Fourth EuroFGI Workshop on Wireless and Mobility

Notes on implementing a IEEE 802.11s Mesh Point

R. G. Garroppo, S. Giordano, D. Iacono, L. Tavanti

Telecommunication Networks Research Group
Dip. Ingegneria dell'Informazione
University of Pisa, Italy
http://netgroup.iet.unipi.it

16-18 January 2008, Barcelona, Spain

Presentation Outline

- Motivation and background
- Overview of IEEE 802.11s
- Architecture and features of the implemented node
- Main implementing issues
- Validation tests
- Conclusions

Motivation

- Wireless Mesh Networks (WMNs) are gaining wide popularity
  - Flexible and cost-effective alternative to set of disjointed APs
  - Many installed networks already available

- Most solutions are:
  - Based on the common IEEE 802.11 MAC/PHY (access)
  - Proprietary and not-interoperable (backbone and applications)

- The IEEE Task Group 11s is trying to define a standard
  - A rather troublesome birth, no official draft released
  - Proposals are still being submitted to TGs

Motivation

- Proposed solutions were evaluated through analysis or simulation only
  - There is no experimental evidence of their goodness
  - Analysis/simulation results are sometimes deeply different from reality
- An experimental testbed could allow…
  - More trustworthy feedbacks from the evaluation tests
  - More confidence in the suitability of the proposals
  - More effective standards
- Implementation of a 802.11s Mesh Point
  - Test and amend the draft as it evolves into the standard

IEEE 802.11s Mesh

- IEEE 802.11s builds on other amendments to the standard:
  - 802.11a/b/g/n for the physical interface
  - 802.11e for accessing the medium
  - 802.11i for security

- …but it also conceives a new network architecture
  - Nodes shall build multi-hop paths to form a fully connected network
  - New mechanisms to configure and operate the Mesh
  - New frame formats

- The Mesh is expected to be transparent to upper layers
  - Routing is performed at link layer (path selection and forwarding)
  - Nodes in the Mesh shall all be seen as one-hop away by L3 services

IEEE 802.11s Mesh

- The basic entity is the Mesh Point (MP)
  - Same features of legacy 802.11 stations
  - Called to set-up and maintain multi-hop paths to relay traffic
    - Form a wireless backbone (Mesh)

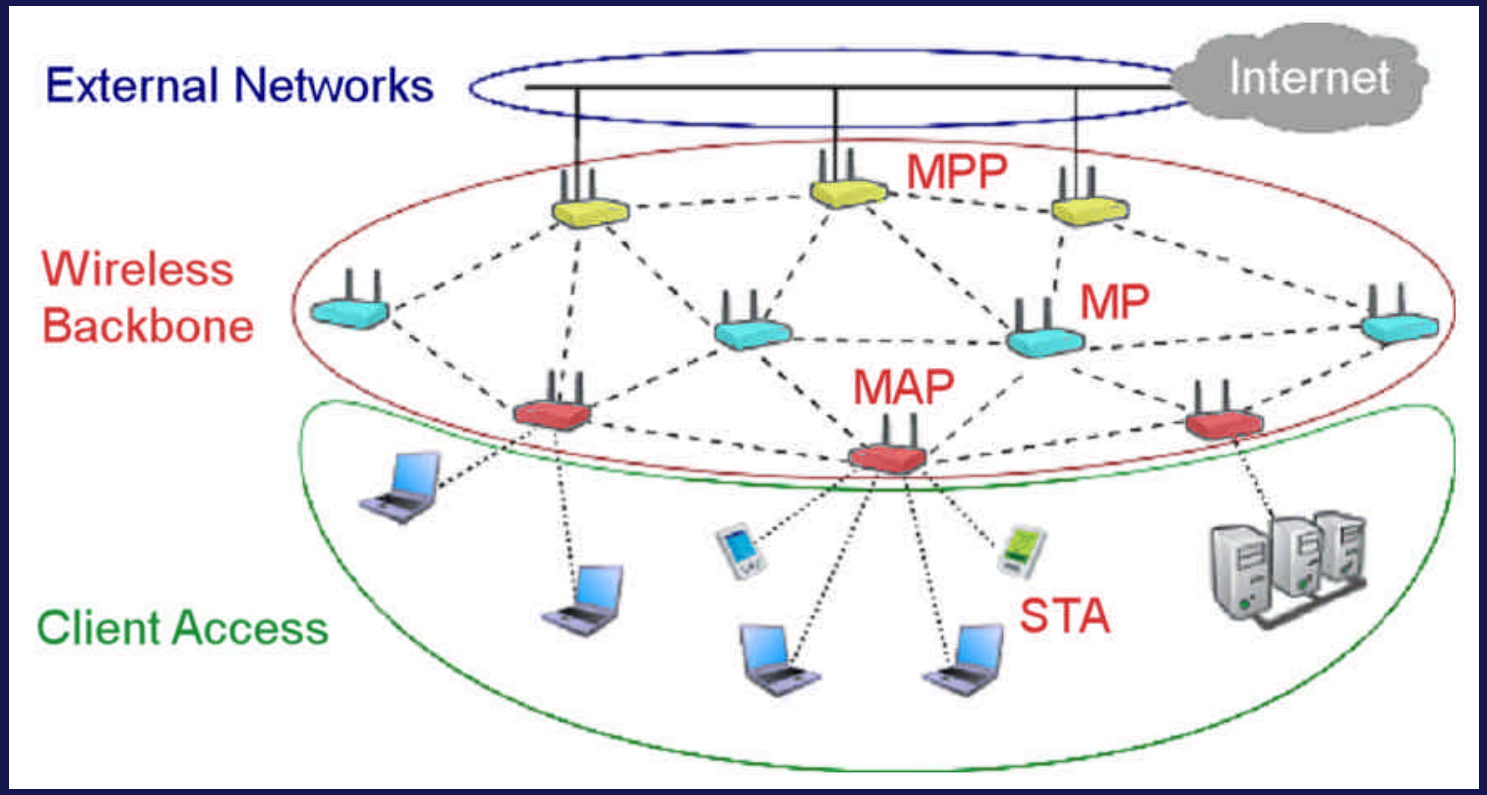
- A MP can also act as an AP ⇨ Mesh Access Point (MAP)
  - Give stations access to network services

- A MP can also act as a Portal ⇨ Mesh Point plus Portal (MPP)
  - Gateway to other networks (e.g. Internet)

- Stations (STAs) are allowed to work with legacy 802.11 cards
  - STAs should have no knowledge of the Mesh

IEEE 802.11s Mesh

- An example 802.11s Mesh network



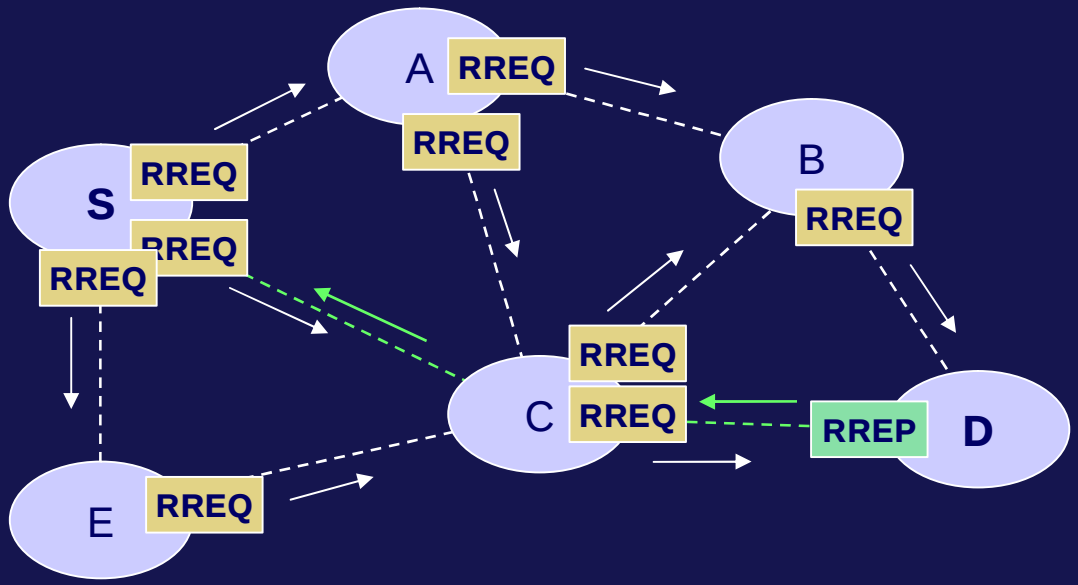
IEEE 802.11s Mesh

- Default path selection and forwarding protocol is HWMP (Hybrid Wireless Mesh Protocol)
  - Combines on-demand routing with a proactively-built tree topology

- On-demand routing is very similar to AODV
  - A source Mesh Point S wanting to send data to a destination MP D broadcasts a RREQ (Route Request)
  - Intermediate nodes store the address of the MP they received the RREQ from and re-broadcast the RREQ
  - When the RREQ reaches D, it replies with a unicast RREP (Route Reply) addressed to S
  - The RREP follows a reverse path to S and the path is established
  - The flags DO (Destination Only) and RF (Reply and Forward) can change the behaviour of the intermediate nodes
  - A RERR (Route Error) frame is used when a failure occurs

IEEE 802.11s Mesh

- The HWMP on-demand algorithm:



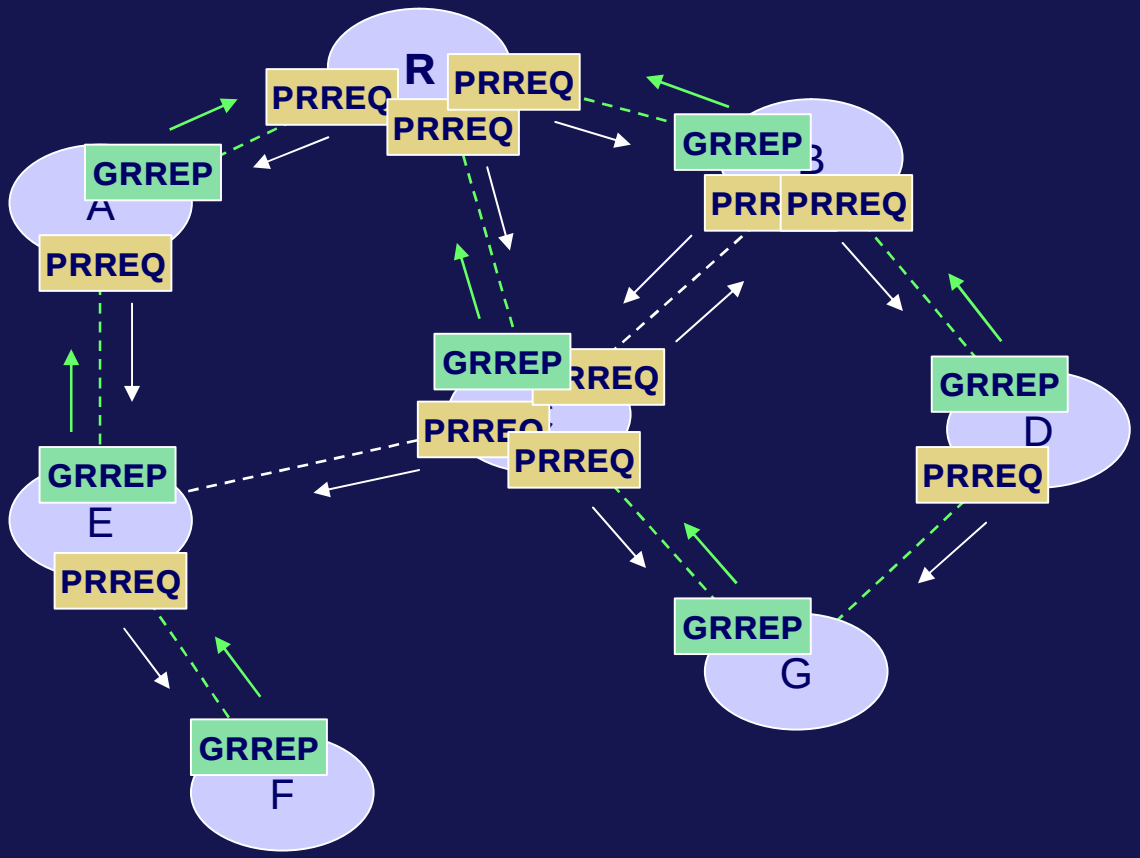
IEEE 802.11s Mesh

- The HWMP tree connects all MPs to the Portal, which is the root
  - A path to/from the outside is always available
  - Broadcast and flooding can be reduced
  - On-demand paths can use the tree links as back-up paths

- The tree can be set up in two ways:
  - The Portal broadcasts Proactive RREQ (PRREQ) frames, and every MP replies with a Gratuitous RREP (GRREP)
    - This technique creates and maintains the entire tree
  - The Portal broadcasts Portal Announcement (PANN) frames, leaving each MP the possibility to set up the path whenever it need it in an on-demand fashion

IEEE 802.11s Mesh

- The HWMP tree:



IEEE 802.11s Mesh

- To select the best paths, 802.11s defines the airtime metric:
  - $C_a = (O+L/r)/(1-e)$
  - MPs continuously monitor their links and exchange metric values with neighbors

- New frame formats: Mesh Management and Mesh Data
  - Extended type
  - Hold a Mesh Header, i.e. 4 or 16 bytes borrowed from the data field
    - used for two extra addresses to forward frames generated by user stations

- Changes to existing frames
  - New Information Elements (IEs) in the data field of the Management frames

IEEE 802.11s Mesh

- Station management
  - Every MAP shall act as a Proxy for its associated STAs
    - Stations do not have awareness of the Mesh!
  - Every MP shall keep a list of all the STAs in the network and the address of their MAP Proxy
    - Proxy Update (PU) and PU Confirmation (PUC) frames

- Other features
  - Mesh Deterministic Access (MDA): a distributed scheduling algorithm based on the reservation of contention free time slots
  - Multi-channel framework
  - Individual and Group addresses (multicast)

Presentation Outline

- Motivation and background
- Overview of IEEE 802.11s
- Architecture and features of the implemented node
- Main implementing issues
- Validation tests
- Conclusions

Implementation

To build our prototype 802.11s Mesh Point…

- We used common hardware (laptop PCs with Atheros cards) and open source software (Linux)
- We realised a software framework partly within the card driver (data frames handling) and partly on top of it (management functions)
  - The same 802.11s defines the new Mesh services as a dedicated logical interface, independent from the legacy MAC functions
  - 802.11s is made of a rather complex set of features, many of which are still in evolution
- We mostly dealt with path selection and frame forwarding

Implementation

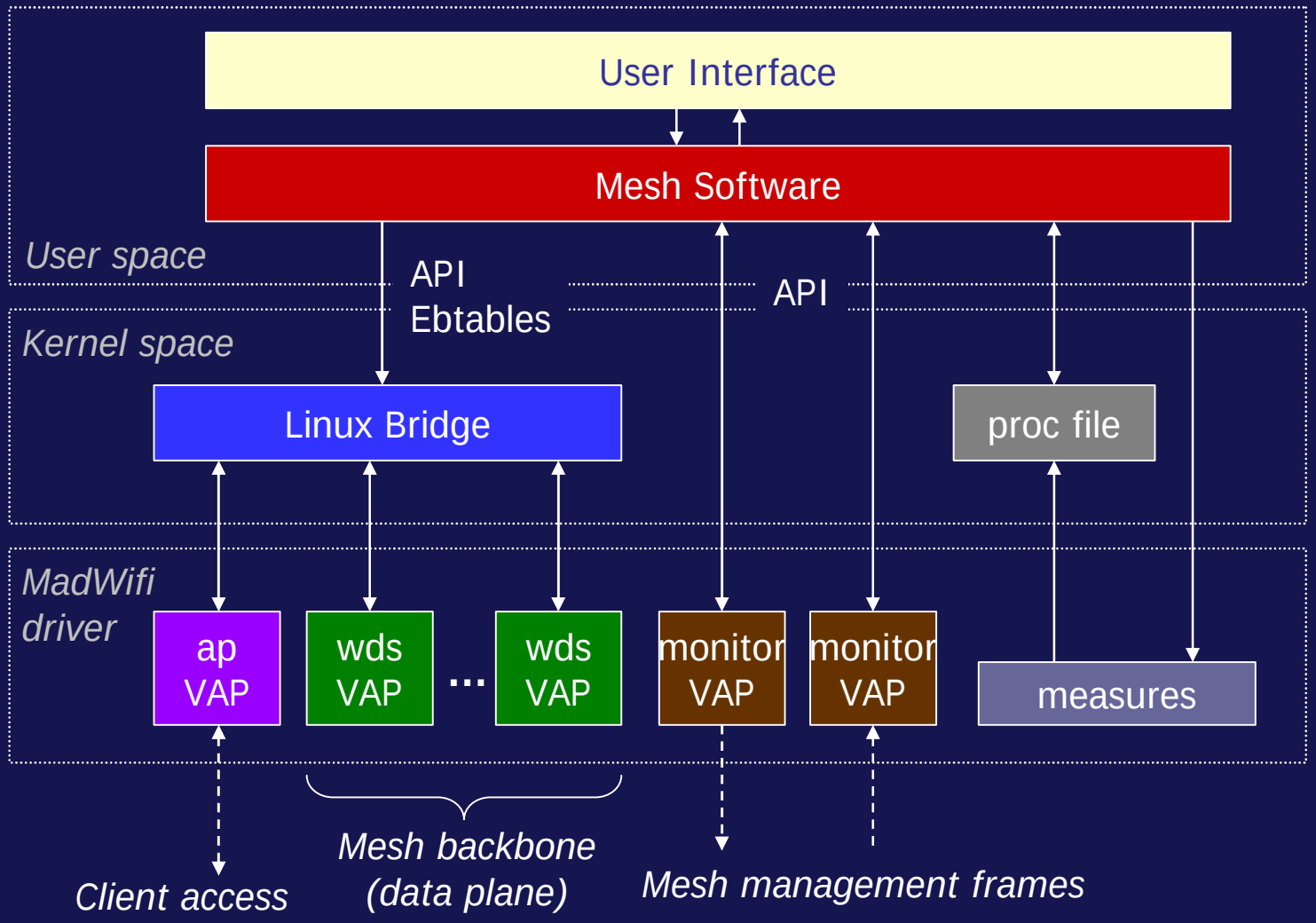
Software Tools:

- The Multiband Atheros Driver for Wi-Fi (MadWifi) allows creating several Virtual Access Points (VAPs)
  - Each VAP appears to the O.S. as different wireless interface
  - Frames entering or exiting the device can undergo a different set of processing rules
  - VAP modes: ap, sta, wds, monitor, etc.

- The Linux bridge has been used to connect the interfaces created by MadWifi.
  - This module behaves exactly like a hardware bridge, forwarding on its ports the frames according to their destination MAC address.
  - Each port of the bridge can be either blocked or open (learning).

Implementation

Architecture of the Mesh Point software framework:



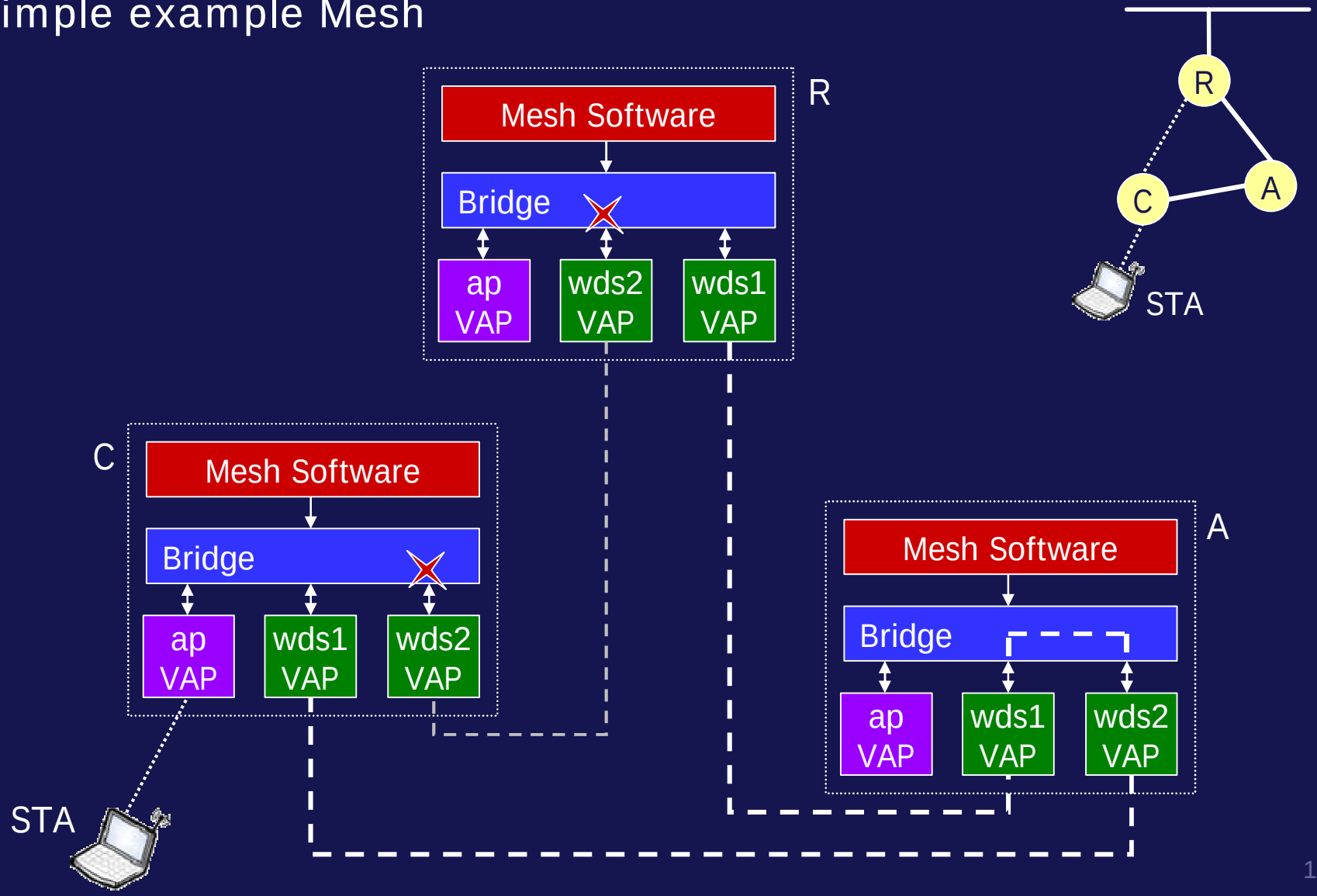
Implementation

- The framework implements two functional levels:
  - The management plane, run at user space, is made of all the procedures to build and maintain the Mesh, exploits the *monitor* VAPs and controls the behavior of the bridge
  - The data plane works entirely at kernel space (Linux bridge and *ap* and *wds* VAPs) to handle the data frames; it exploits the paths set by the management plane

- Each *wds* VAP is connected to its homologue in a neighbor MP
  - The network appears as a series of *wds* LAN segments
  - Mesh paths are formed by enabling the ports that are connected to MPs that are parent or children in the tree built by the HWMP proactive mode (and closing the others)

Implementation

A simple example Mesh



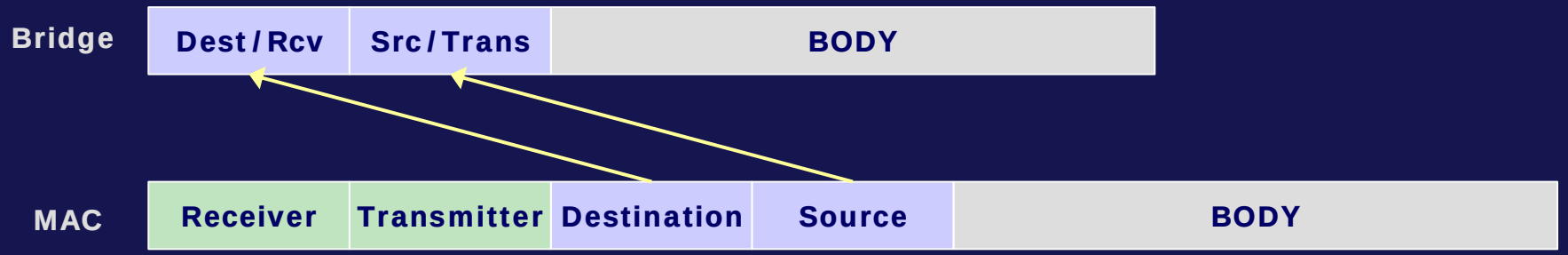
Presentation Outline

- Motivation and background
- Overview of IEEE 802.11s
- Architecture and features of the implemented node
- Main implementing issues
- Validation tests
- Conclusions

Implementing issues

The Linux bridge and the six-address frame format

- The bridge only knows two-address Ethernet frames (one-hop)
- Multi-hop paths use four/six addresses (multi-hop)
- The driver maps 802.11 frames to Ethernet frames:



- The bridge is not aware of hosts more than one hop away

  ⇨ It cannot properly forward frames!
- We can “instruct” it with frames in which the address of the source (of the end-to-end path) appears
  - E.g. RREQ, ARP, etc.

Implementing issues

The new 802.11s frames

- Mesh Management frames were created and used by the Mesh Software module through the Monitor VAPs

- …but modified Data Frames…
  - could not be handled by the Mesh Software module
    - Processing is too slow at User space
  - could not be handled by the driver
    - A deep revision of the driver was not feasible

⇨ Problem-specific solutions

Implementing issues

Station management

- 11s requires all MPs to be aware of all STAs in the Mesh
  - Overhead may become a meaningful burden

⇨ Put the knowledge of the whole set of STAs on the Portal
  - Coherent with realistic scenarios
  - No performance degradation

- Mesh Management frames are hardly useable for handling STAs
  - Designed for communications between MPs

⇨ Added an extra address field to account for the station the MAP acts on behalf of
  - MAP handles and converts the frames coming from / going to STAs

Implementing issues

Broadcast frames

- Many broadcasts limit network performance

⇨ Broadcasts management frames from the stations are converted by MAP into frames addressed to the Portal

  – Their aim is usually getting some information from the Portal (e.g. ARP messages)

⇨ Broadcasts generated by the Portal constrained to the HWMP tree

  – All network functionalities have been retained

Implementing issues

Improvements to HWMP

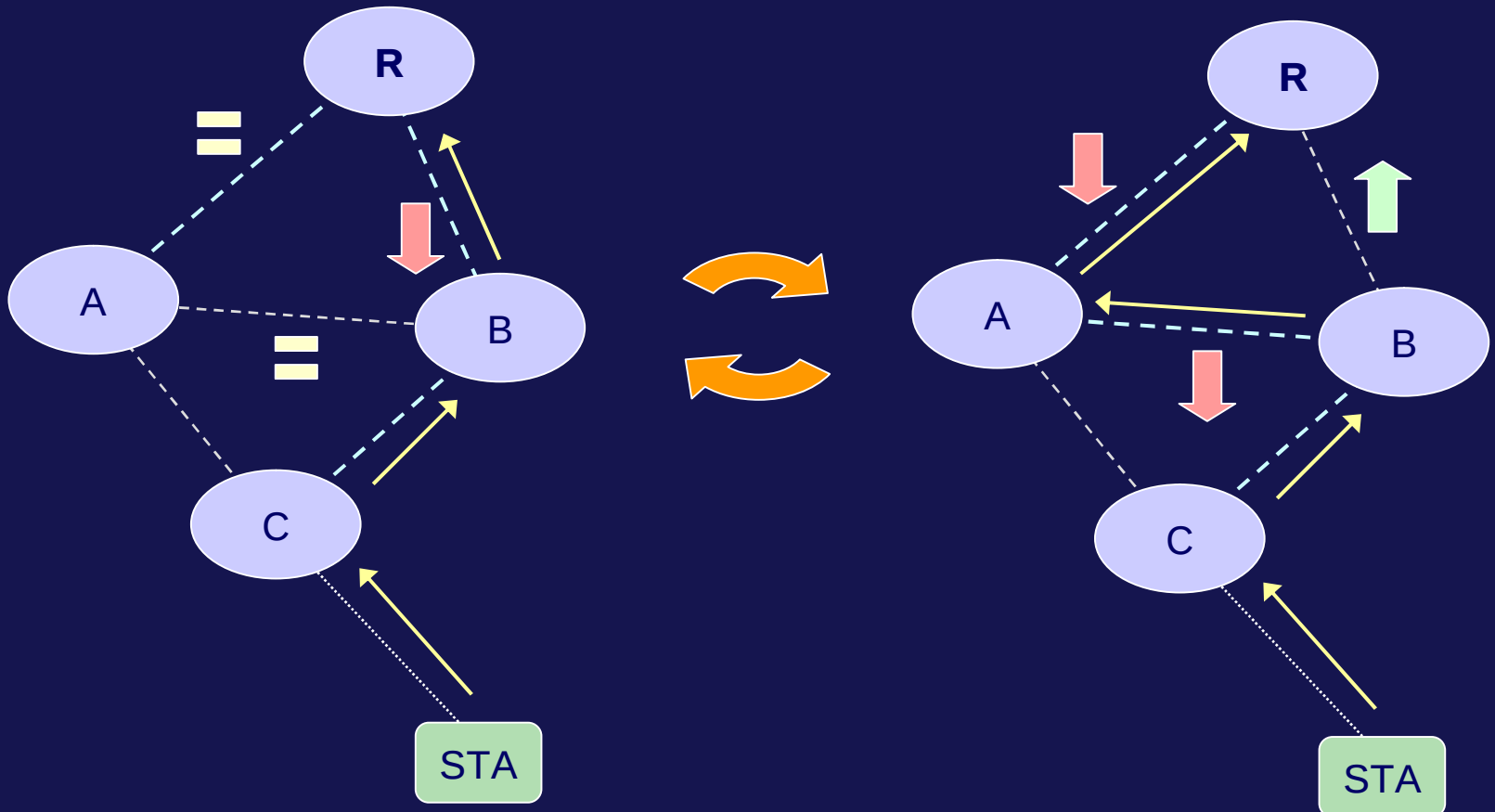
- Not always is the first received RREQ the one coming from the shortest/best path (access is CSMA/CA)
  - ⇨ After receiving a RREQ, MP waits a short time to receive RREQs from different paths
    - MP re-broadcasts only the PRREQ received from the neighbor closest to the root
    - MP merges the RREQ and re-broadcasts only one RREQ
    - ⇨ Reduced overhead and increased protocol stability

- 802.11s dictates that the complete list of dependent nodes is inserted in the GRREP
  - ⇨ We put just the direct children of the MP
    - Reduce the protocol overhead
    - Better knowledge of the topology of the Mesh

Implementing issues

The Airtime metric

- Very sensitive to link usage
  - Sort of ping-pong effect among paths!



Implementing issues

The Airtime metric

- Very sensitive to link usage
  - Sort of ping-pong effect among paths!

⇨ Change the metric definition

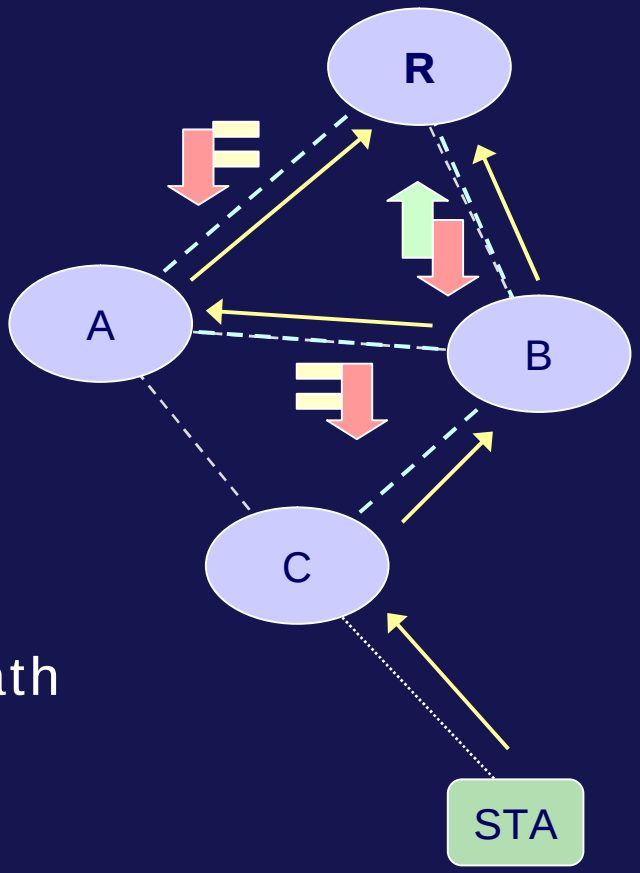
$$C_a = (k_1 + k_2/SNR)/(1-e)$$

  - Lower dependence on link congestion

⇨ Moving average on more updates

⇨ More than one PRREQ before changing path

  - Increase path stability



Presentation Outline

- Motivation and background
- Overview of IEEE 802.11s
- Architecture and features of the implemented node
- Main implementing issues
- Validation tests
- Conclusions

Validation tests

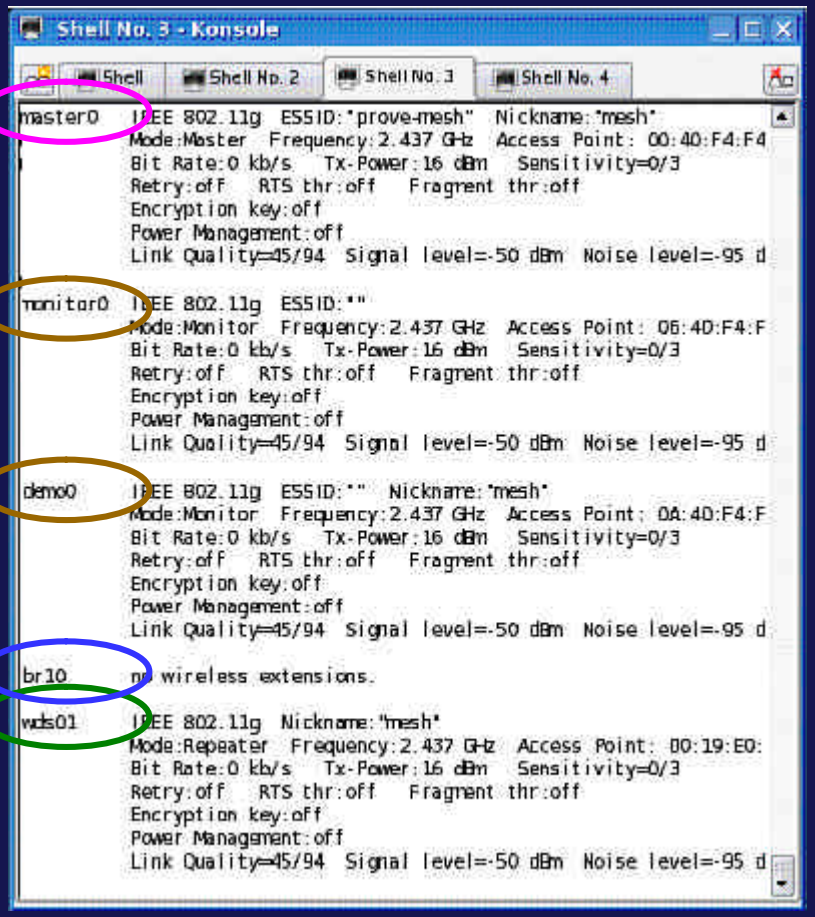
- We assumed that
  - a HWMP tree is built with PRREQ/GRREP
  - on-demand paths used for communications among stations

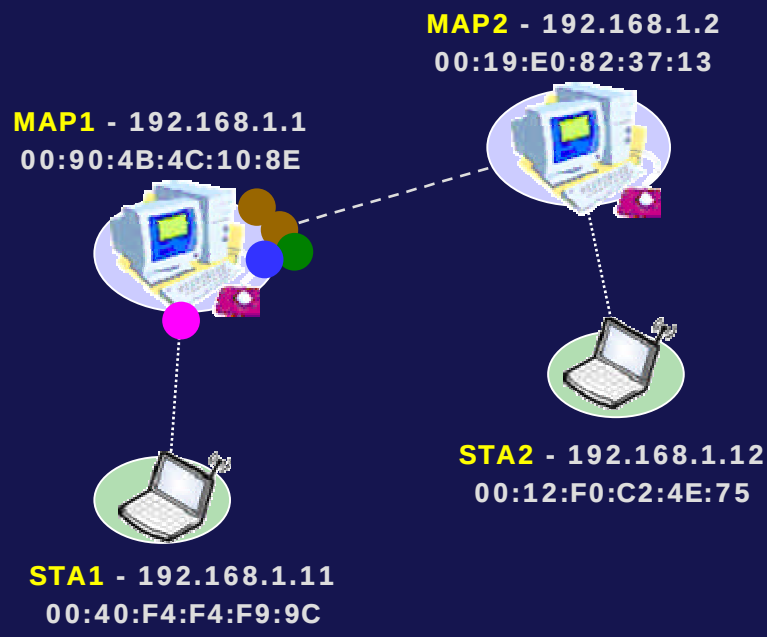
- A working example: ARP
  - A station begins a web session with an ARP request (broadcast)
  - The ARP request is mapped by the Proxy MAP into a unicast frame, with destination=MPP and source=STA
  - The bridges of the MPs on the HWMP tree…
    - learn the association incoming port – STA address
    - forward the frame on the port to the MPP (known from PRREQs)
  - The Portal sends the ARP reply
  - The bridges of the MPs on the HWMP tree now knows on which port the ARP reply must be sent

Validation test #1

Creation of an on-demand path between two stations

- MAPs act at layer 2 only (IP addresses just for debug)
- STAs act at layer 3 (IP addresses are necessary)



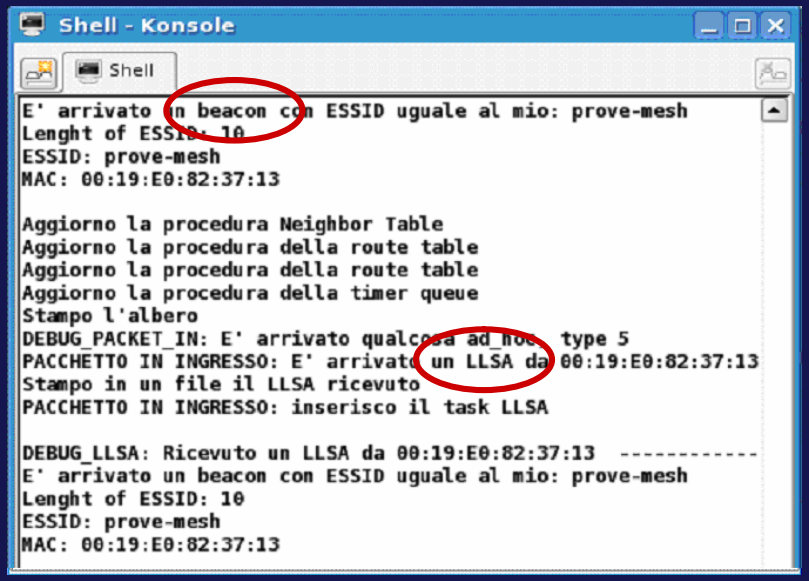


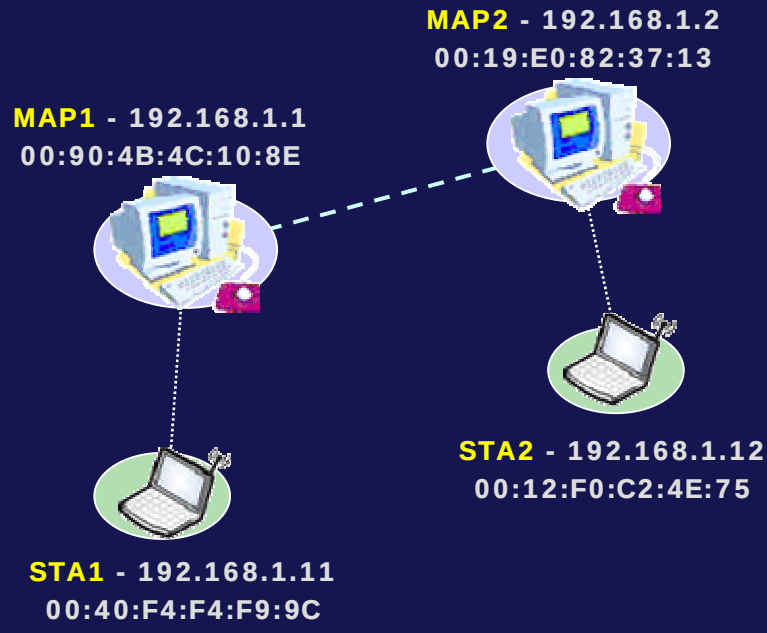
Validation test #1

Creation of an on-demand path between two stations

- MAPs act at layer 2 only (IP addresses just for debug)
- STAs act at layer 3 (IP addresses are necessary)

- MAP1 and MAP2 connect to each other





Validation test #1

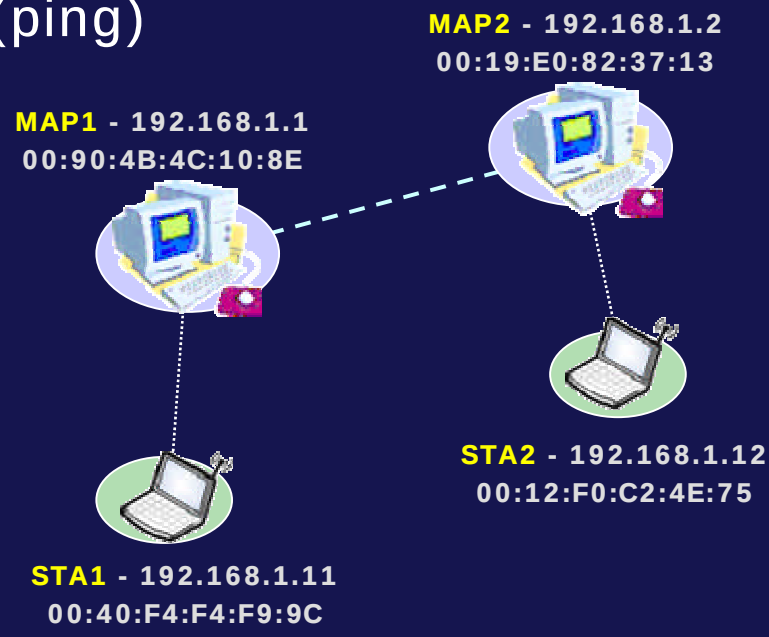
Creation of an on-demand path between two stations

- MAPs act at layer 2 only (IP addresses just for debug)
- STAs act at layer 3 (IP addresses are necessary)

- MAP1 and MAP2 connect to each other
- STA1 wants to set up a path to STA2 (ping)
  - ARP Request
    - STA1 → MAP1 → MAP2 replies

arp **MAP1**

IP address	HW type	Flags	HW address
192.168.1.11	0x1	0x2	00:40:F4:F4:F9:9C
192.168.1.2	0x1	0x2	00:19:E0:82:37:13
192.168.1.12	0x1	0x2	00:00:00:00:00:00
131.114.52.1	0x1	0x2	00:01:30:18:8F:00

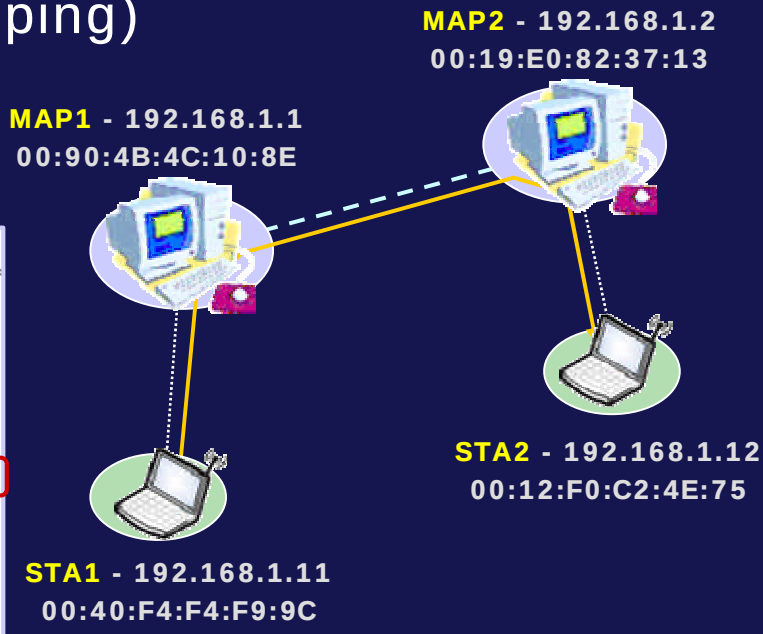


Validation test #1

Creation of an on-demand path between two stations

- MAPs act at layer 2 only (IP addresses just for debug)
- STAs act at layer 3 (IP addresses are necessary)

- MAP1 and MAP2 connect to each other
- STA1 wants to set up a path to STA2 (ping)
  - ARP Request
    - STA1 → MAP1 → MAP2 replies
  - RREQ/RREP




```
RREP ricevuto
=====================================

Tempo: 17.255138

id: 2
Flag proxy: 1
hop count:: 0
TTL:: 32
Destinatario RREP  MAC : 00:40:F4:F4:F9:9C
Seq. number destinatario: 0
Proxy MAC : 00:90:4B:4C:10:8E
Lifetime: non implementato
Path Metric: 0
Originator  MAC : 00:12:F0:C2:4E:75
Originator seqn: 2
```

STA1

MAP1

STA2

Shell - Konsole

Shell | Shell No. 2 | Shell No. 3 | Shel

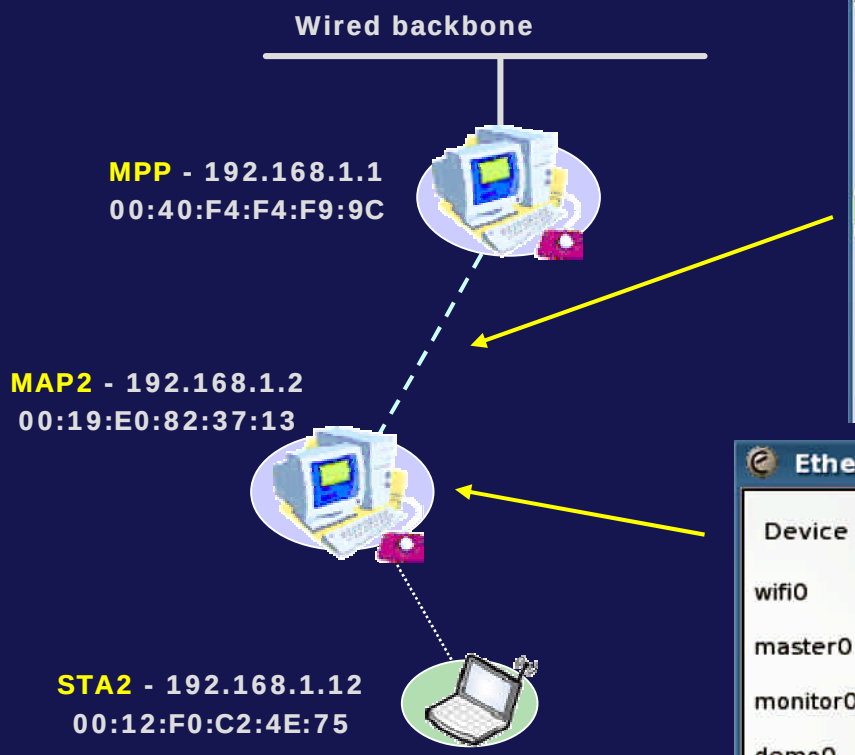
MAP1

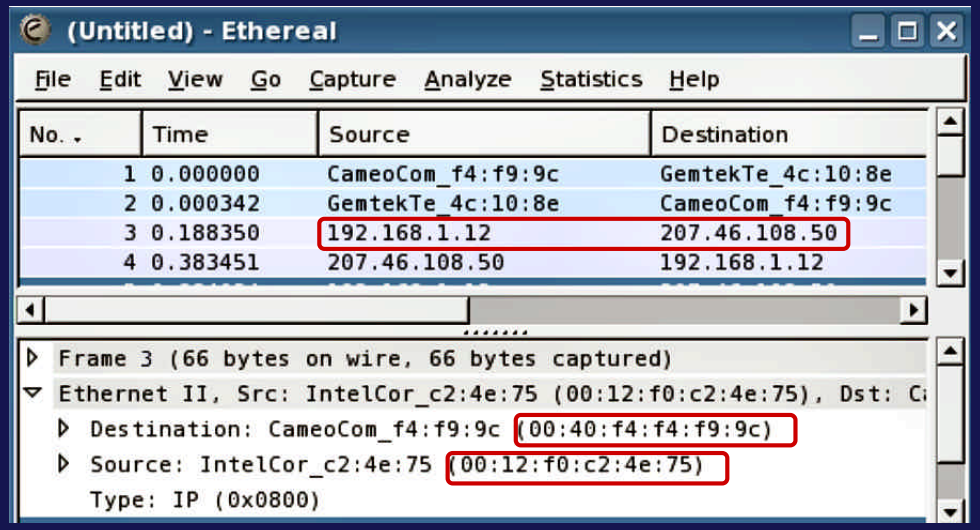
```
DEBUG_ARP: ip è 192.168.1.12
DEBUG_ARP ipadd è 192.168.1.12
DEBUG_ARP corrispondenza trovata 192.168.1.12
DEBUG_ARP: genero un proxy RREQ per raggiungere  00:12:F0:C2:4E:75
DEBUG_ARP: genero un proxy RREQ per raggiungere 192.168.1.12
DEBUG_ARP: Proxy address: 00:90:4B:4C:10:8E
```

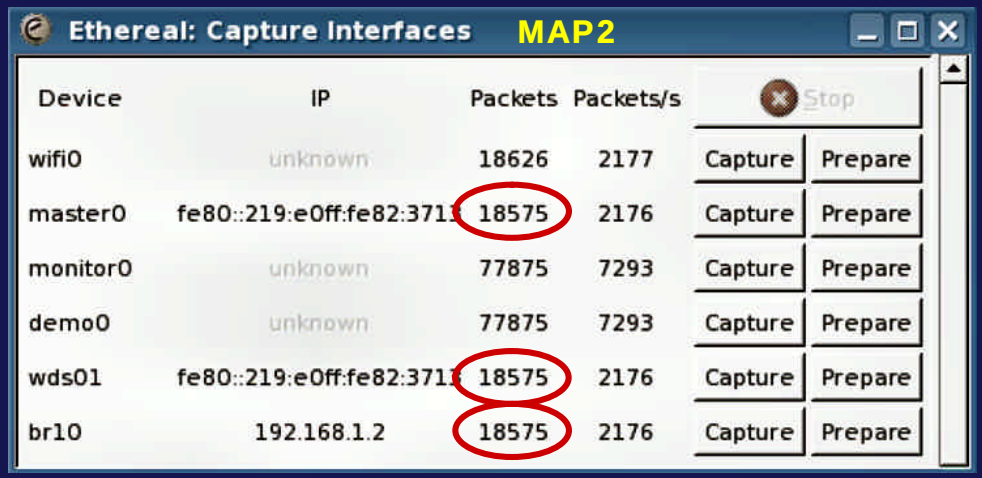
Validation test #2

Connection to an external network

- STA2 connects to a web site



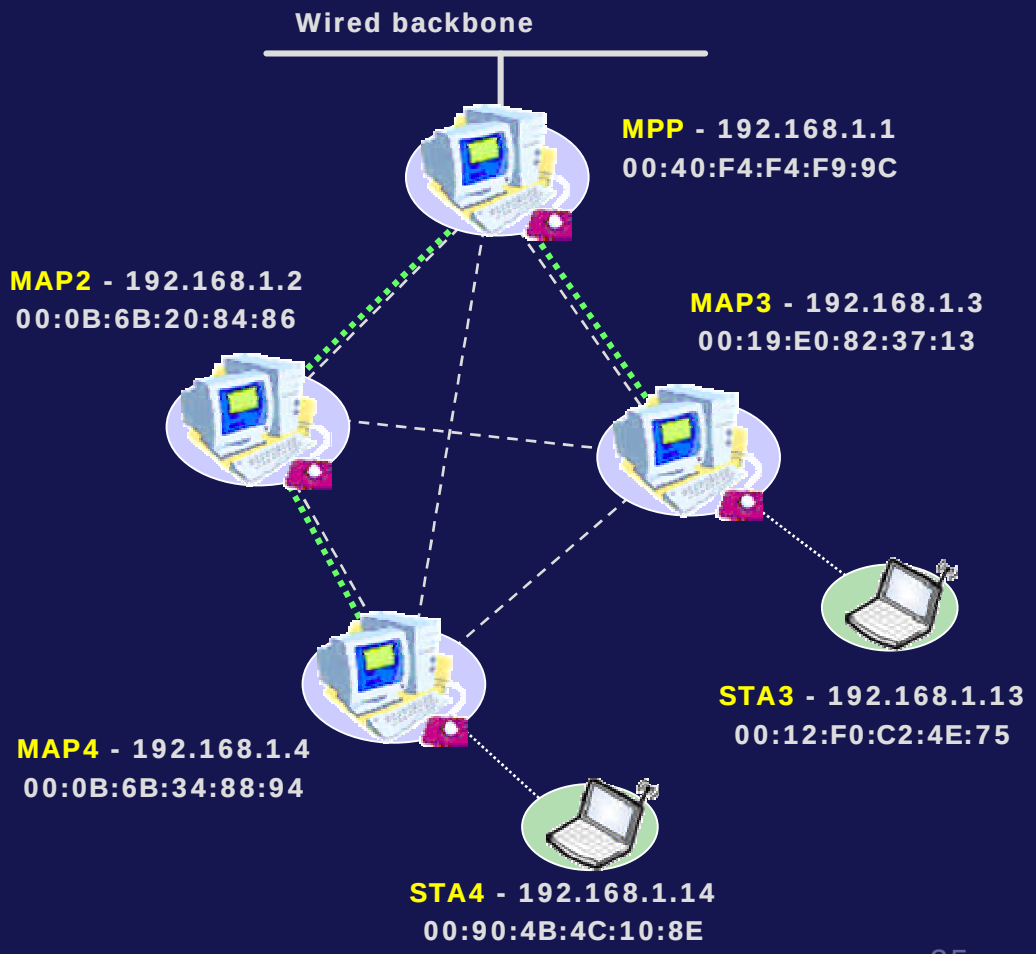




Validation test #3

Mixed scenario: HWMP tree plus on-demand

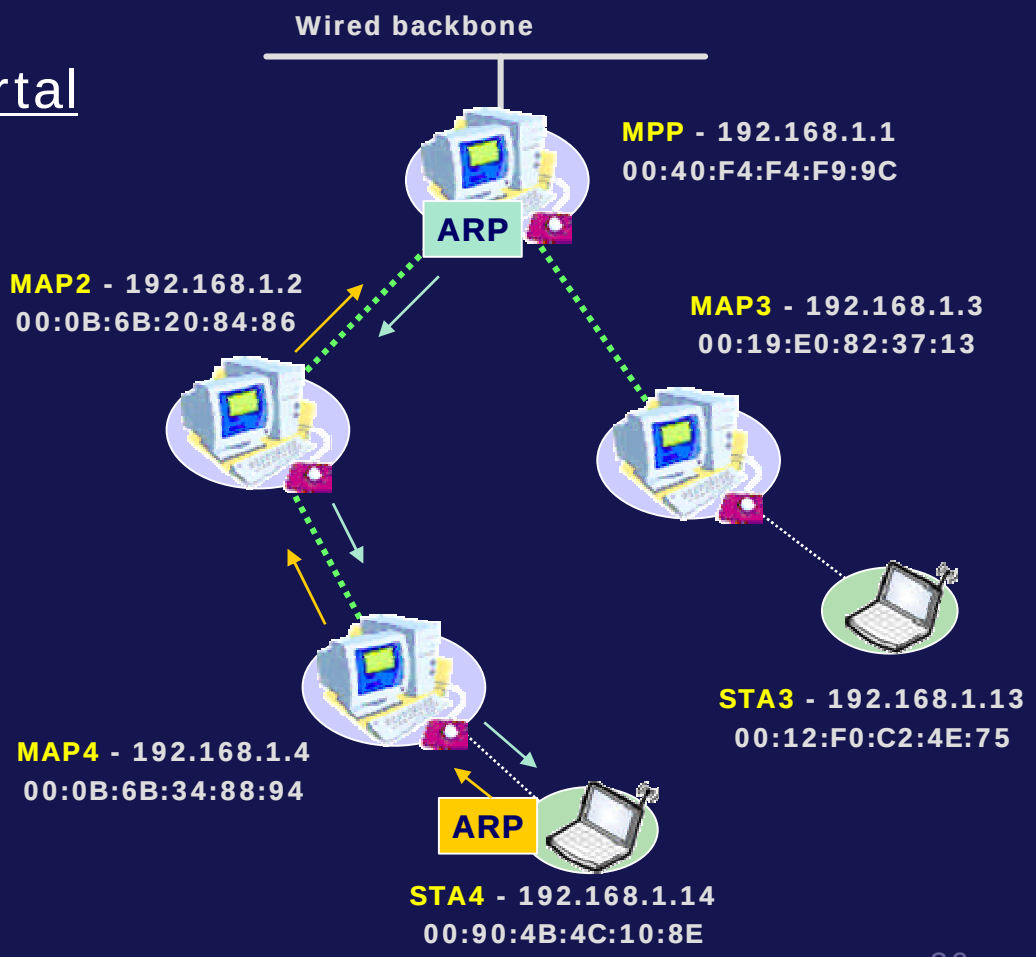
- The tree is built (PRREQ/GREEP exchange)



Validation test #3

Mixed scenario: HWMP tree plus on-demand

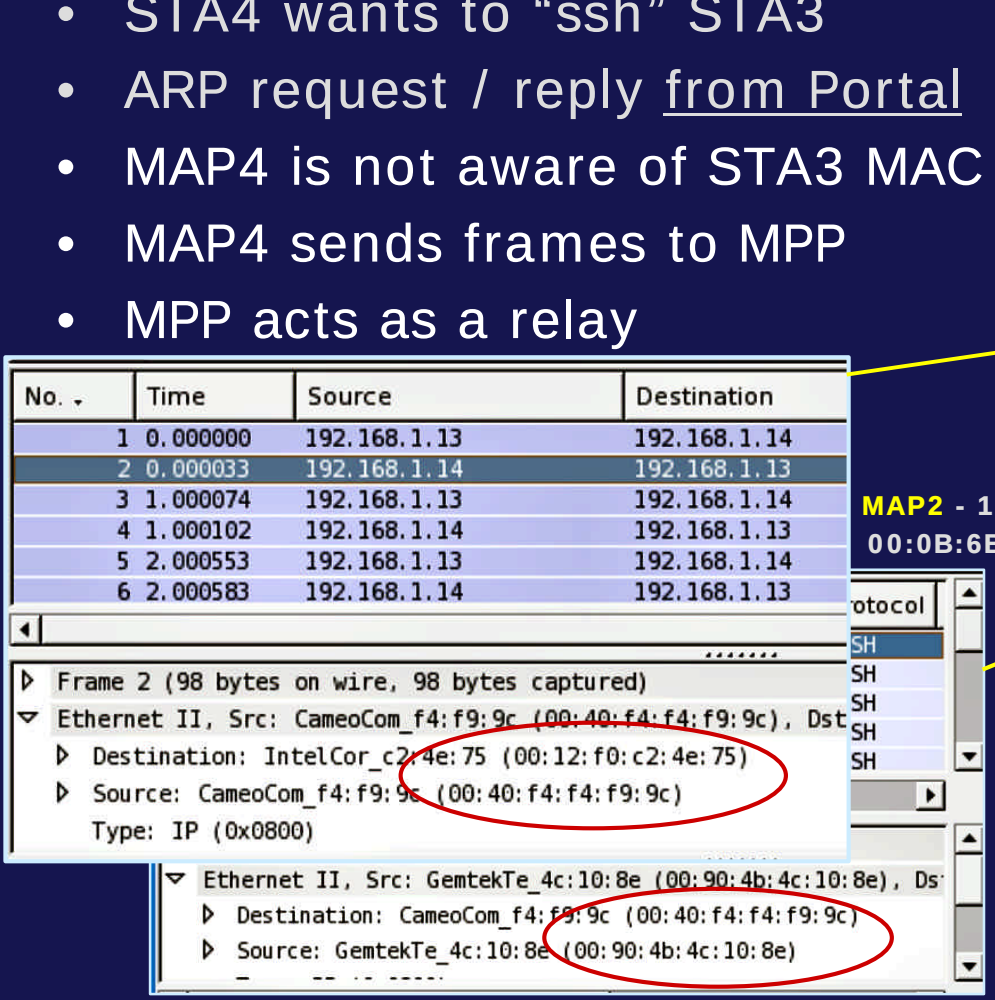
- The tree is built (PRREQ/GREEP exchange)
- STA4 wants to “ssh” STA3
- ARP request / reply <u>from Portal</u>

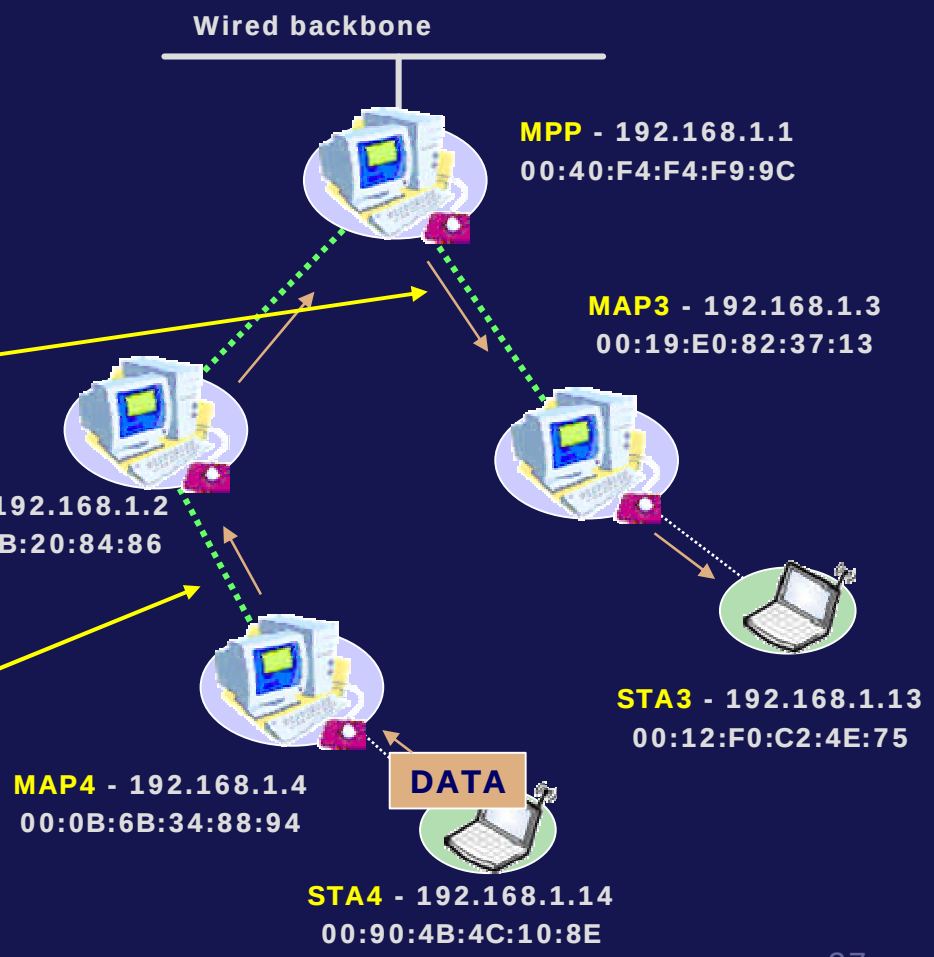


Validation test #3

Mixed scenario: HWMP tree plus on-demand

- The tree is built (PRREQ/GREEP exchange)
- STA4 wants to “ssh” STA3
- ARP request / reply from Portal
- MAP4 is not aware of STA3 MAC
- MAP4 sends frames to MPP
- MPP acts as a relay

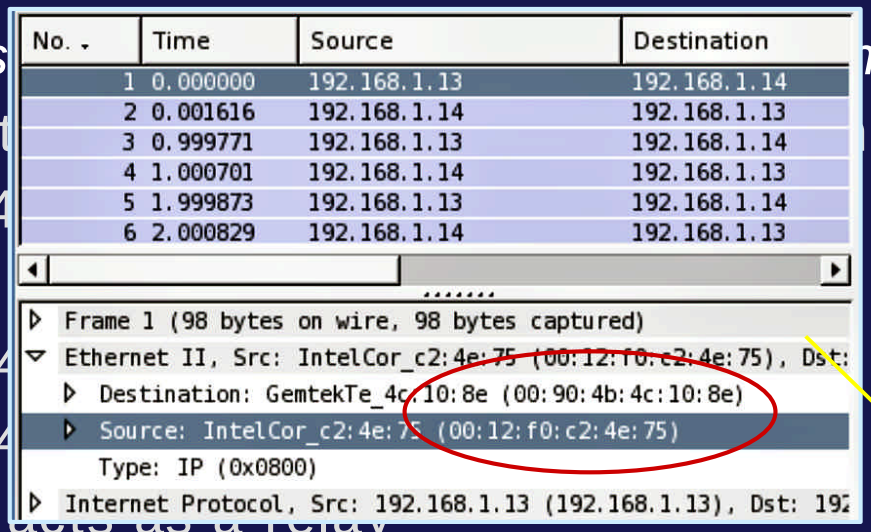


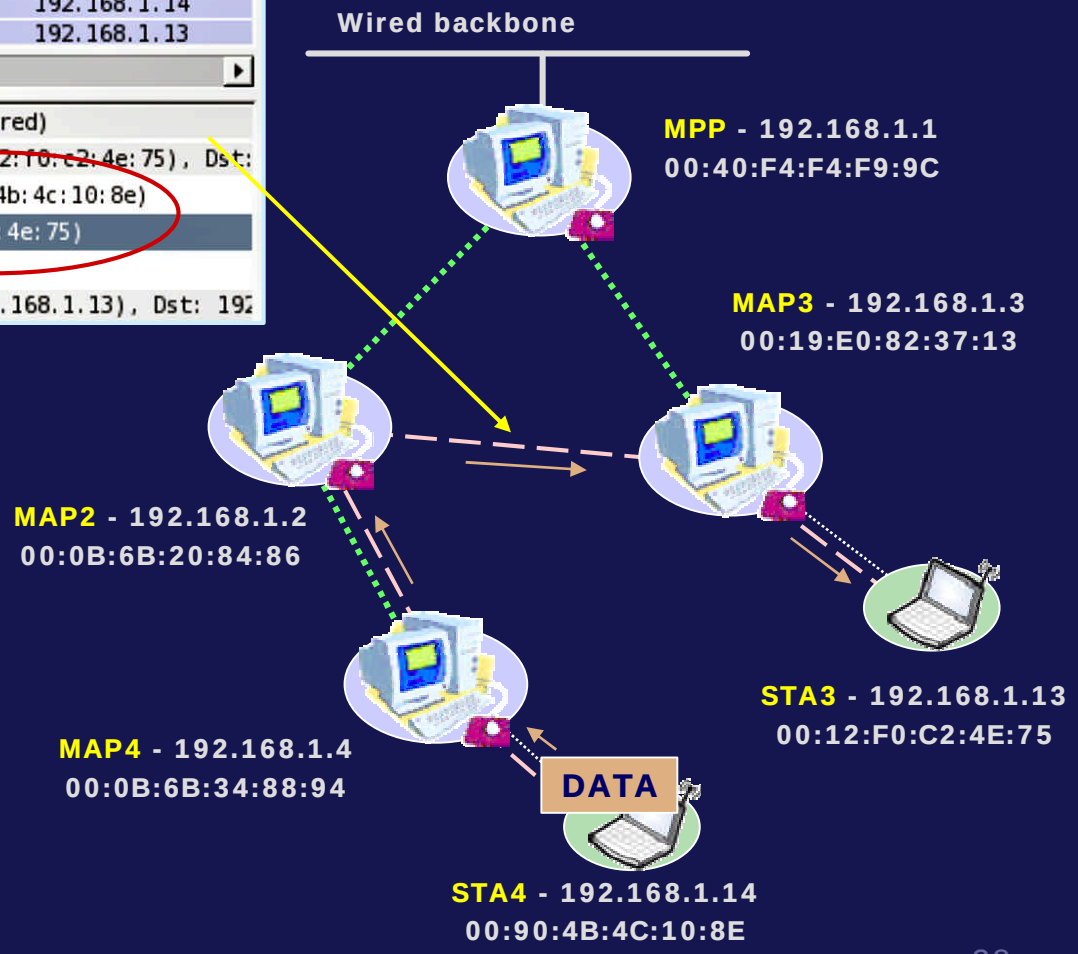


Validation test #3

Mixed s *mand*

- The t *ange)*
- STA4
- ARP
- MAP4
- MAP4
- MPP acts as a relay
- Not the optimal path!
- MAP4 starts RREQ
- MAP3 sends RREP
- On-demand path is set up





Conclusions

- We built a prototype IEEE 802.11s Mesh Point
  - Using common hw/sw
- Useful for testing existing and new functionalities
- Revealed some shortcomings/flaws
  - The Airtime metric may create “ping-pong” effects among paths
  - Station management needs to improved
    - Enhancing the use of Proxy MAP
    - Using more complete frames
    - Letting the Portal only have knowledge of all STAs
  - HWMP can also be improved
    - E.g. PRREQ/GRREP
- 802.11s seems to underestimate that stations, i.e. users, are the main source/sink of traffic