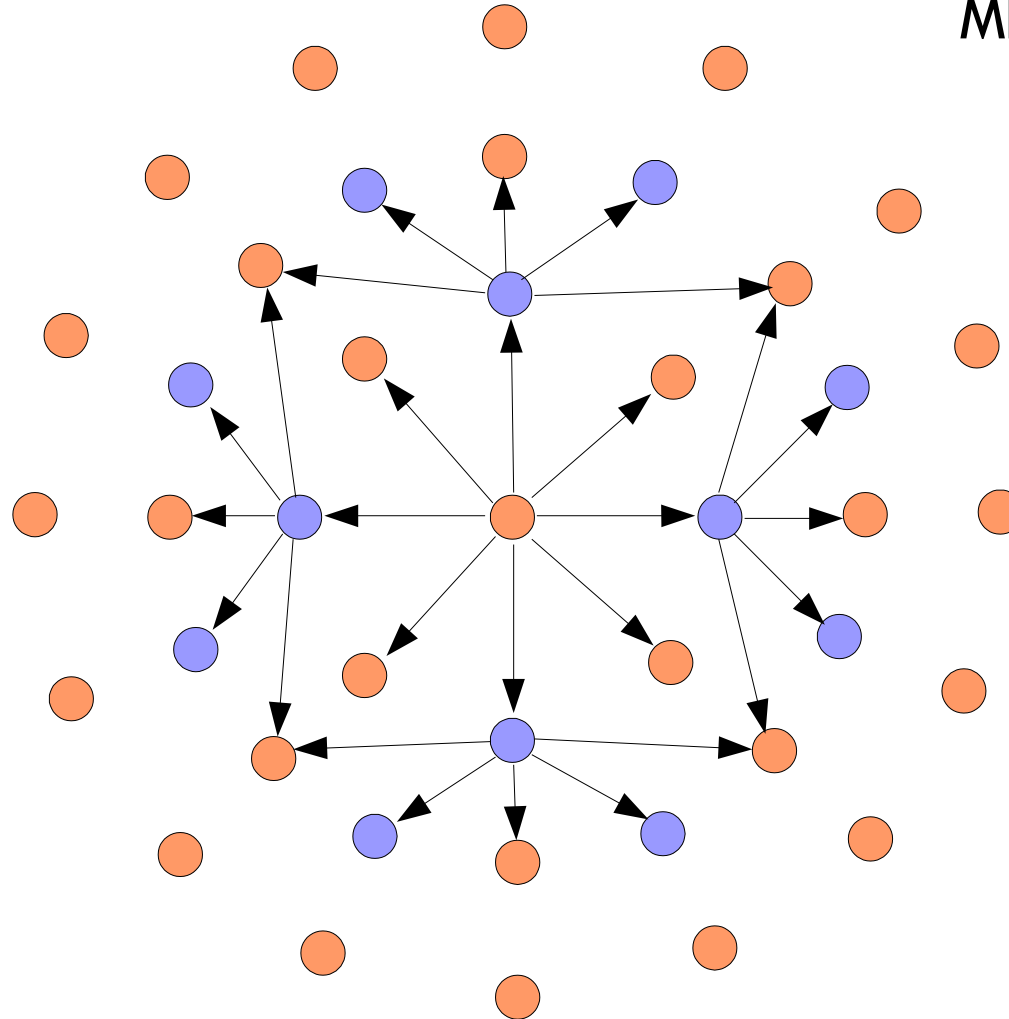
MPR flooding 1





# Multipoint Relaying - example

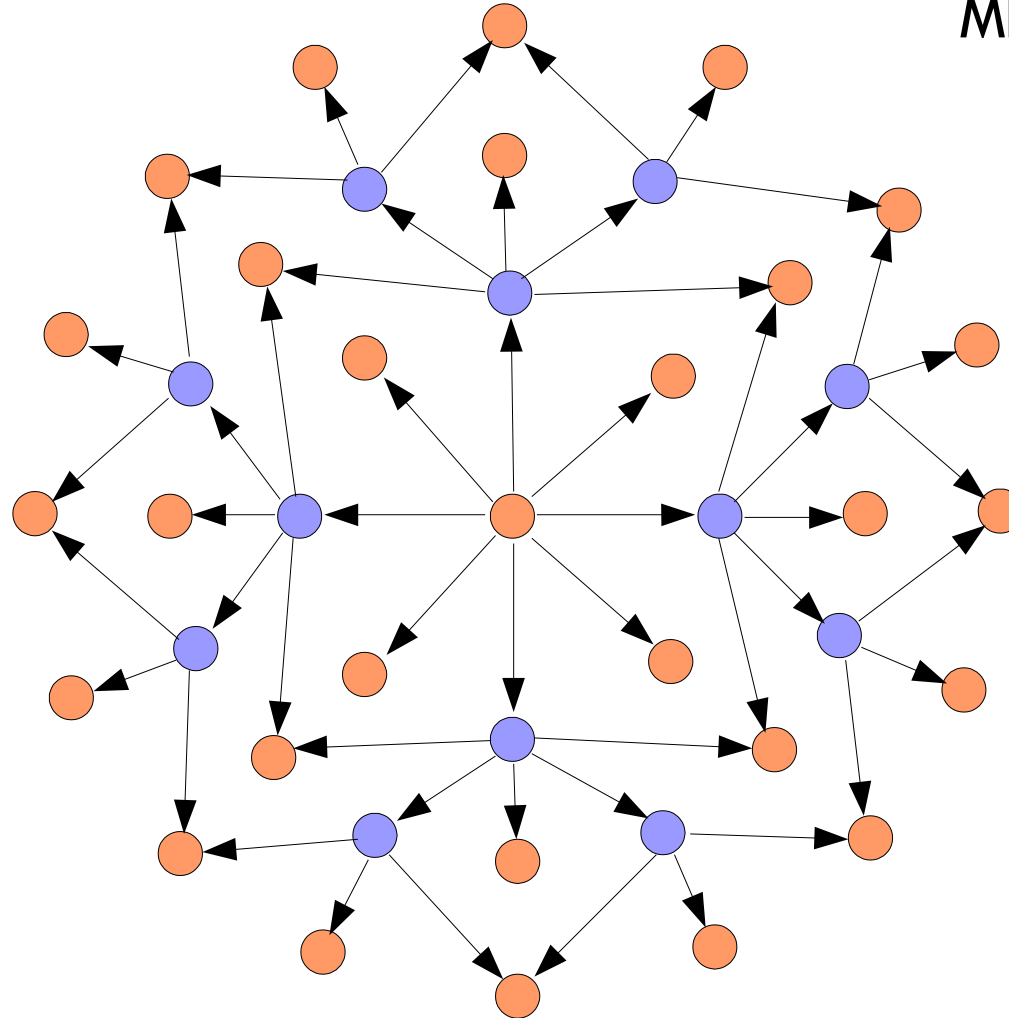
MPR flooding 2





# Multipoint Relaying - example

MPR flooding 3





# LinkState functionality

---

In a classic link-state scheme all nodes flood the network with link-state information.

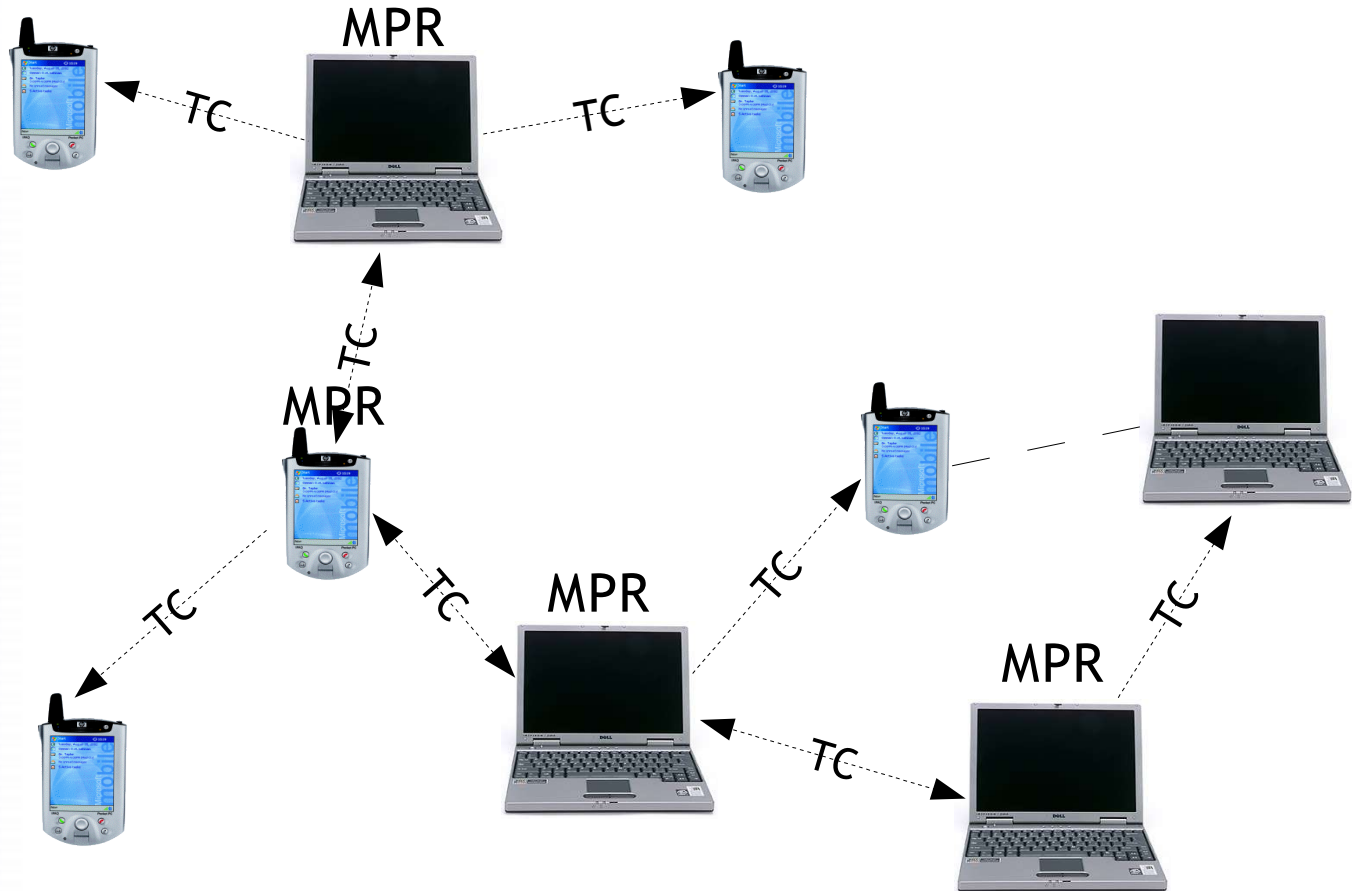
OLSR has two link-state optimizations:

- Only MPR selectors are declared in link-state messages. This minimizes the size of link-state messages.
- Only nodes selected as MPRs will generate link-state messages. This minimizes the set of nodes emitting link-state messages.



# LinkState Example

Messages declaring link-state are called Topology Control messages(TC)





# More functionality

---

We have only looked at the basic functionality of OLSR. OLSR also features:

- Can run on multi-homed hosts using MID messages
  - External network connectivity through HNA messages
  - Node willingness to act as MPR.
  - Link hysteresis
  - Tuneable MPR selector scheme
  - Tuneable TC generatoin
- ...and more



# Implementing OLSR

<http://www.olsr.org>



# The project

Goal: Develop a RFC3626 compliant implementation and look into possible extensions

Based on draft3 compliant nolsrd1a10 by INRIA

Drafts went from version 7 to 11 during spring -03

The last draft was version 11 - now experimental RFC(3626)

Almost all code rewritten - better to start from scratch?



# Technical

## 0.4.4 - latest release

Implementation for GNU/Linux systems(kernel >2.0)

Implemented in pure C

All areas of the RFC covered

Supports IPv4 and IPv6

Plugin support

Licensed under the General Public License(GPL)



# Hardware on which OLSRd runs

If it runs GNU/Linux it runs olsrd :-)

Standard i386 and PPC based computer systems.



The MIPS based WLAN accesspoint WRT54G from LinkSys(Cisco) and the *MeshCube*



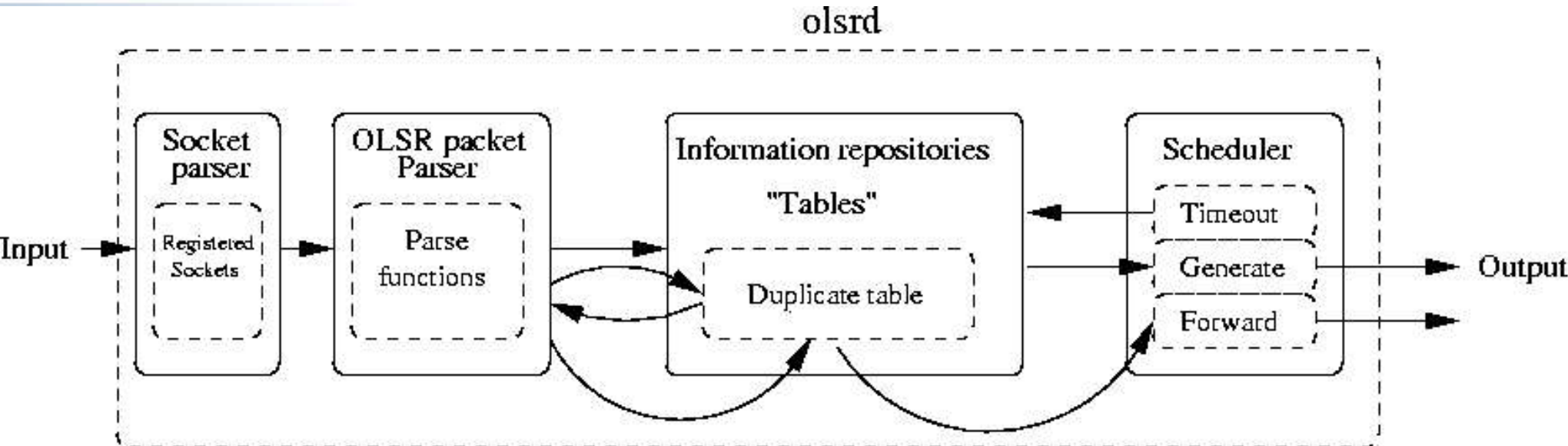
The strong-ARM based iPAQ from Compaq/HP





# Basic overview

Olsrd has a very modular design

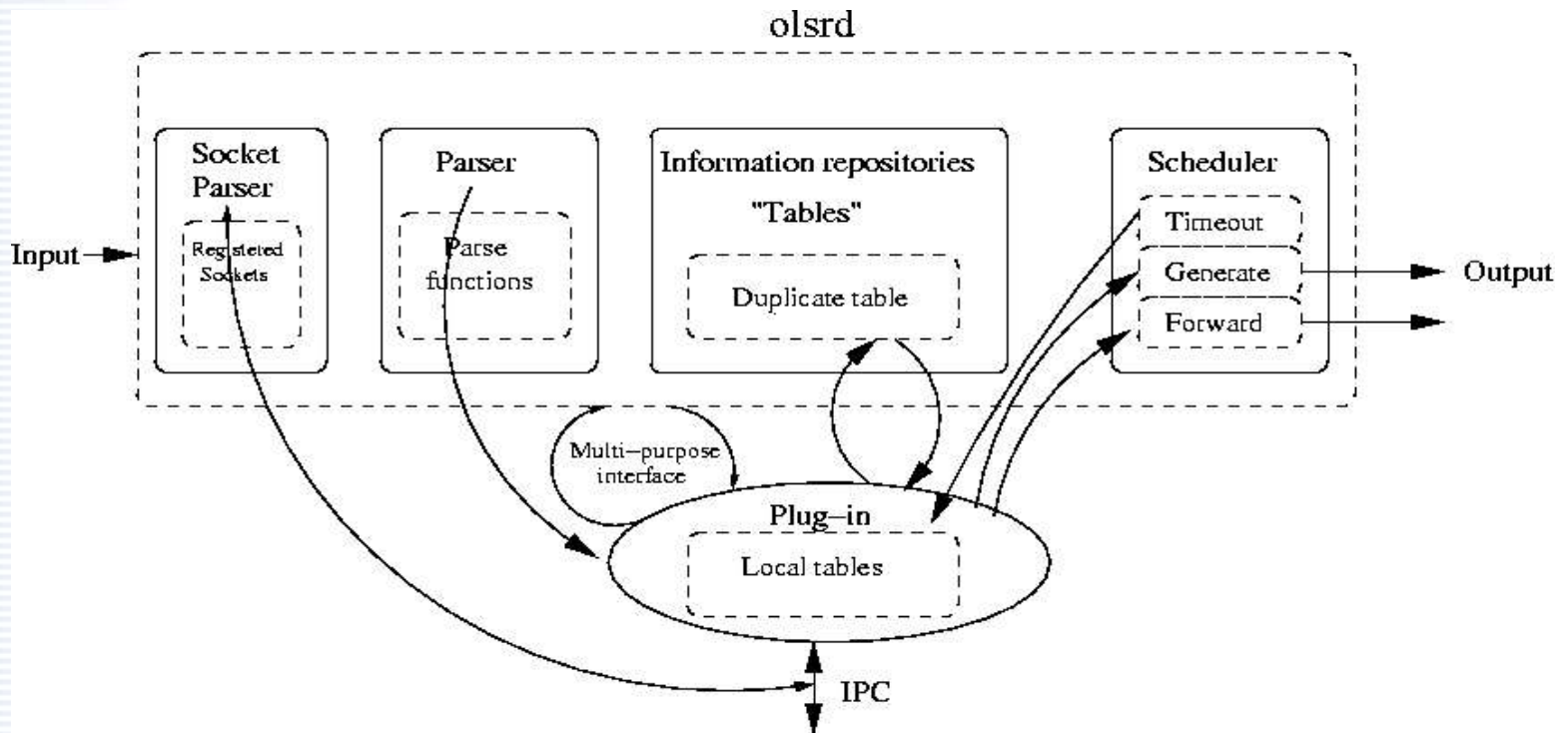


The scheduler runs in a thread of its own.



# Loadable plugins

A plugin(dynamically loaded library) is a piece of code that can be linked and “loaded” by an application in runtime.



Example: Can use OLSRs MPR flooding for net-wide broadcasts for services like DNS/service discovery.



# Loadable plugins

- No need to change any code in the olsr daemon to add custom packages.
- Users are free license plugins under whatever terms they like.
- Plug-ins can be written in any language that can be compiled as a dynamic library. GNU/Linux even allow scripts!
- No need for people with extended OLSR versions to rely on heavy patching to maintain functionality when new olsrd versions are released.



# Extensions

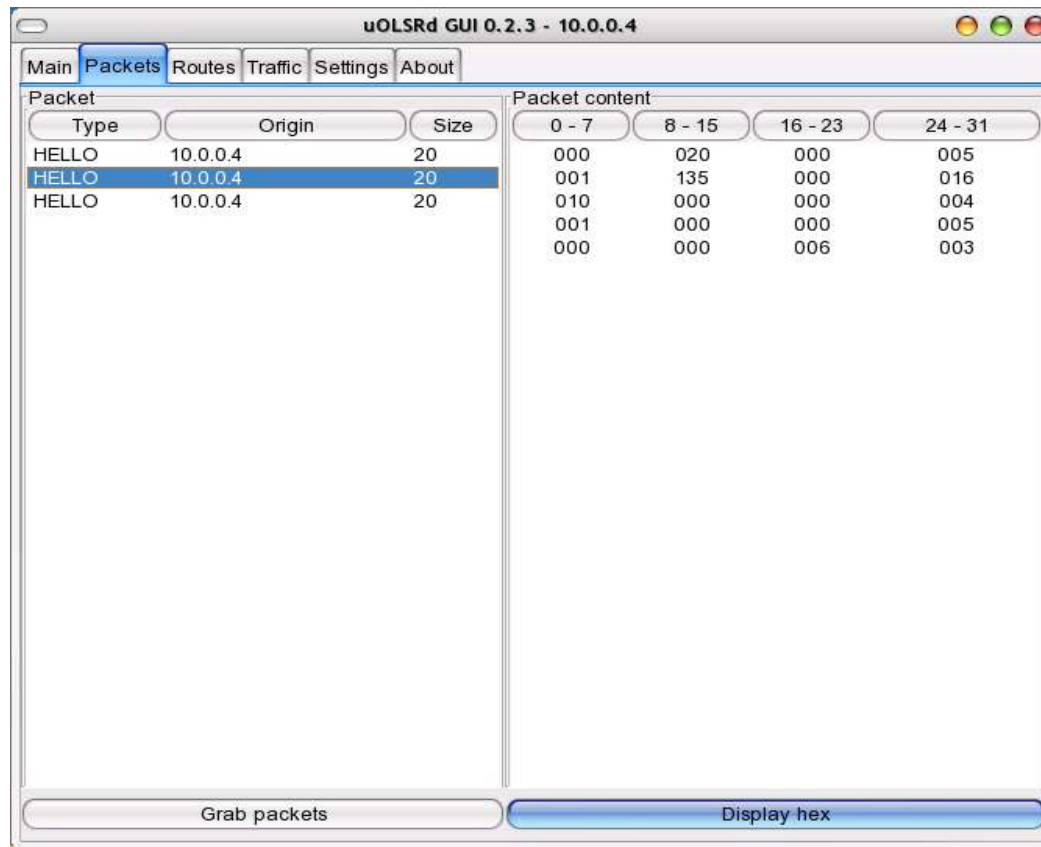
---

- GUI front-end
- Address auto configuration
- Control traffic security
- Gateway tunneling



# A GUI front-end

A GUI front end is written using the GTK library  
It communicates with the daemon over IPC and does not interfere with OLSR operation in any way.





# Pro-active autoconfig - PAA

---

A IP address auto-configuration protocol for ad-hoc networks routed by a pro-active protocol.

Developed at Thales Communications Norway(A. Tønnesen, A Hafslund and P. Englestad).

A plugin for olsrd exists, but it is not yet publicly released.

New nodes are assigned unused IP addresses through use of strong DAD, OLSRs MPR flooding is used to carry messages.



# Secure routing

---

OLSR itself is vulnerable to all sorts of attacks.

- A node advertises links to non-neighbor nodes.
- A node pretends to be another node.
- A node forwards altered control messages.
- A node does not forward messages as required by OLSR.
- A node forwards broadcast control messages unaltered, but does not forward unicast data traffic.
- A node replays previously recorded control traffic from another node.
- A Internet gateway can do all kinds of nasty stuff!

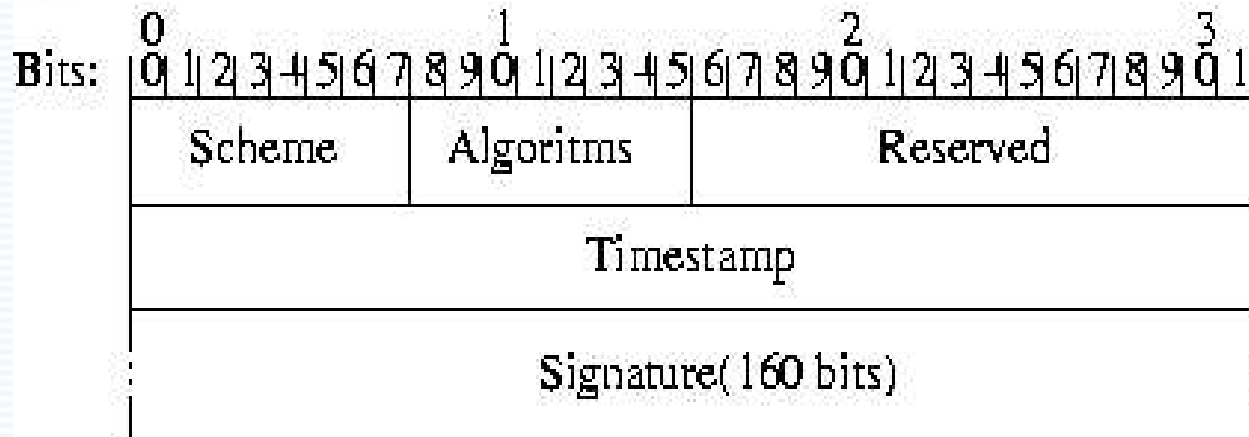


# Secure routing

Solution:

Maintain control traffic data integrity by introducing a flexible security mechanism. Could also ensure confidentiality.

All OLSR packets has a ending security message. The content is based on what scheme is used.





# Secure routing

---

Secure OLSR plugin:

Available from the next olsrd release.

A signature of the message and a shared key is added to every OLSR packet. A timestamp is also included to prevent replay attacks.

A timestamp-exchange mechanism is included. This is a three way handshake.

A key exchange and authentication mechanism could work on top of this solution.

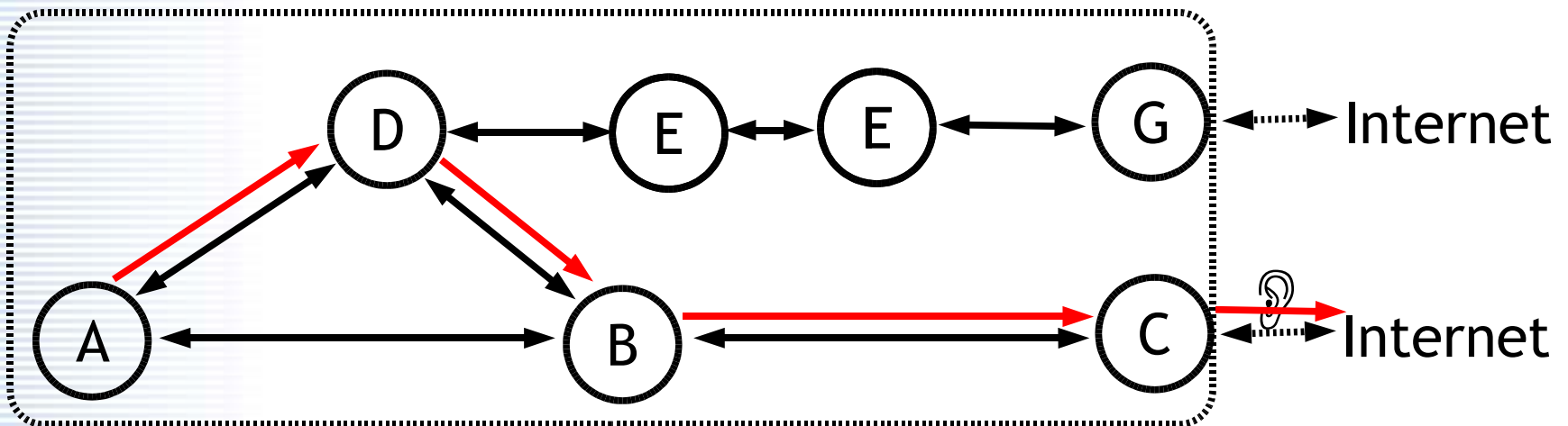


# Gateway tunneling

Background:

A node has no actual control of what gateway it uses if multiple gateways for the same network are available.

Traffic is routed hop-by-hop



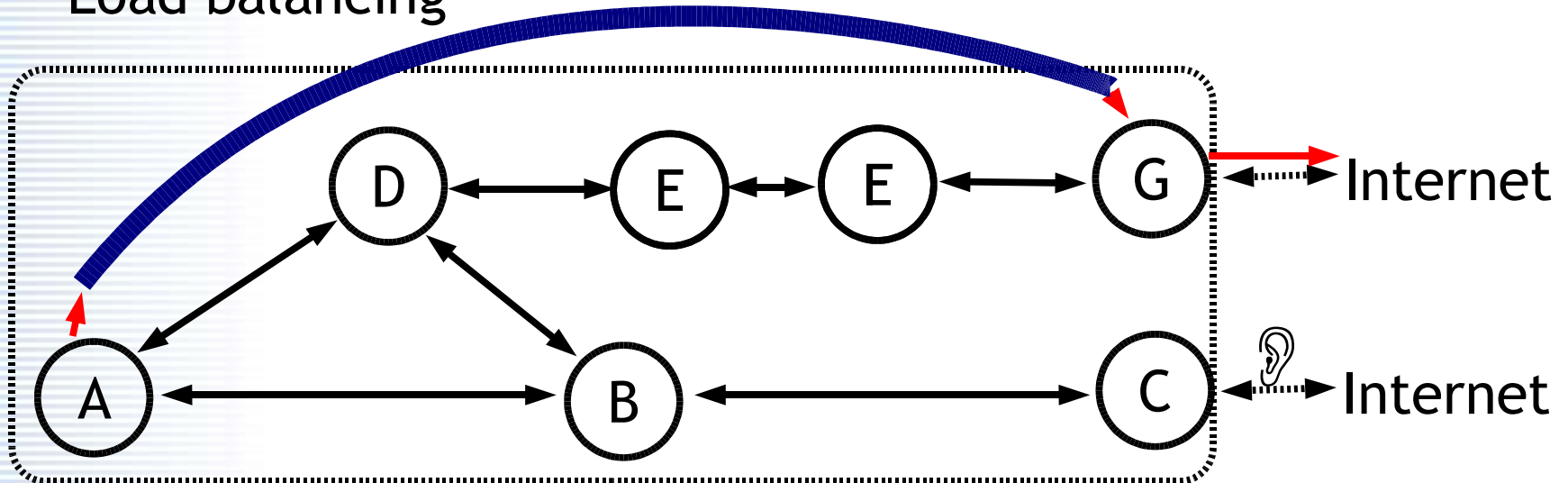


# Gateway tunneling

**Solution:** Node A sets up A IP-in-IP tunnel to the desired gateway and uses it regardless of hop-count.

Motivation:

- Avoid possible TCP session breakage
- Security
- Load balancing





# MANETs and free community networks

---

- Can “free” users be trusted to operate in a distributed manner (e.g. “cheating” in P2P networks)?
- Need for a “credit” enforcement mechanism?
- Only use MANET routing in dedicated relay stations (e.g. in cars) creating a backbone?

Trust does not scale :-)



# Further reading

---

- Unik olsrd homepage <http://www.olsr.org>
- Uppsala University AODV implementation  
<http://user.it.uu.se/~henrikl/aodv/>
- MANET IETF working group  
<http://www.ietf.org/html.charters/manet-charter.html>
- INRIA OLSR page <http://hipercom.inria.fr/olsr/>
- Protean Forge - OLSR software (CRC and NRL)  
<http://pf.itd.nrl.navy.mil/projects/olsr/>
- OpenWrt linux on WRT54G <http://openwrt.ksilebo.net/>
- Lille Sans Fil, French free community network  
<http://www.lillesansfil.org/:olsr>



# Questions?

andreto@olsr.org

[www.olsr.org](http://www.olsr.org)

Thank you - *Vielen dank!*

