# OMNeT++ Community Summit, 2015

# Beyond INET 3.0

IBM Research - Zurich, Switzerland – September 3 - 4, 2015

Levente Mészáros

# Overview

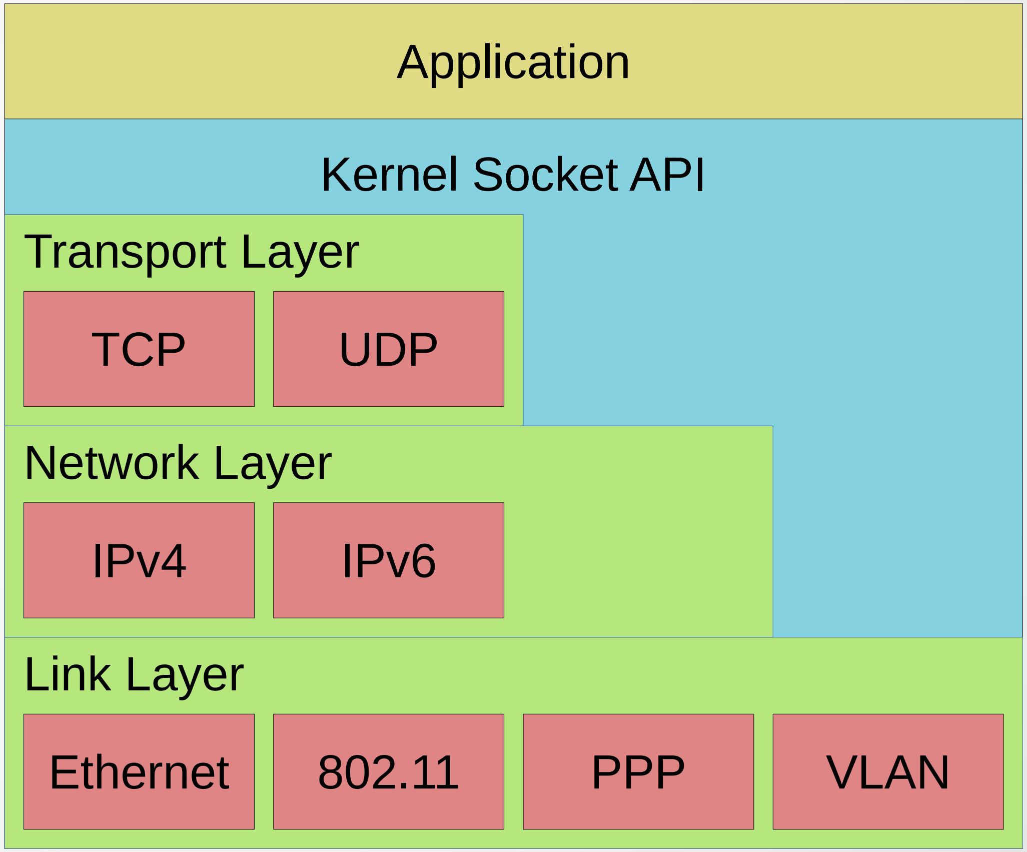
Network node architecture refactoring

Cross-layer communication and optimization

Mobility refactoring

# Real World Network Node Architecture

- Real world applications often use different kind of sockets and protocols simultaneously



Application

Kernel Socket API

Transport Layer

TCP

UDP

Network Layer

IPv4

IPv6

Link Layer

Ethernet

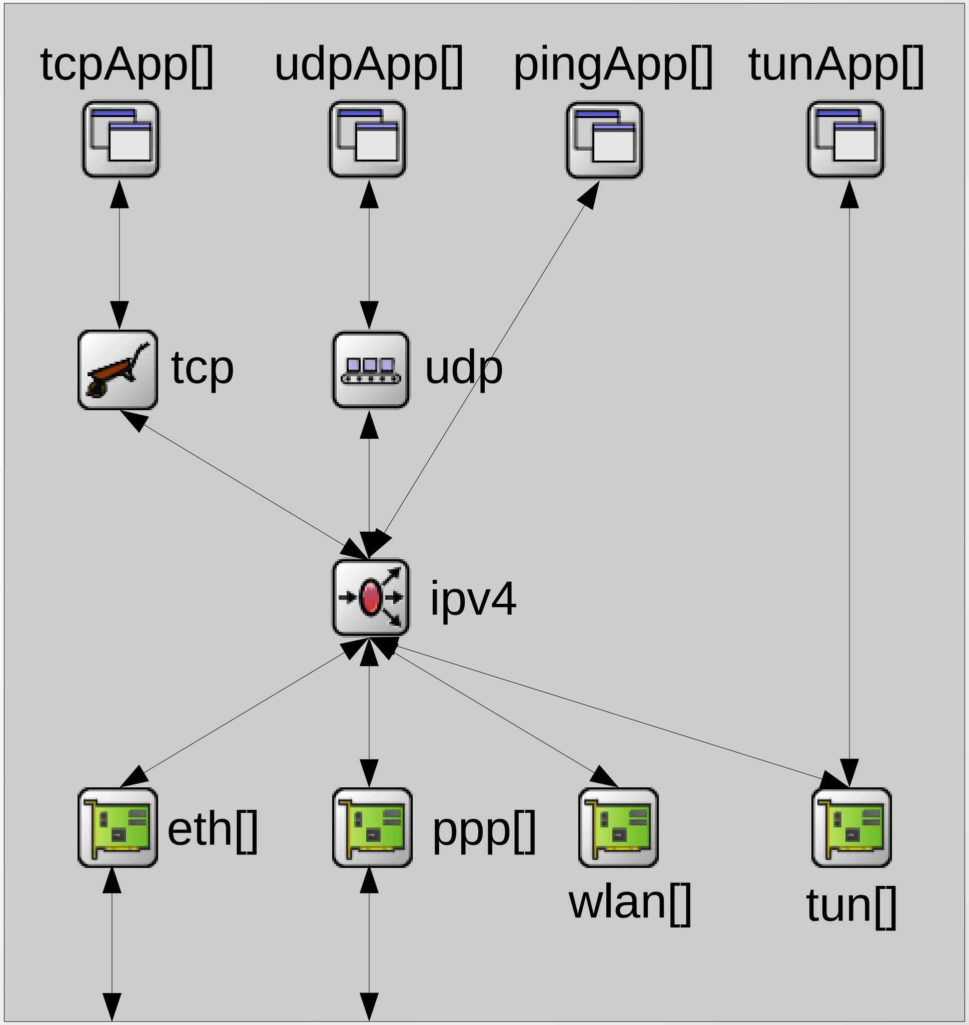
802.11

PPP

VLAN

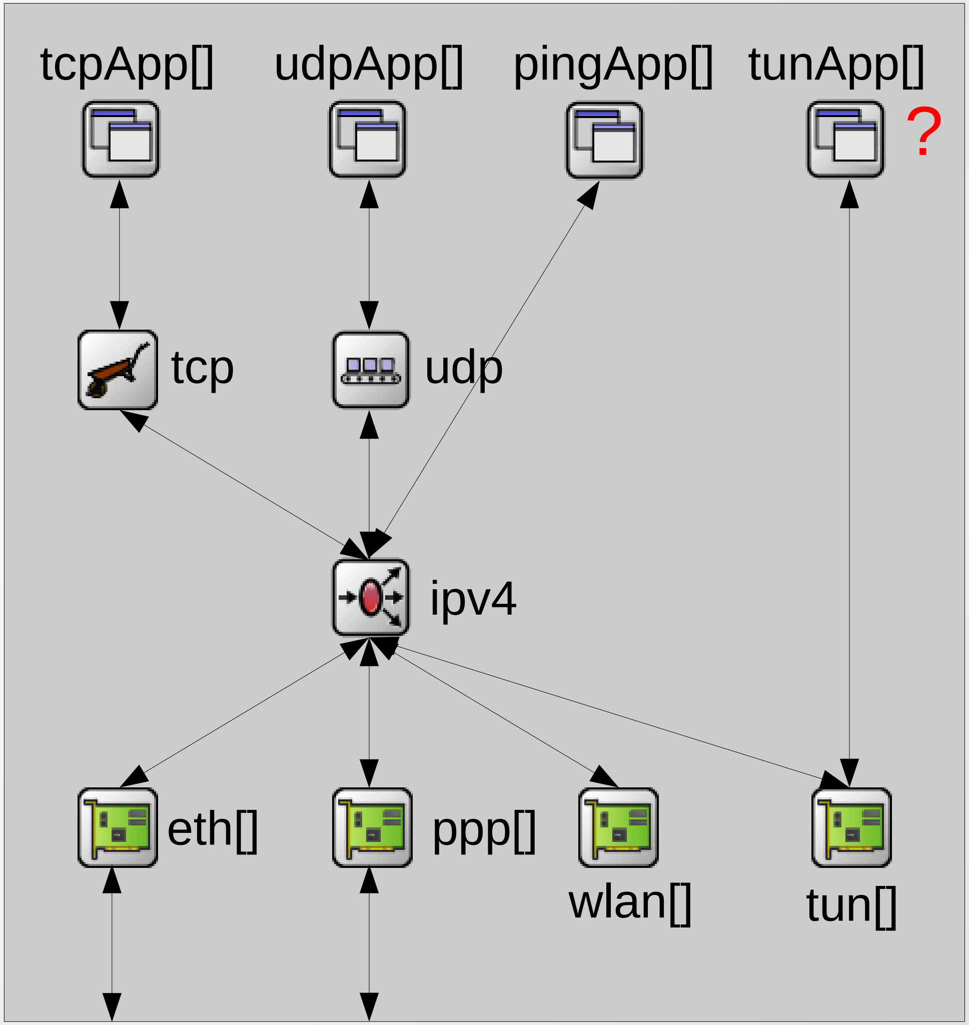
# Current Network Node Architecture 1

- Nodes currently have separate submodule vectors for different kinds of applications
- What if an application wants to use TCP and UDP simultaneously?



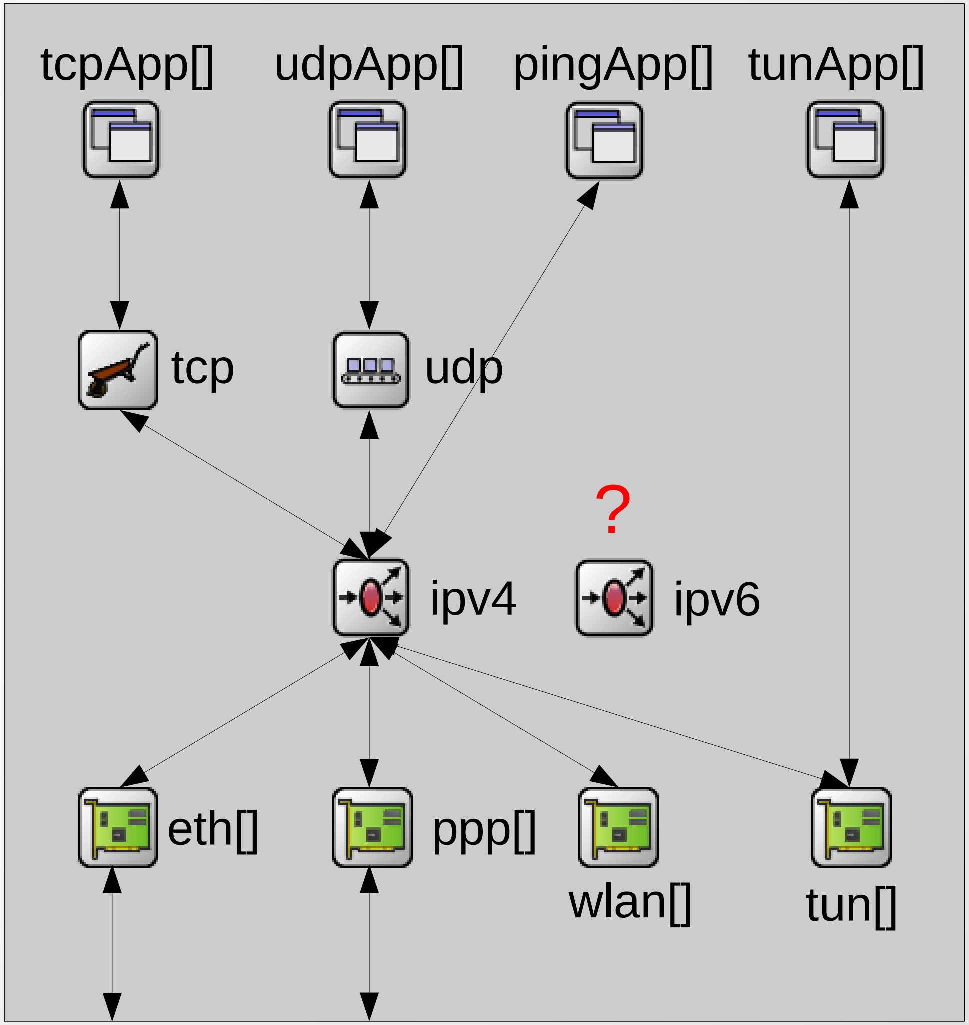
# Current Network Node Architecture 2

- TUN application are currently connected to TUN interfaces only
- TUN applications are useless without being connected to other protocols
- Should we connect them to TCP, UDP, IPv4, or all of them?



# Current Network Node Architecture 3

- Dual network stack needs extra dispatch mechanism towards network protocols
- Where should the dispatch mechanism be?
- In general, protocols of adjacent OSI layers should be in many-to-many relationship

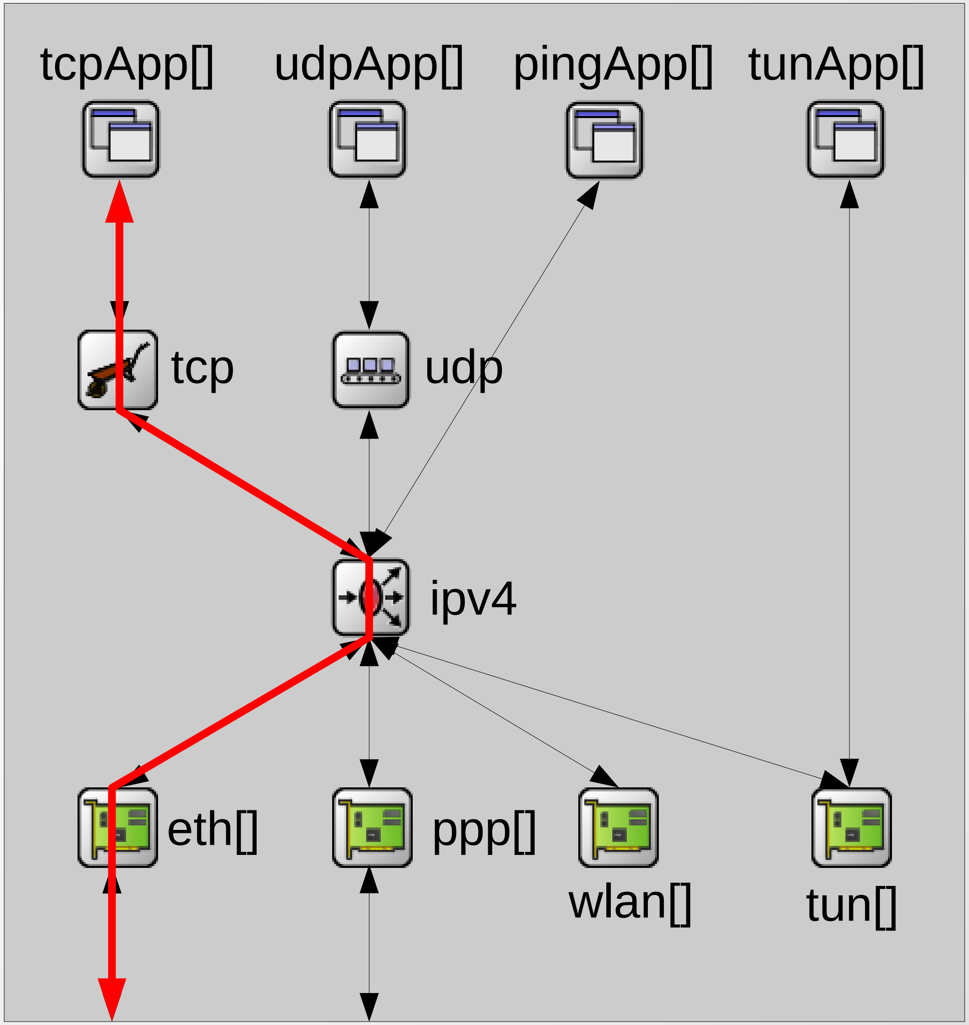


# TCP Applications

- Limitation: TCP applications exclusively and directly connect to TCP

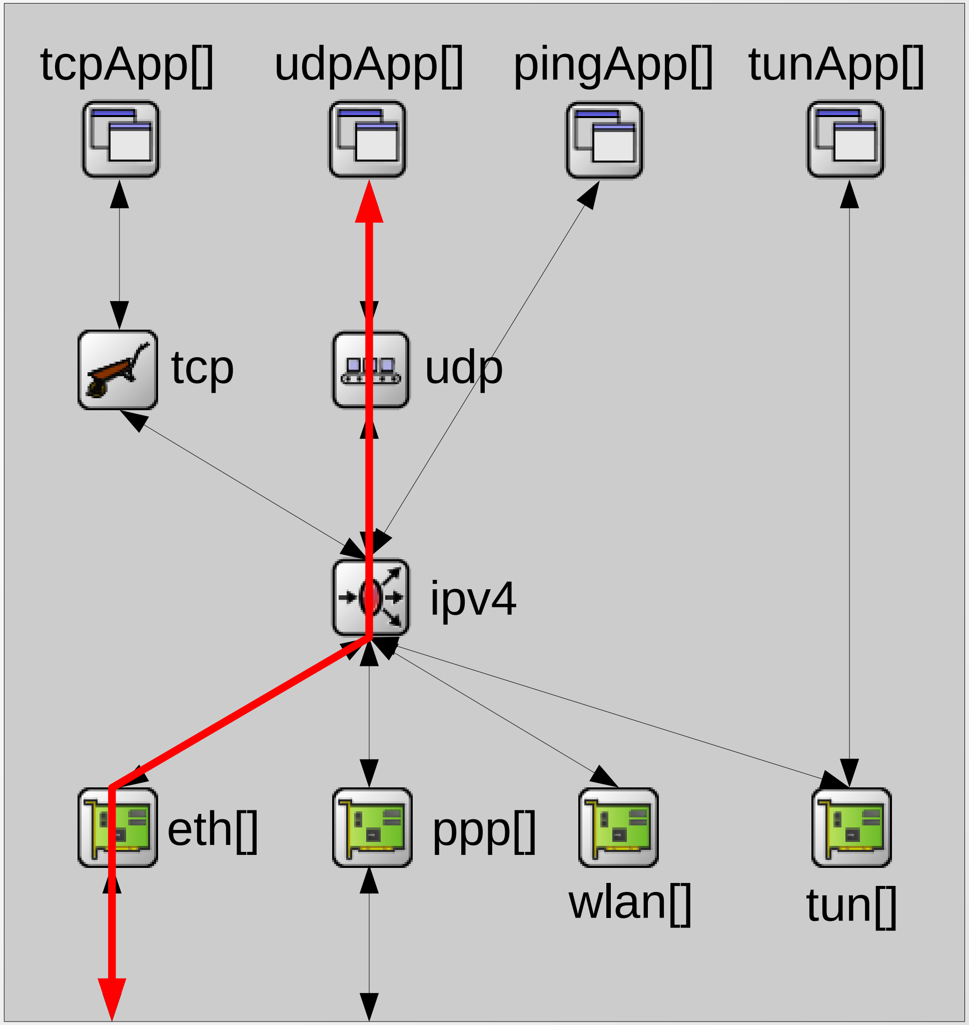
- Dispatch from TCP towards applications is built into TCP

- Dispatch from IPv4 towards TCP is built into IPv4



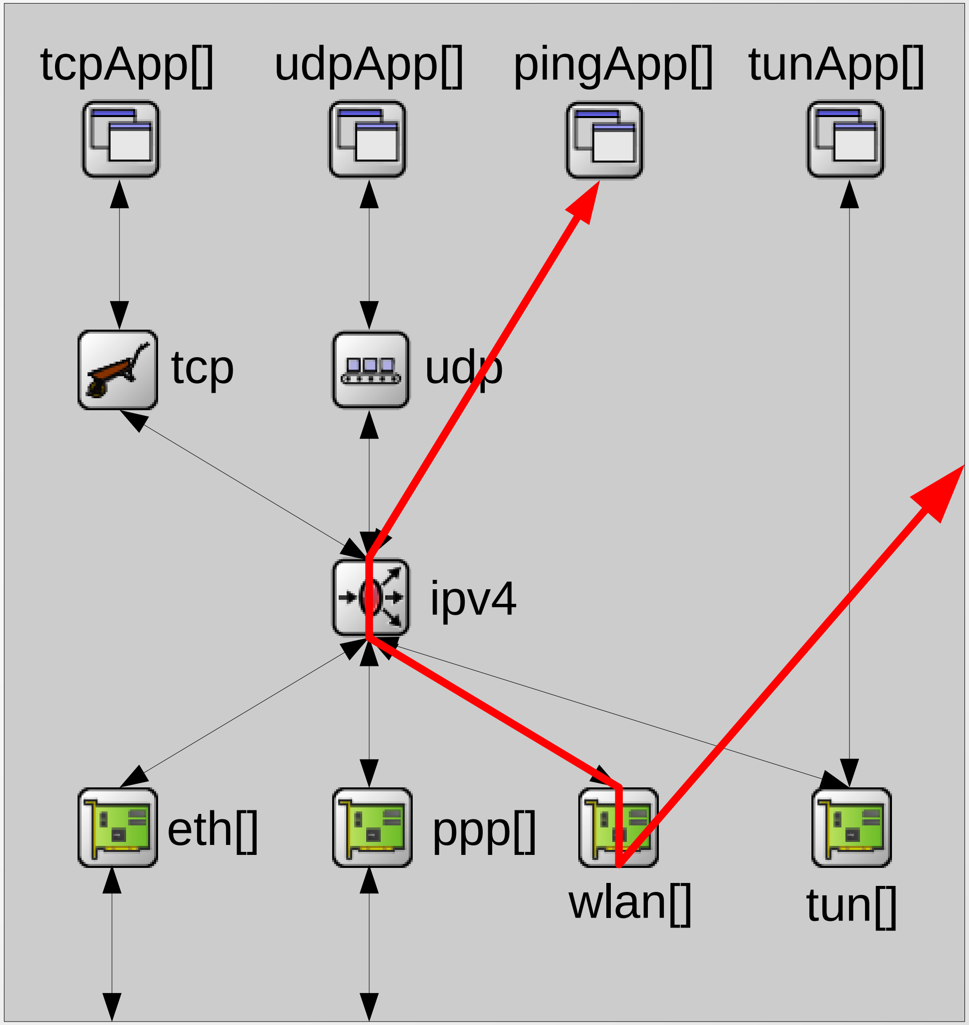
# UDP Application

- Limitations: UDP applications exclusively and directly connect to UDP
- Dispatch from UDP towards applications is built into UDP
- Dispatch from IPv4 towards UDP is built into IPv4



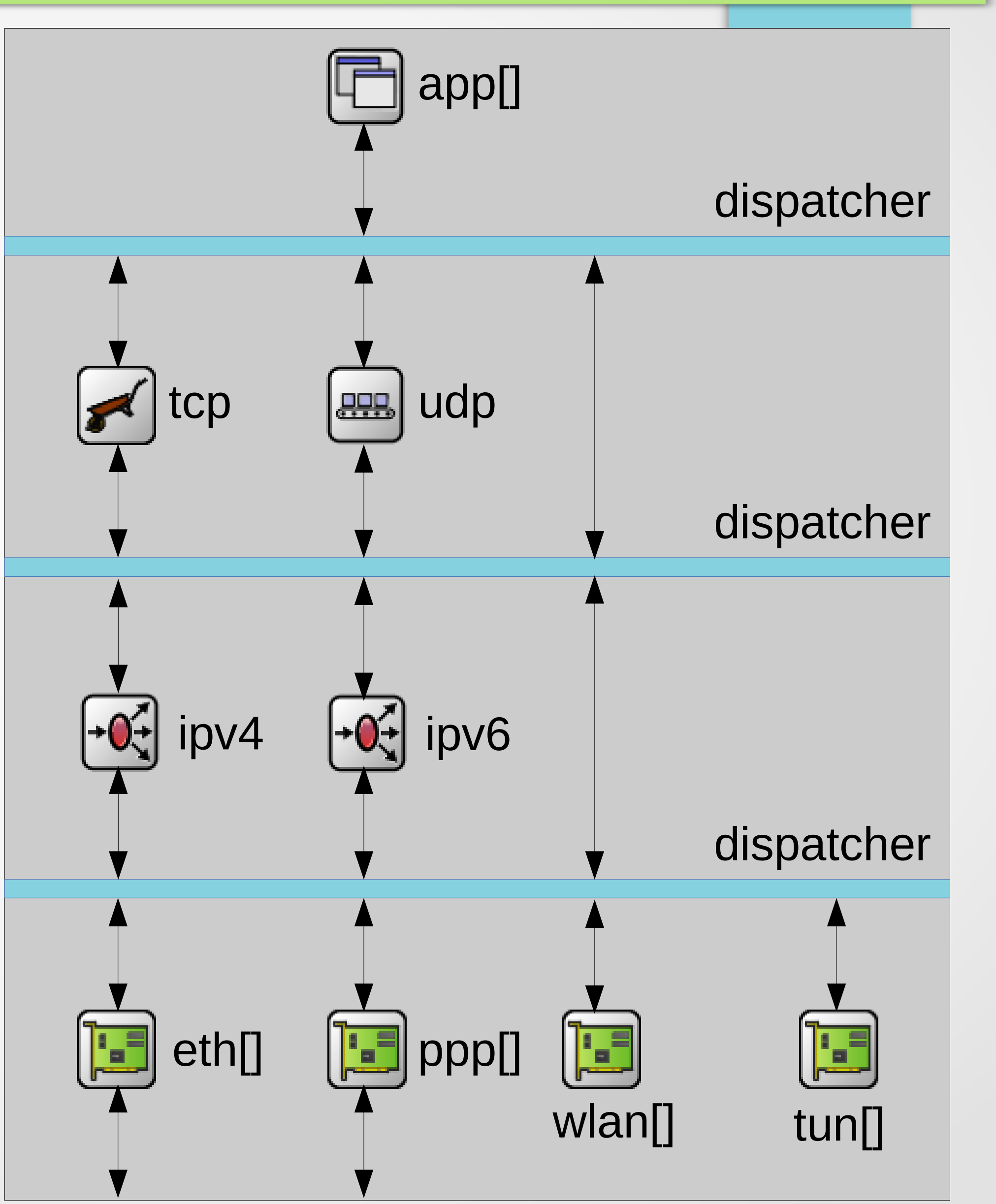
# Ping Application

- Network layers currently have separate gates for ping applications (pingIn, pingOut)
- IPv4 needs to do special ICMP packet handling to reach ping applications



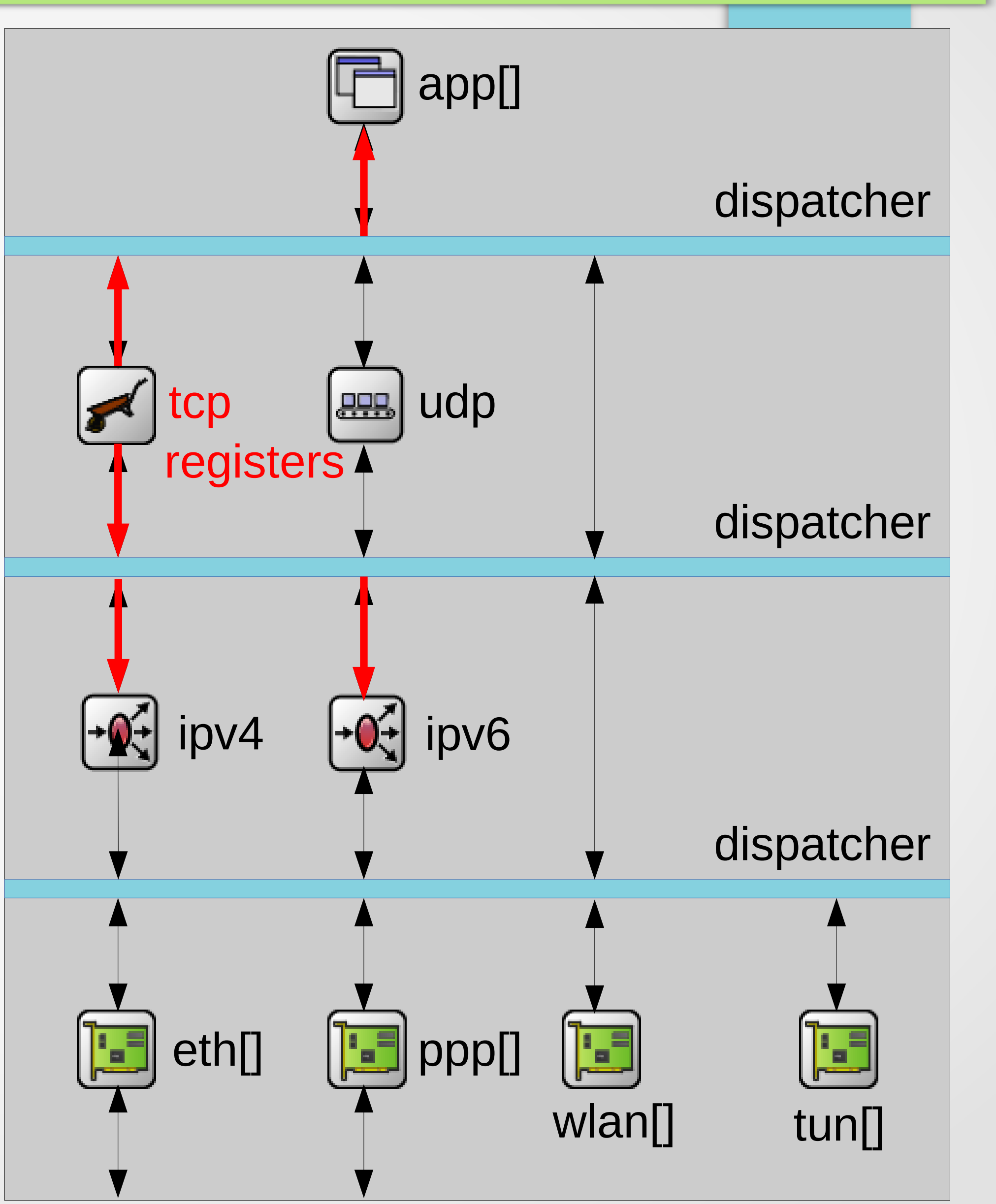
# New Network Node Architecture

- Added separate packet dispatchers between OSI layers
- Eliminated dispatch mechanisms from individual protocols



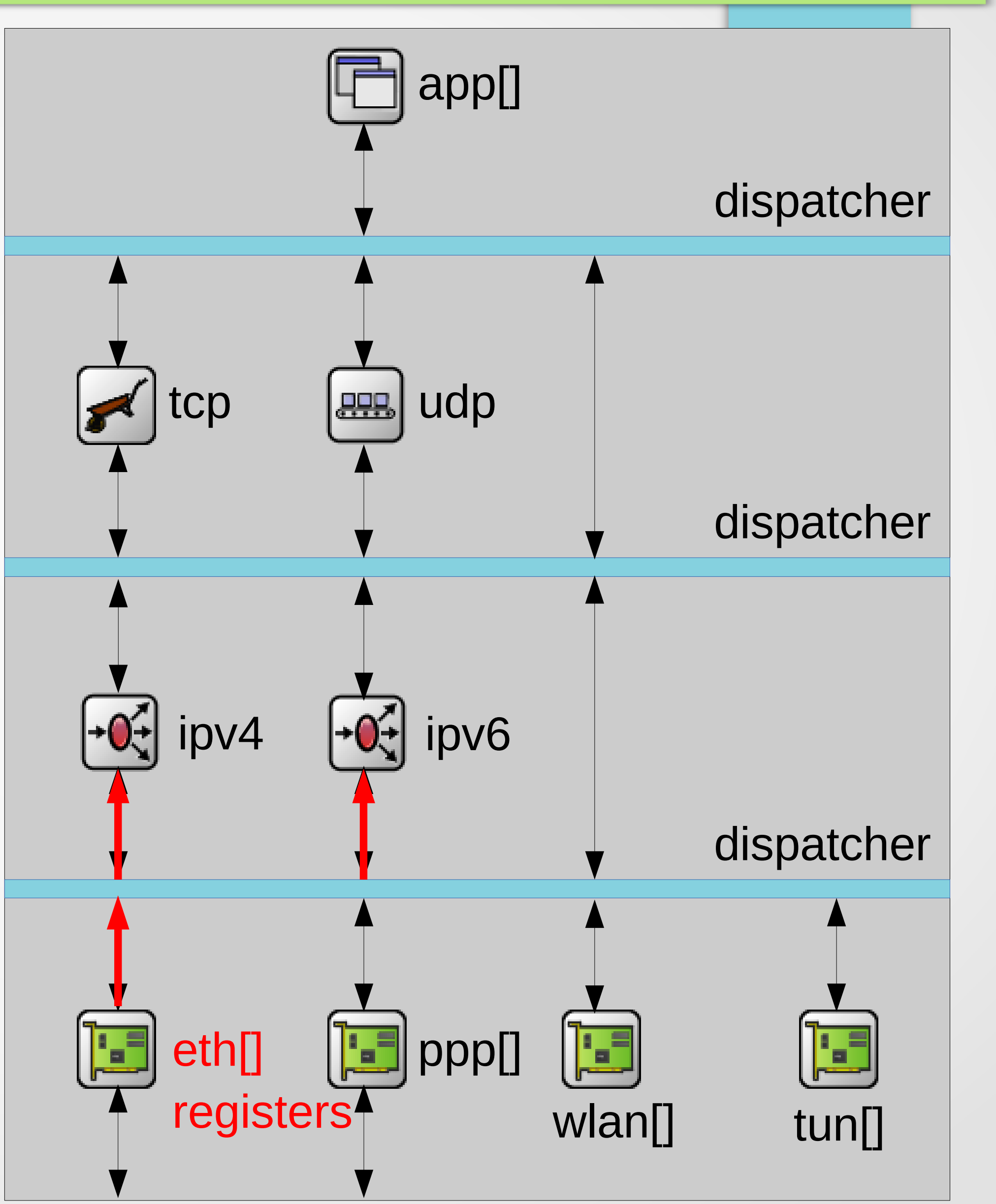
# Protocol Registration Mechanism

- Protocols have to register themselves in the dispatcher (protocolId, gate)



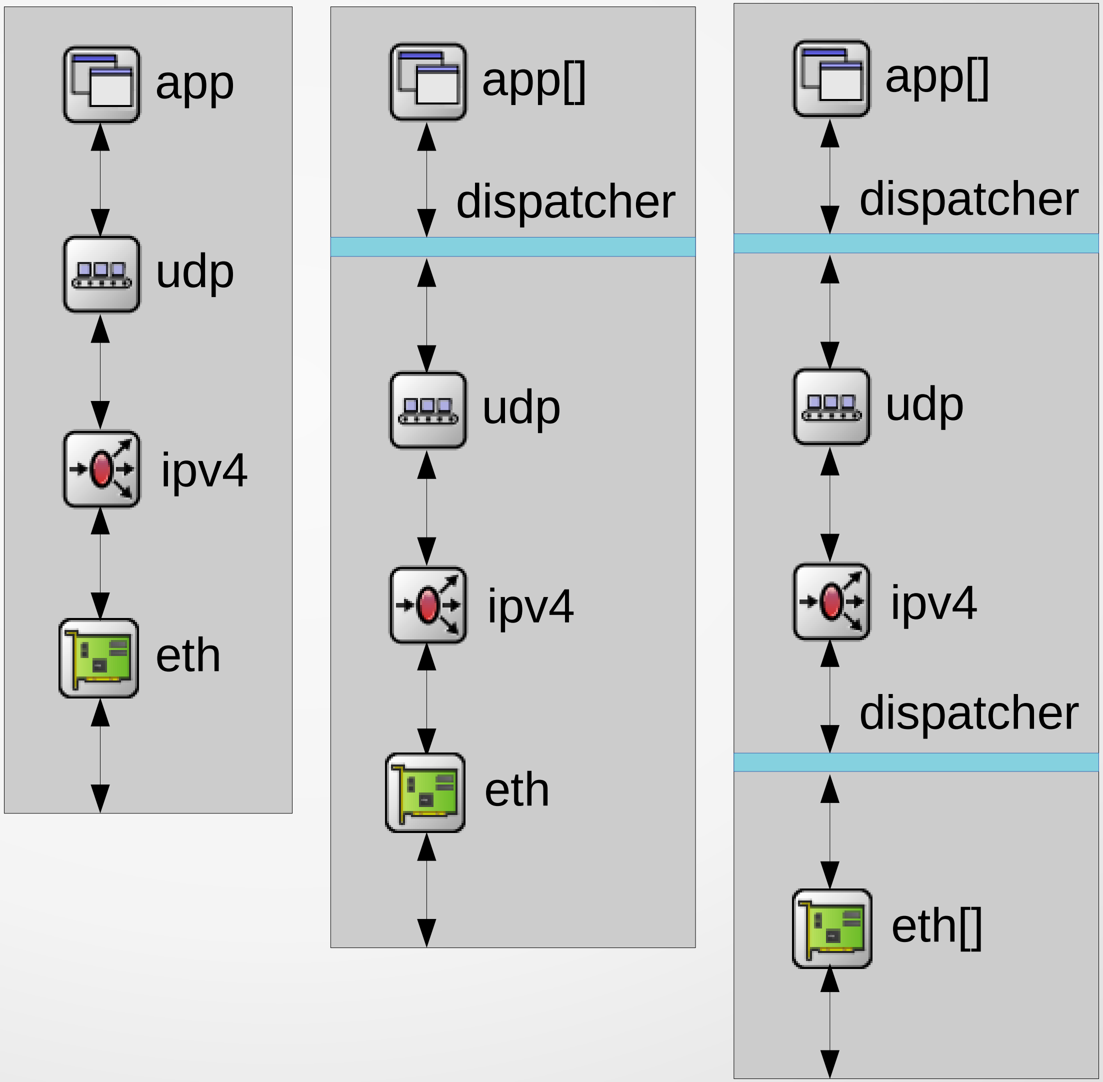
# Interface Registration Mechanism

- Interfaces also have to register themselves in the dispatcher (interfaceId, gate)



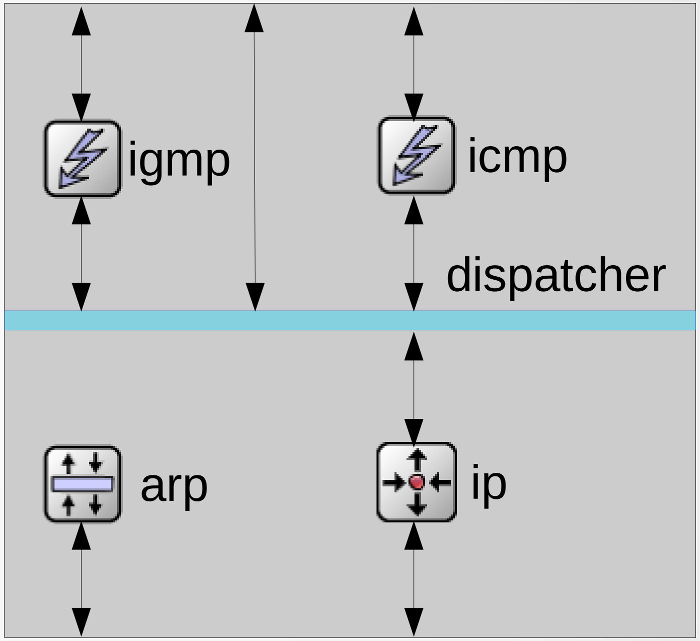
# Simpler Network Node Architectures

- Dispatchers are optional
- Modules can still be organized in other ways



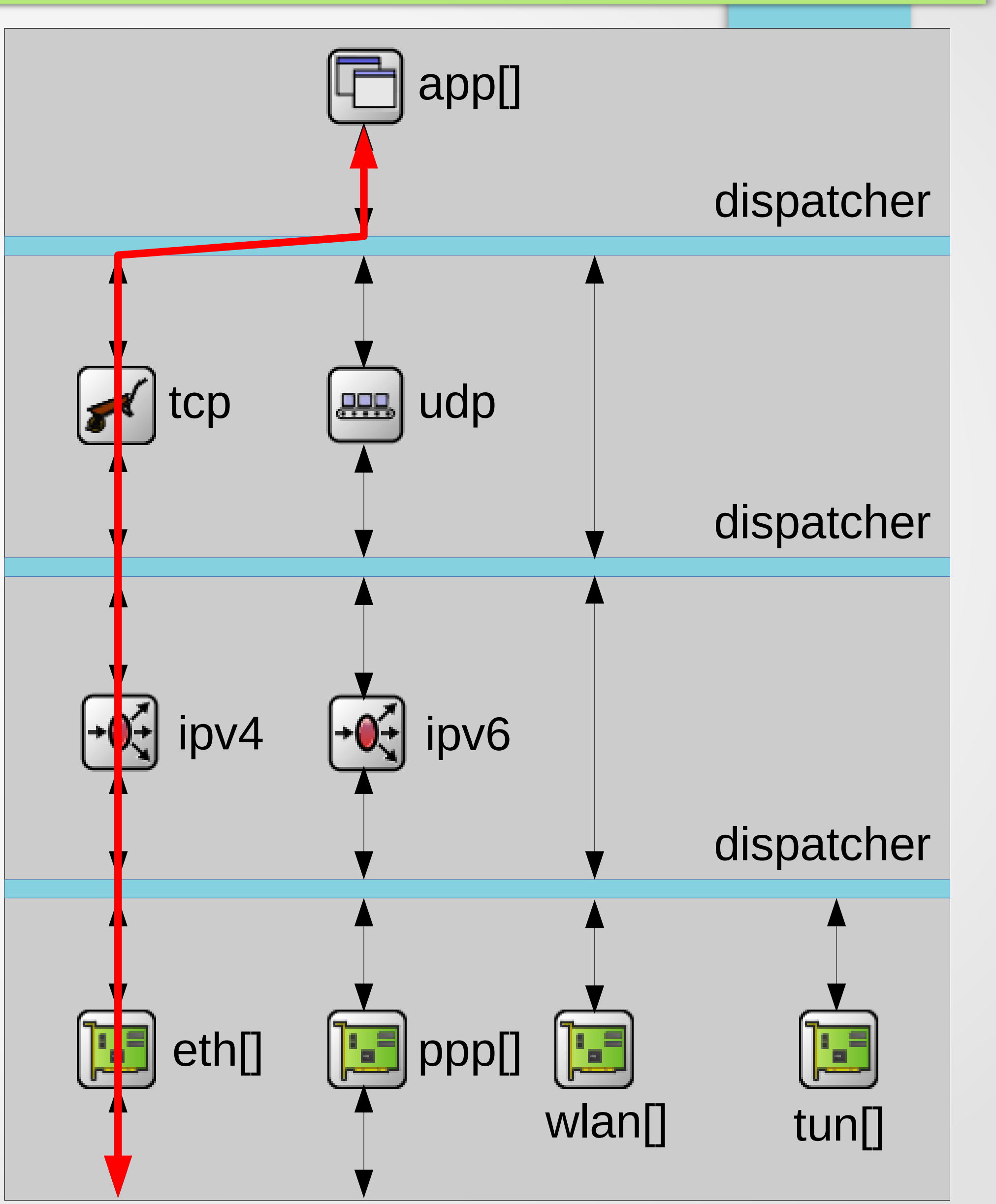
# IPv4 Network Layer Architecture

- IPv4 network layer also uses a dispatcher internally
- ARP and IP don't exchange packets, thus no connection between them



# TCP Application

- TCP application still uses a TCPSocket to send and receive packets
- Dispatcher learns where sockets are, based on socket open and close commands

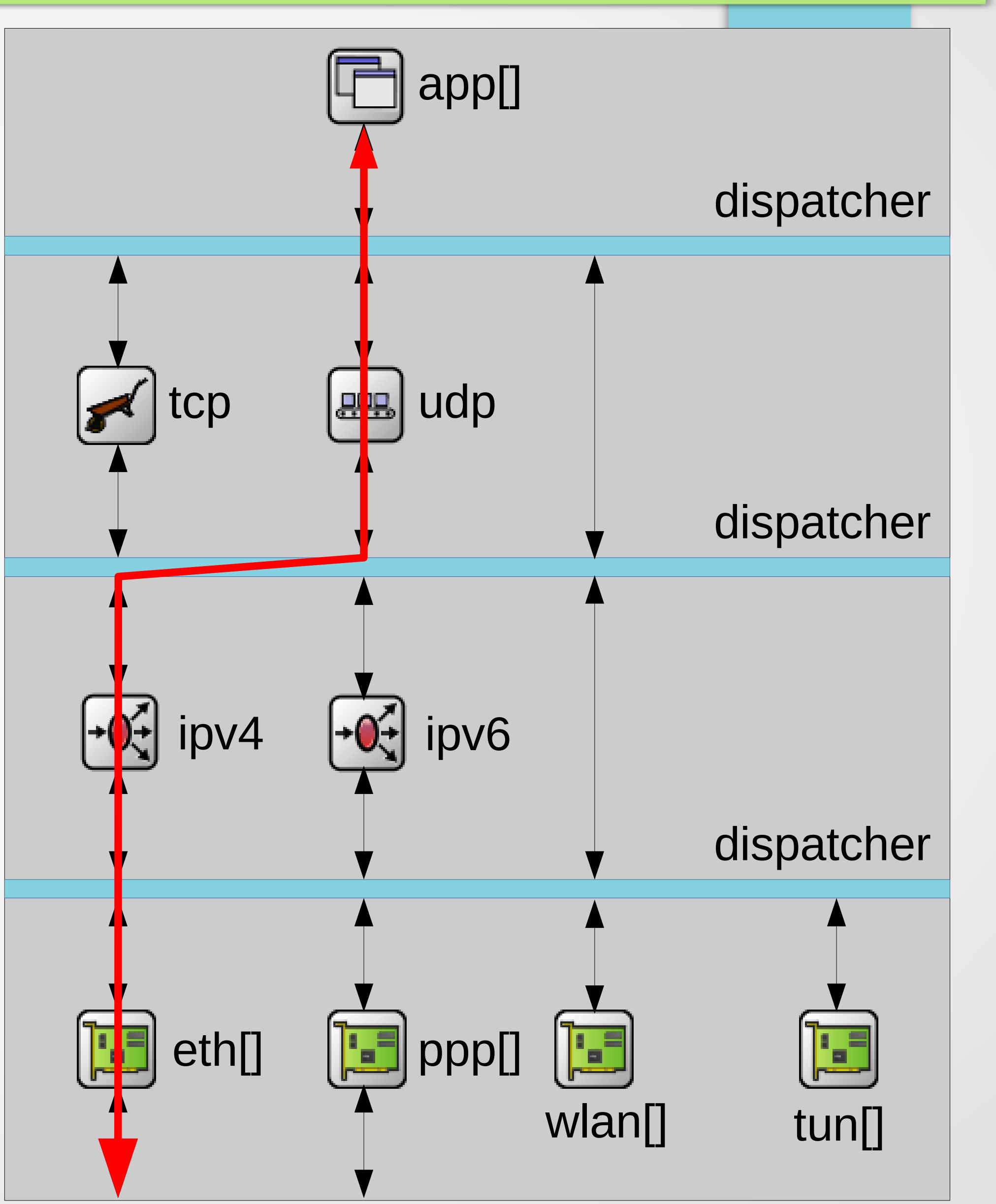


# UDP Application

- UDP application still uses a UDPSocket to send and receive packets

- Dispatcher routes packets based on the destination protocol

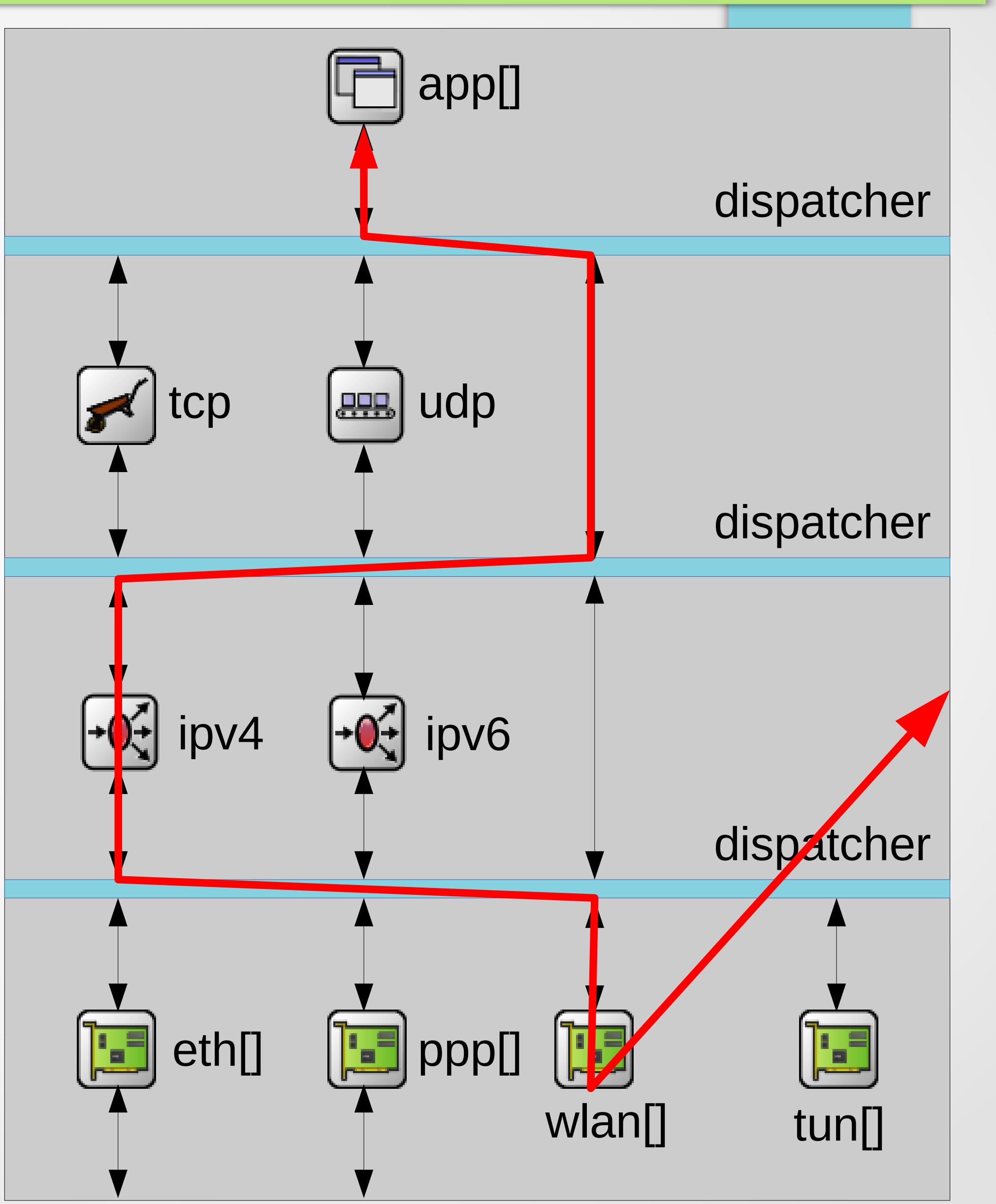
- Destination protocol is determined from control info and packet class



# Ping Application

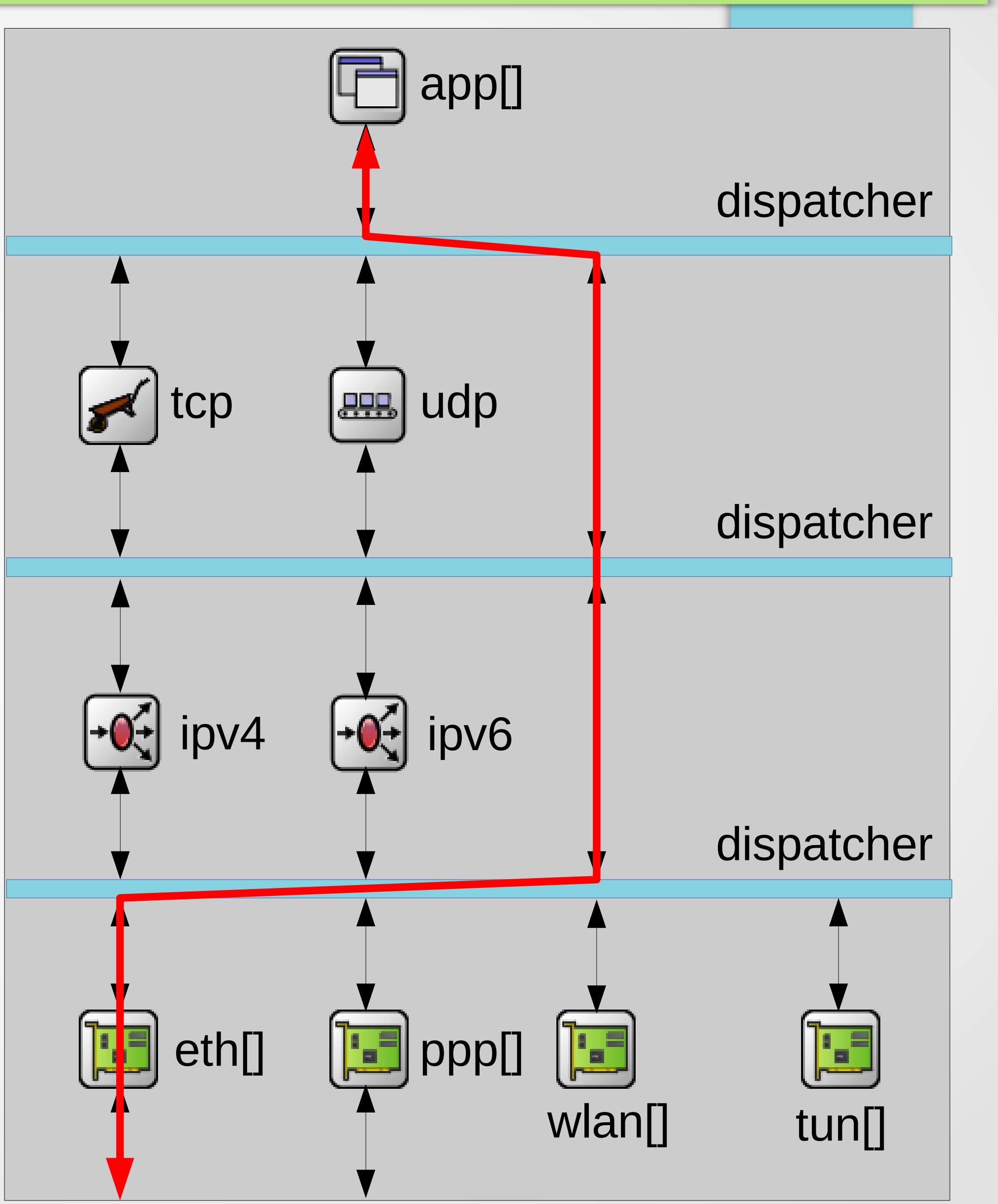
- Ping application uses a raw IPv4Socket to send and receive ICMP echo packets

- There's no special network layer gate for ping applications



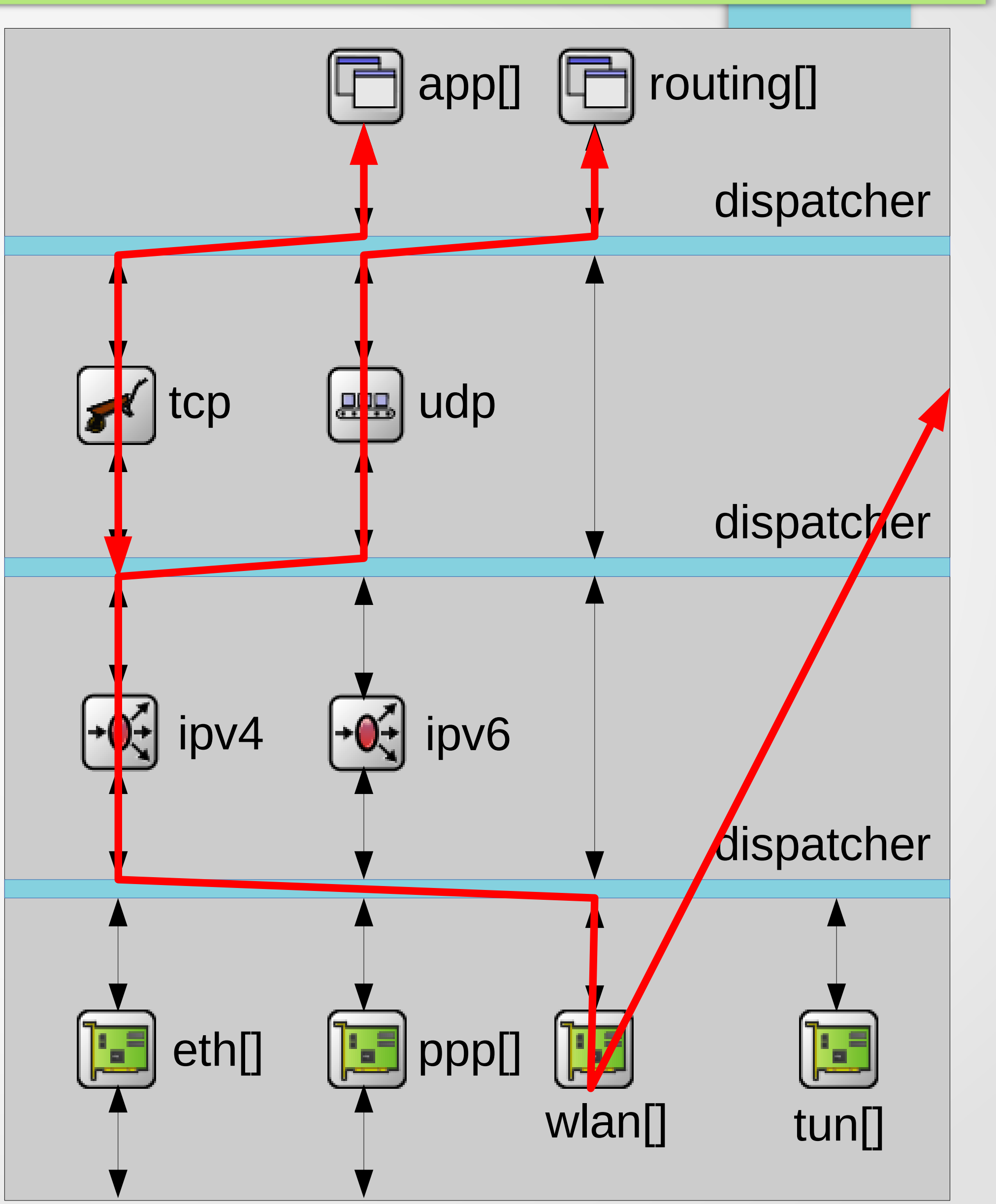
# Ethernet Application

- Ethernet applications can directly communicate using ethernet sockets



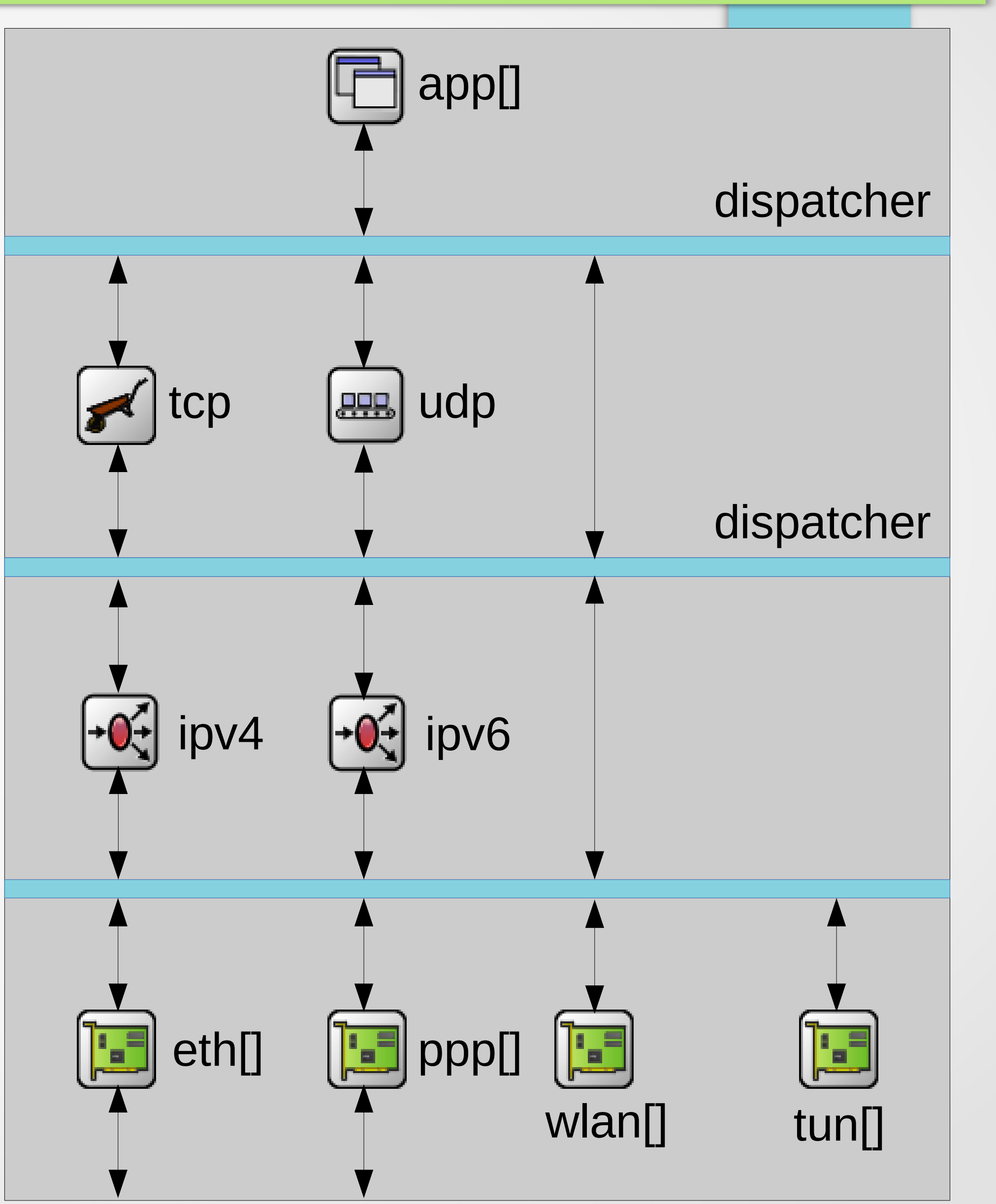
# MANET Routing Application

- MANET routers now share the same network node architecture
- Routing application can be replaced from INI file
- Routing applications can still use all kinds of sockets



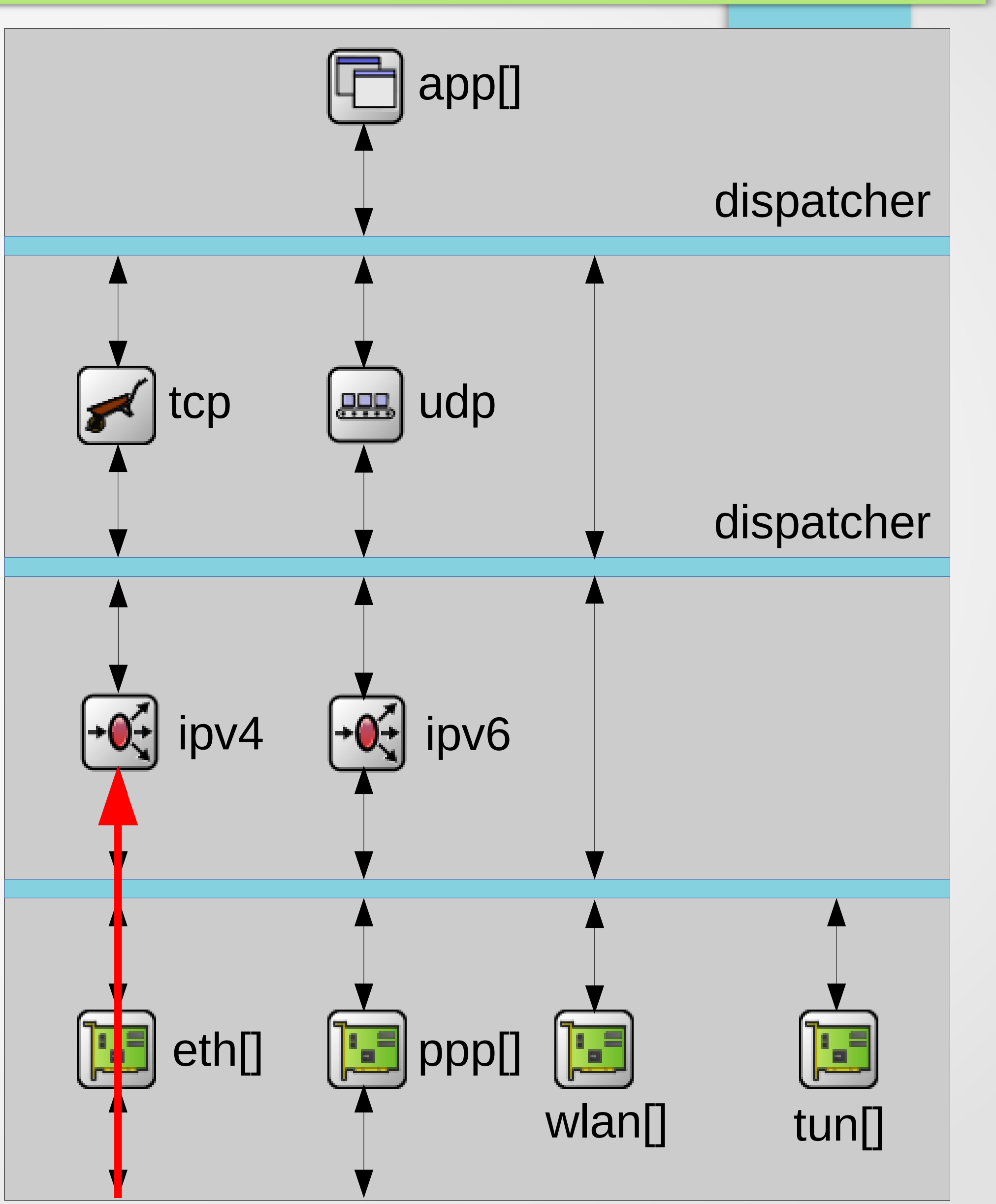
# Tunnel Application

- Tunnel application simultaneously opens a TUN device and a raw IPv4Socket



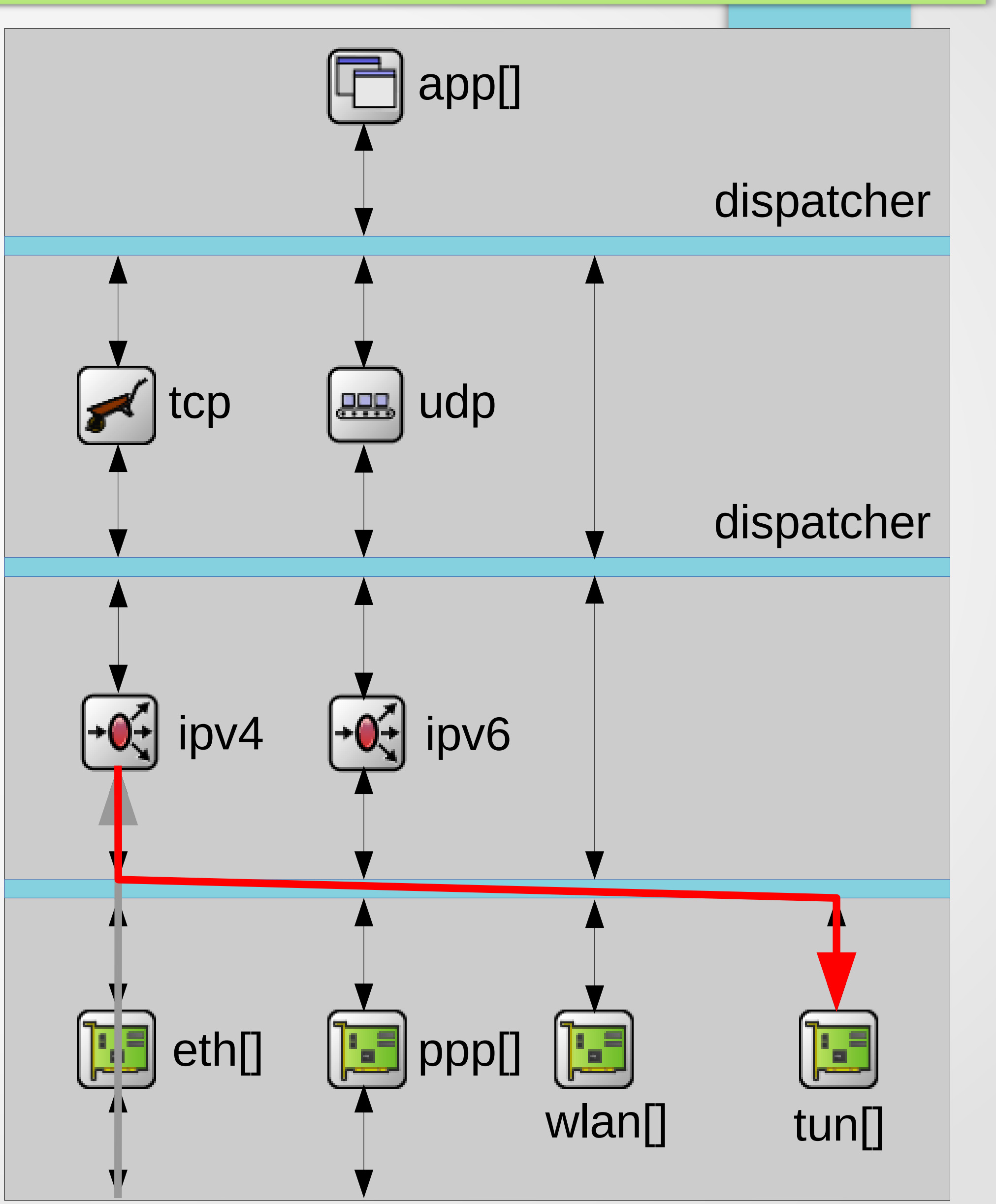
# Tunnel Application 1

- Node receives packet from network



# Tunnel Application 2

- Node receives packet from network
- IPv4 forwards packet to TUN interface

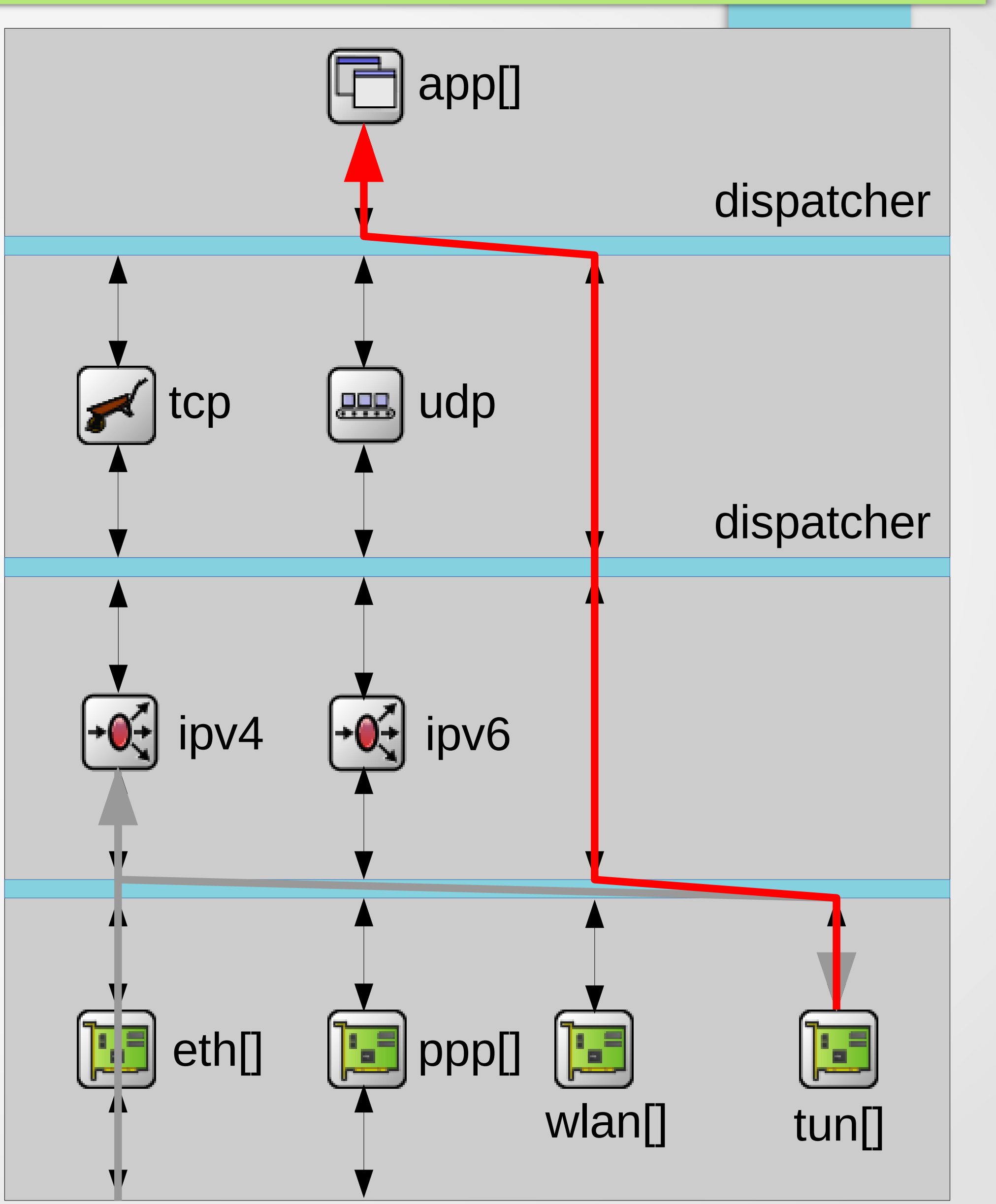


# Tunnel Application 3

- Node receives packet from network

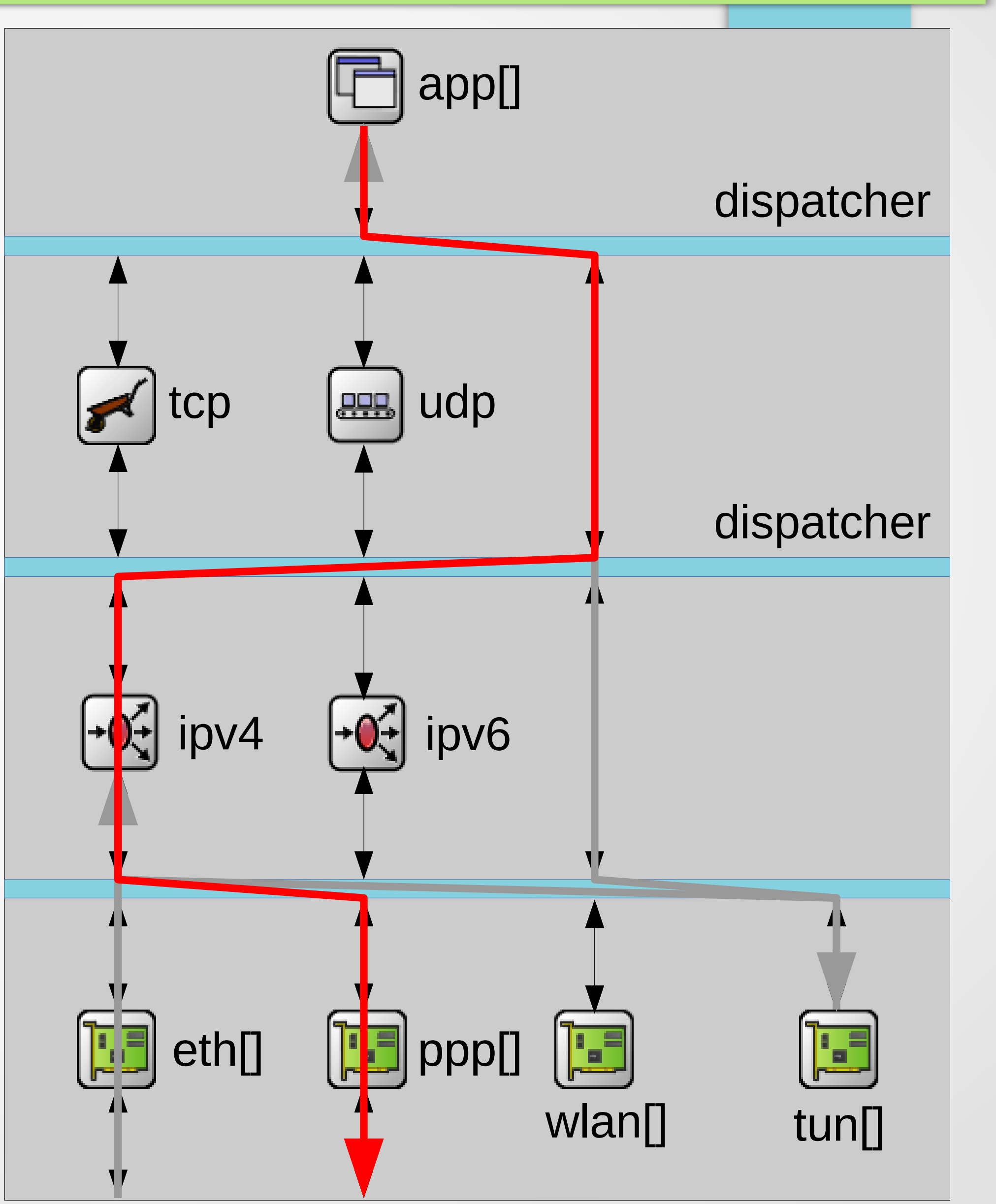
- IPv4 forwards packet to TUN interface

- Application receives packet from TUN interface

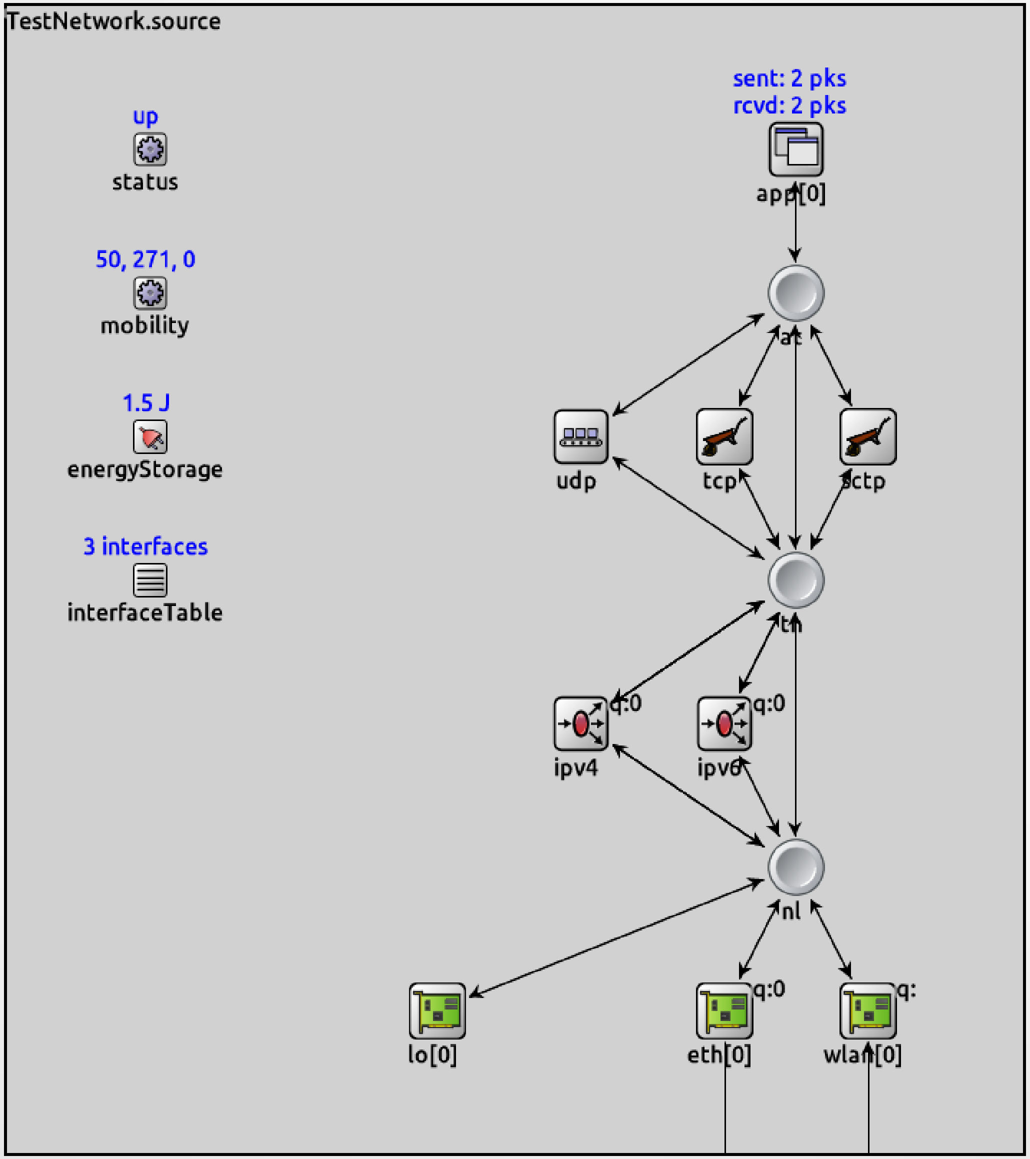


# Tunnel Application 4

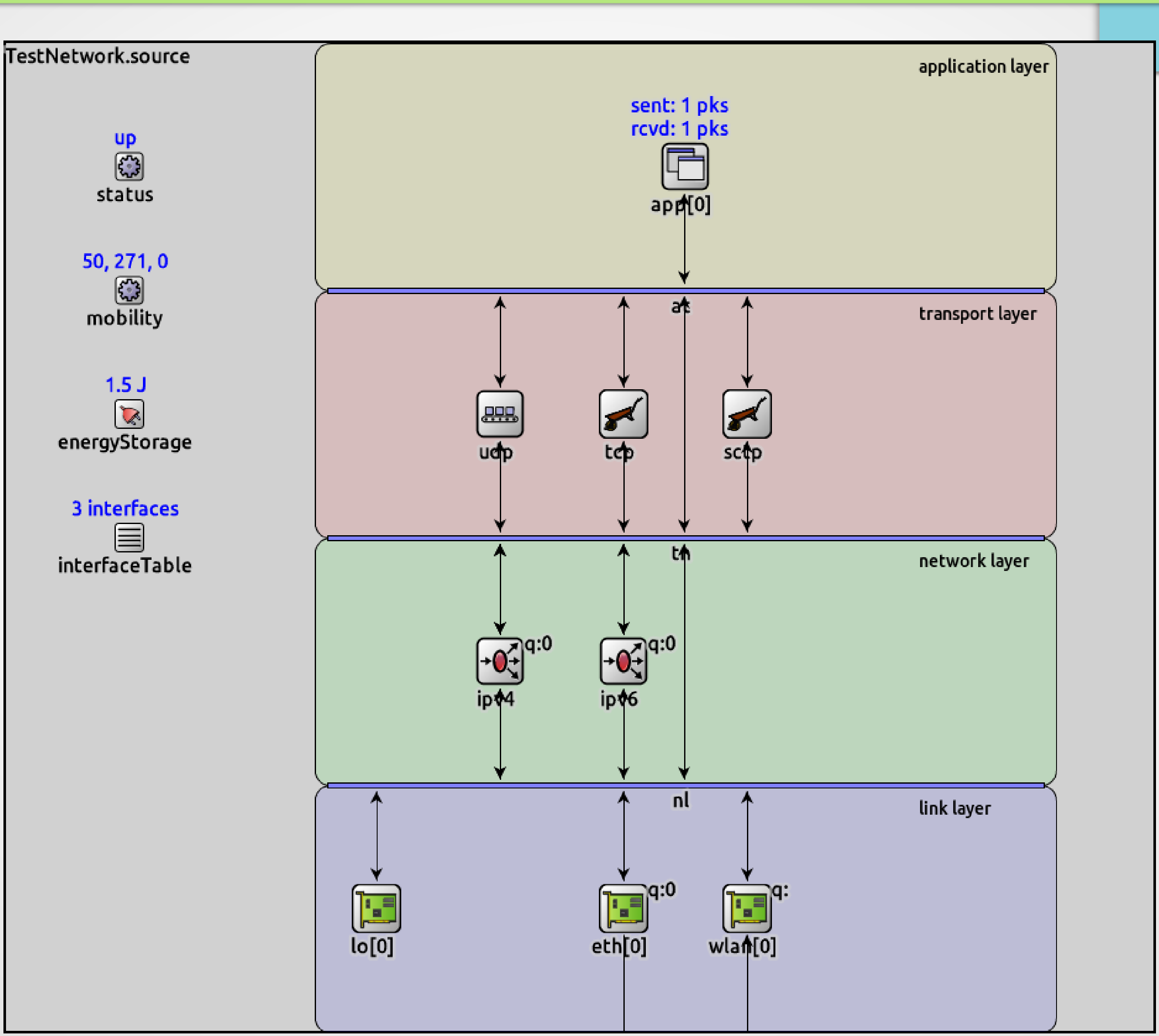
- Node receives packet from network
- IPv4 forwards packet to TUN interface
- Application receives packet from TUN interface
- Application sends packet inside another IPv4 packet



# Standard Host 1



# Standard Host 2



# Overview

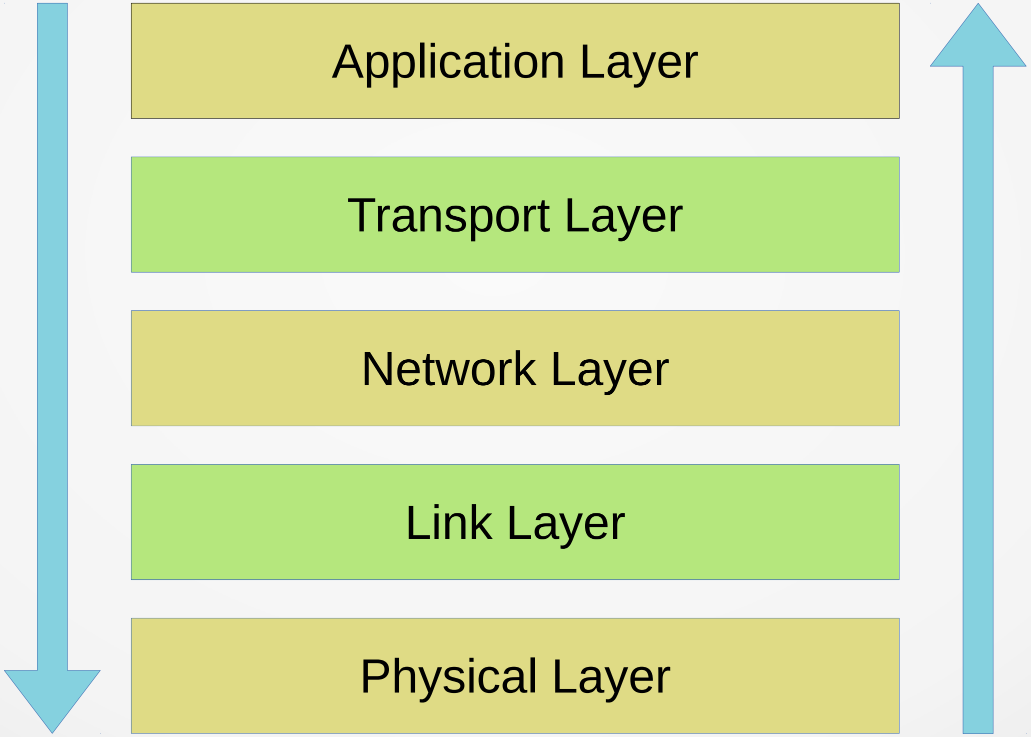
Network node architecture refactoring

Cross-layer communication and optimization

Mobility refactoring

# Cross-layer design

- Quality of service parameters
- Resource optimization parameters



Application Layer

Transport Layer

Network Layer

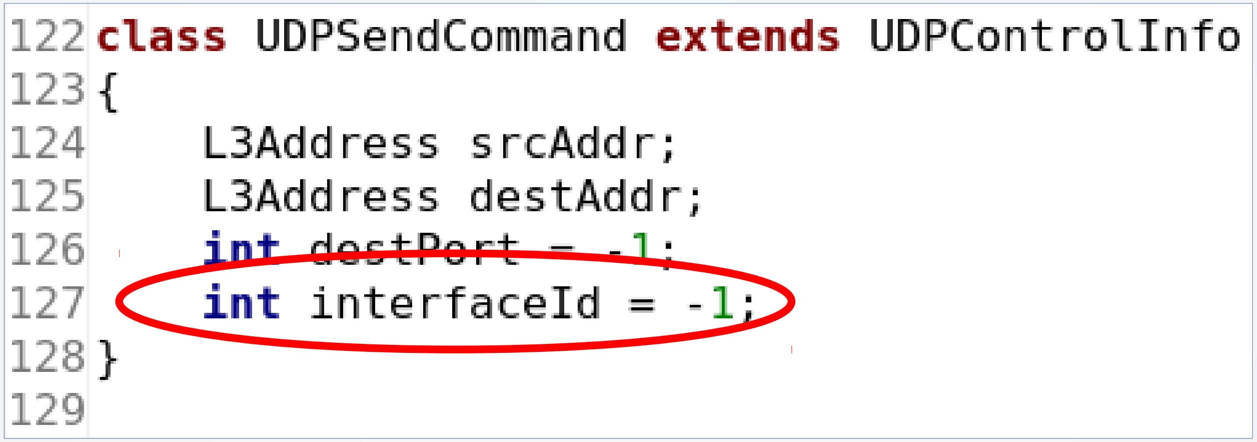
Link Layer

Physical Layer

- Link quality indication
- Physical channel conditions

# Current Cross-Layer Communication

- Applications send UDP packets on a specific interface by setting the interfaceId in UDPControlInfo



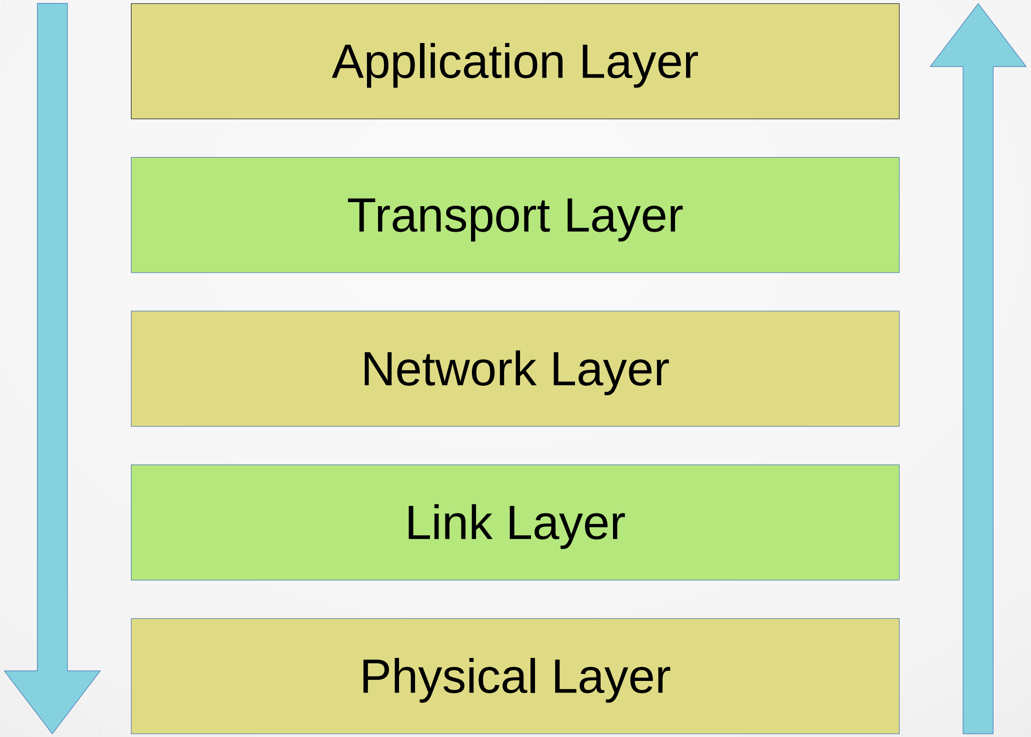
```
class UDPSendCommand extends UDPControlInfo
{
    L3Address srcAddr;
    L3Address destAddr;
    int destPort = -1;
    int interfaceId = -1;
}
```

- Currently applications cannot even specify type of service parameter to control the quality of service functionality

# New Cross-Layer Communication

- As packets go through the layers

- Packets collect various request tags



Application Layer

Transport Layer

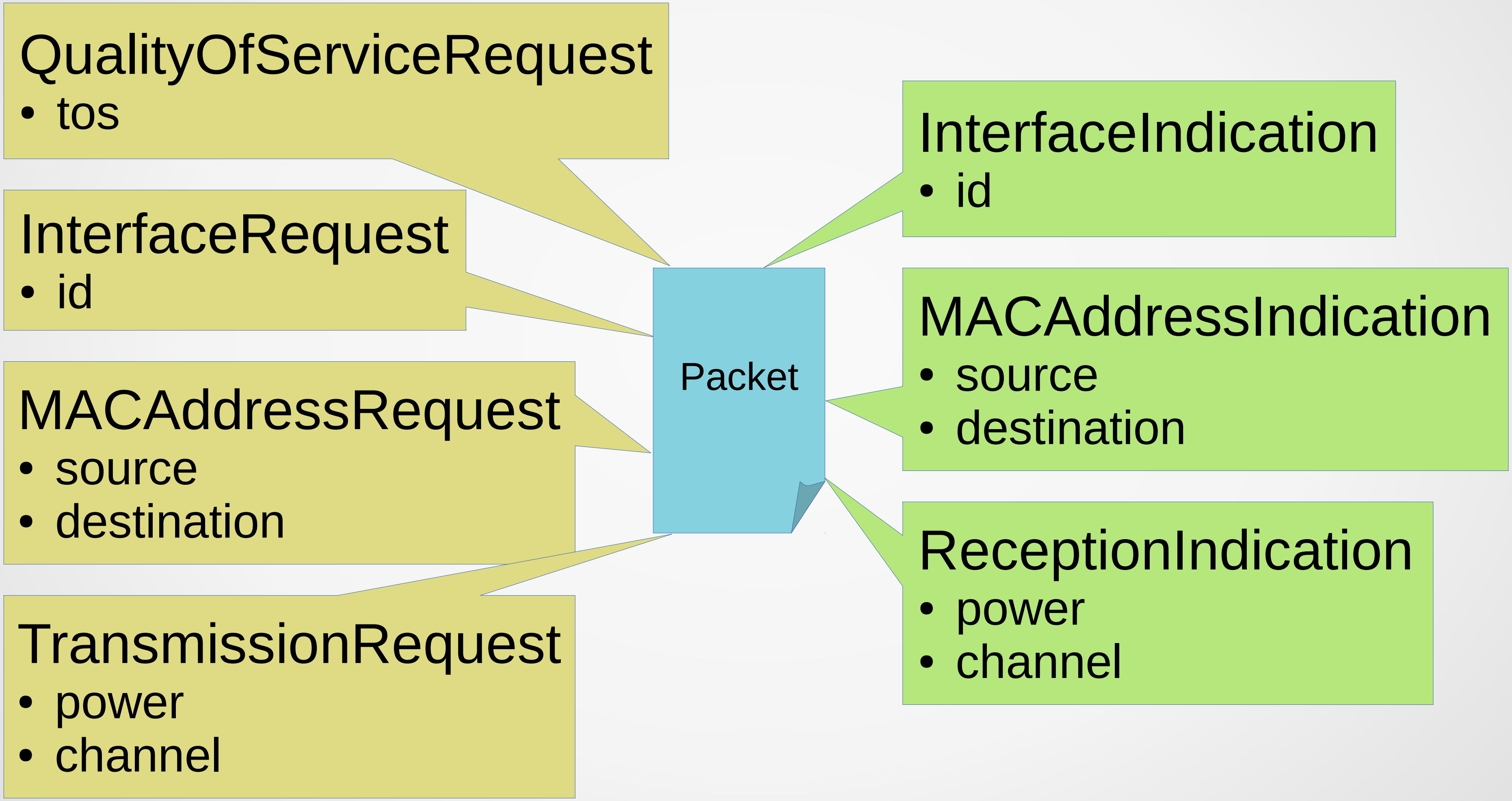
Network Layer

Link Layer

Physical Layer

- Packets collect various indication tags

# Tag Examples



QualityOfServiceRequest

- tos

InterfaceRequest

- id

MACAddressRequest

- source
- destination

TransmissionRequest

- power
- channel

Packet

InterfaceIndication

- id

MACAddressIndication

- source
- destination

ReceptionIndication

- power
- channel

# Overview

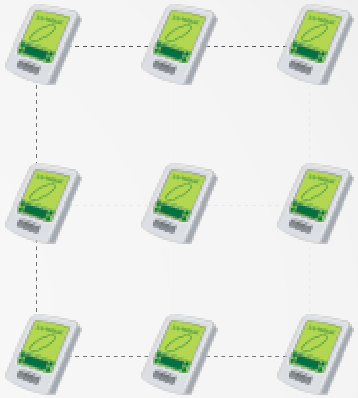
Network node architecture refactoring

Cross-layer communication and optimization

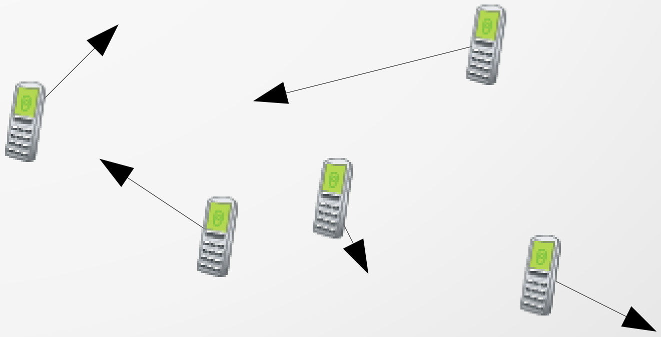
Mobility refactoring

# Current Mobility Models 1

- There are mobility models that do static positioning



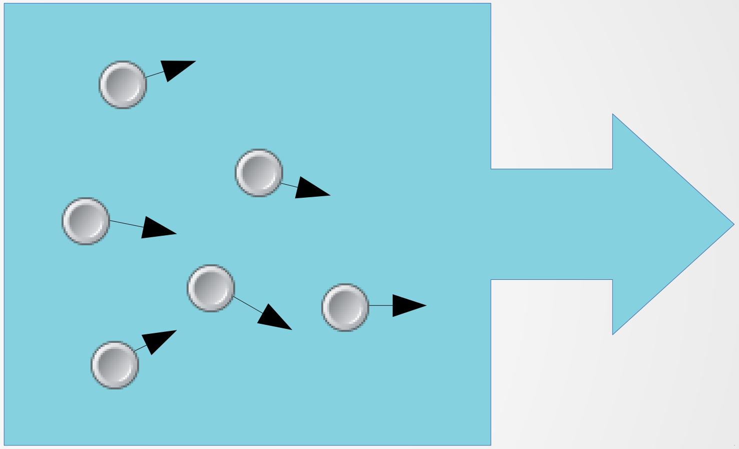
- There are mobility models that move nodes around



- How to combine these models?

# Current Mobility Models 2

- There are group mobility models with built-in differences for individual nodes



- How to reuse existing models for group mobility?
- How to model a docking ship with passengers moving around?

# Current Mobility Models 3

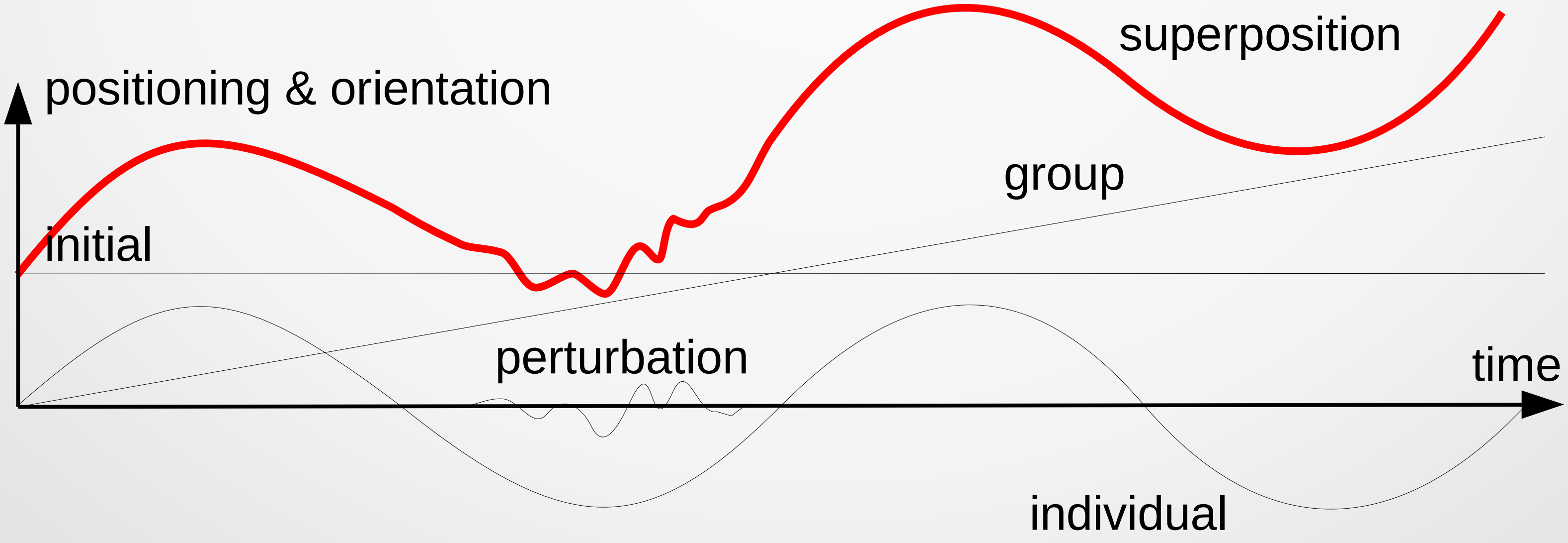
- Sometimes positioning and orientation are best expressed in separate mobility models (e.g. satellites)

- Sometimes elevation may be derived from the position on the surface of the Earth (e.g. a moving vehicle)

# Mobility Superposition

- Support combining different positioning and orientation models using superposition

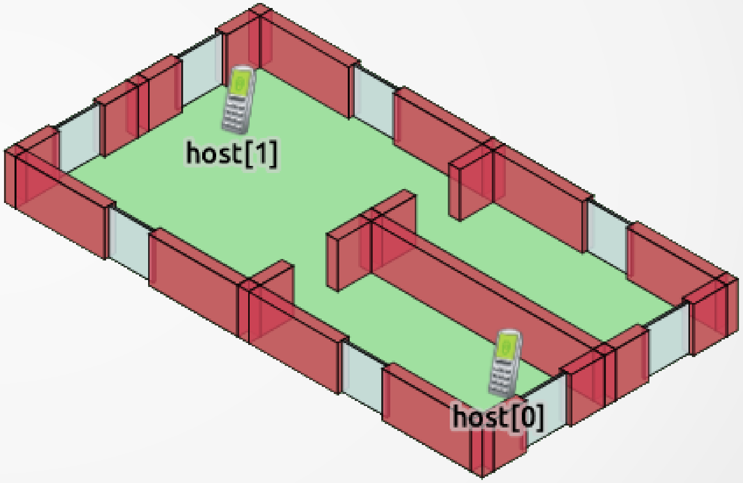
- Support start/end time (limits) for mobility models



# Coordinate Systems

- Geographic coordinate system such as WGS-84

- Abstract Cartesian coordinate system



- How to express coordinates inside buildings around a city?
- How to express antenna orientation of a vehicle separately?

# Questions and Answers

Thank you for your kind attention!