# Network Layer Protocols of IoT

Compiled by :-

Asst. Prof. Rashmi M. Pote.

## Network Layer Routing Protocols

- In this section, we discuss some standard and non-standard protocols that are used for routing in IoT applications. It should be noted that we have partitioned the network layer in two sublayers:
- routing layer which handles the transfer the packets from source to destination, and an
- encapsulation layer that forms the packets. Encapsulation mechanisms will be discussed in the next section.

# INTERNET PROTOCOL VERSION 4(IPV4)

## BASICS OF IPV4

- IPv4 is the first network protocol to interconnect different networks regardless of the medium used.
- Globally unique addressing scheme
- Any two nodes can communicate directly

## IPV4 ADDRESSING

- Every node is identified by a four byte address
- Networks are divided by subnet classes each class has a fixed number of network bits
- Communication between nodes on different networks is established by routers

## INTERNET PROTOCOL VERSION 6(IPV6)

## BASICS

- General perception is that “IPv6 has not yet taken hold” IPv4 Address run-out is not “headline news” yet

- More discussions and run-out plans proposed Private sector requires a business case to “migrate”

- No easy Return on Investment (RoI) computation. But reality is very different from perception! Something needs to be done to sustain the Internet growth

- IPv6 or NAT or both or something else?

## NETWORK ADDRESS TRANSLATION

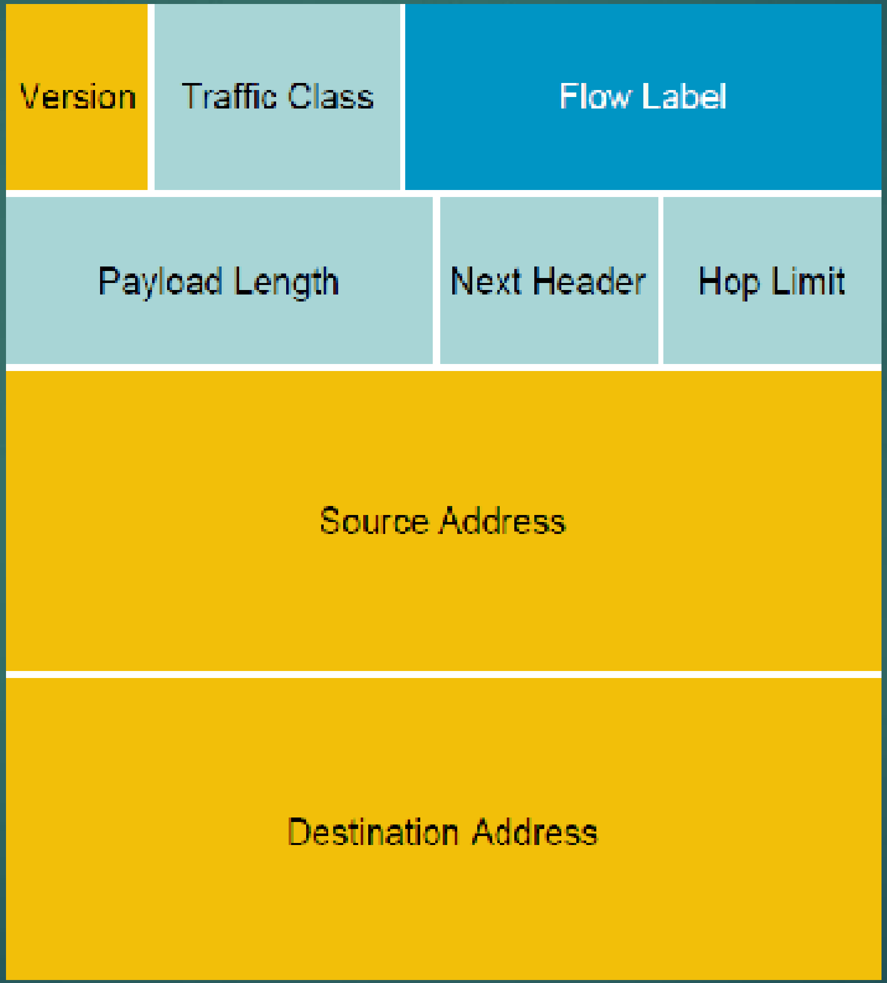
- Private address space and Network address translation (NAT) could be used instead of IPv6

- But NAT has many serious issues:
  - Breaks the end-to-end model of IP
  - Layered NAT devices
  - Mandates that the network keeps the state of the connections
  - How to scale NAT performance for large networks?
  - Makes fast rerouting difficult
  - Service provision inhibited

## PROTOCOLS AND STANDARDS

- Expanded address space:
  - Address length quadrupled to 16 bytes
- Header Format Simplification:
  - Fixed length, optional headers are daisy-chained
  - IPv6 header is twice as long (40 bytes) as IPv4 header without options (20 bytes)
- No checksum at the IP network layer
- No hop-by-hop segmentation
  - Path MTU discovery
- 64 bits aligned
- Authentication and Privacy Capabilities
  - IPsec is mandated
- No more broadcast

## IPV6 HEADER


Version
Traffic Class
Flow Label
Payload Length
Next Header
Hop Limit
Source Address
Destination Address

## LARGE ADDRESS SPACE

- IPv4:
  - 32 bits
  - = 4,294,967,296 possible addressable devices

- IPv6:
  - 128 bits: 4 times the size in bits
  - = 3.4 x 1038 possible adressable devises
  - =340,282,366,920,938,463,463,374,607,431,768,211,456
  - ~5 x 1028 addresses per person on the planet

## MULTICAST ADDRESSES

- Broadcasts in IPv4:
  - Interrupts all devices on the LAN even if the intent of the request was for a subset
  - Can completely swamp the network (“broadcast storm”)

- Broadcasts in IPv6:
  - Are not used and replaced by multicast

- Multicast:
  - Enables the efficient use of the network
  - Multicast address range is much larger

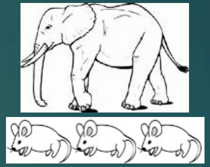
# IoT Ecosystem

| | |
|---|---|
| **Applications** | Smart Health, Smart Home, Smart Grid Smart Transport, Smart Workspaces, … |
| **Session** | MQTT, CoRE, DDS, AMQP , … |
| **Routing** | 6LowPAN, RPL, 6Lo, 6tsch, Thread, 6-to-nonIP , … |
| **Datalink** | WiFi, Bluetooth Smart, Zigbee Smart, Z-Wave, DECT/ULE, 3G/LTE, NFC, Weightless, **HomePlug GP**, 802.11ah, **802.15.4**, G.9959, WirelessHART, DASH7, ANT+ , LoRaWAN, … |
| **Software** | Mbed, Homekit, AllSeen, IoTvity, ThingWorks, EVRYTHNG , … |
| **Operating Systems** | Linux, Android, Contiki-OS, TinyOS, … |
| **Hardware** | ARM, Arduino, Raspberry Pi, ARC-EM4, Mote, Smart Dust, Tmote Sky, … |

| **Security** | **Management** |
|---|---|
| TCG, Oath 2.0, SMACK, SASL, ISASecure, ace, CoAP, DTLS, Dice | IEEE 1905, IEEE 1451, … |

# 6LowPAN

- IPv6 over Low Power Wireless Personal Area Networks
- How to transmit IPv6 datagrams (elephants) over low power IoT devices (mice)?



- **Issues**:

1. **IPv6 address formation**: 128-bit IPv6 from 64-bit EUI64
2. **Maximum Transmission Unit** (MTU): IPv6 at least 1280 bytes vs. IEEE 802.15.4 standard packet size is 127 bytes

| 802.15.4 Header | Security Option | Payload |
|---|---|---|
| 25B | 21B | 81B |

3. **Address Resolution**: 128b or 16B IPv6 addresses. 802.15.4 devices use 64 bit (no network prefix) or 16 bit addresses
4. **Optional mesh routing in datalink layer**
   $\Rightarrow$ Need destination and intermediate addresses.

Ref: G. Montenegro, et al., "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," RFC 4944, Sep 2007, http://tools.ietf.org/pdf/rfc4944

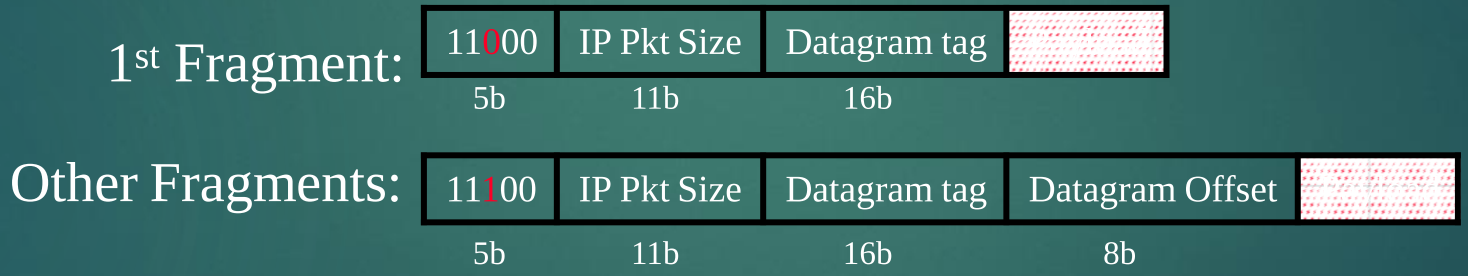
# 6LowPAN Broadcast Header

- For Mesh broadcast/multicast
- A new sequence number is put in every broadcast message by the originator

| Dispatch $01010000_2$ | Sequence Number |
|---|---|
| 8b | 8b |

## 6LowPAN Fragment Header

- Dispatch = 11*x*00 (5 bits) ⇒ Fragment Header
- Full packet size in the first fragment's fragment header
- Datagram tag = sequence number
  ⇒ Fragments of the same packet
- Fragment Offset in multiples of 8 bytes



## 6LowPAN: Summary

- **3 New Headers**:
  - Mesh addressing: Intermediate addresses
  - Hop-by-Hop: Mesh broadcasts
  - Destination processing: Fragmentation
- **Address Formation**: 128-bit addresses by prefixing FE80::
- **Header compression**:
  - HC1+HC2 header for link-local IPv6 addresses
  - IPHC compression for all IPv6 addresses

## 6LoWPAN

- IPv6 over Low power Wireless Personal Area Network (6LoWPAN) is the first and most commonly used standard in this category. It efficiently encapsulates IPv6 long headers in IEEE802.15.4 small packets, which cannot exceed 128 bytes. The specification supports different length addresses, low bandwidth, different topologies including star or mesh, power consumption, low cost, scalable networks, mobility, unreliability and long sleep time.

## Lossy Network

- Low power and **Lossy Networks** (LLNs ) are made up of many embedded devices with limited power, memory, and processing resources. They are. interconnected by a variety of links, such as IEEE 802.15.4, Bluetooth, Low Power WiFi, wired or other low power PLC (Powerline Communication)

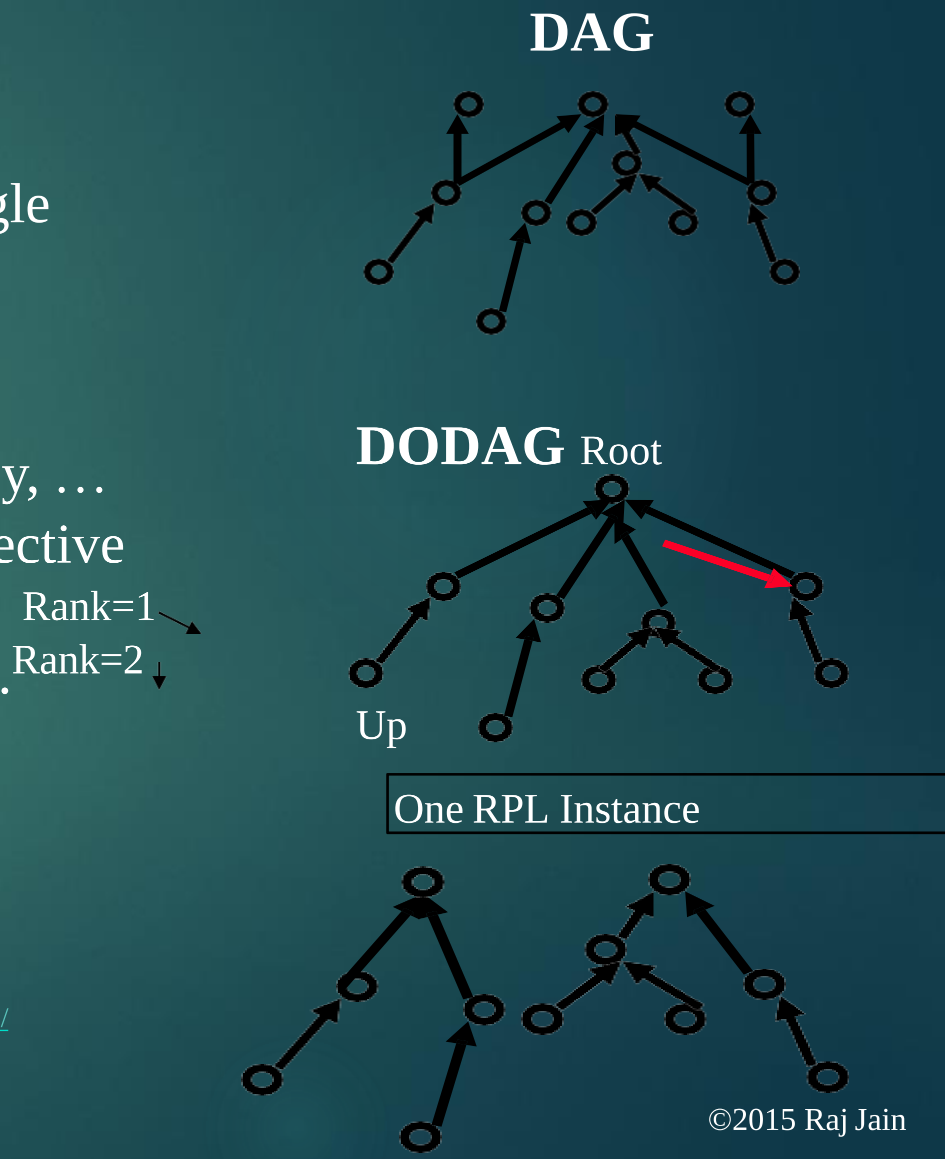
# Routing Protocol for Low-Power and Lossy Networks (RPL)

- Developed by IETF Routing over Low-Power and Lossy Networks (ROLL) working group
- Low-Power and Lossy Networks (LLN) Routers have constraints on processing, memory, and energy.
  ⇒ Can't use OSPF, OLSR, RIP, AODV, DSR, etc
- LLN links have high loss rate, low data rates, and instability
  ⇒ expensive bits, dynamically formed topology
- Covers both wireless and wired networks
  Requires **bidirectional** links. May be symmetric/asymmetric.
- Ideal for n-to-1 (**data sink**) communications, e.g., meter reading
  1-to-n and 1-to-1 possible with some extra work.
- Multiple LLN instances on the same physical networks

Ref: T. Winder, Ed., et al., "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," IETF RFC 6550, Mar 2012, https://ietf.org/doc/rfc6550/

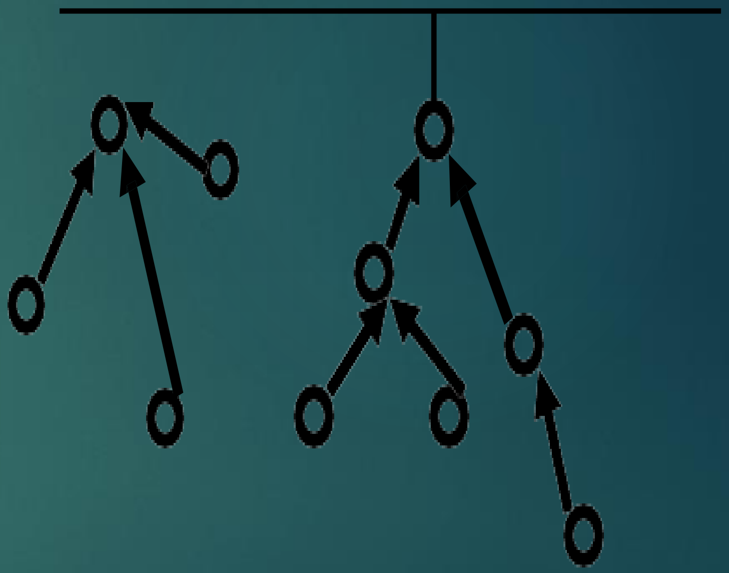
# RPL Concepts

- **Directed Acyclic Graph (DAG)**: No cycles
- **Root**: No outgoing edge
- **Destination-Oriented DAG (DODAG)**: Single root
- **Up**: Towards root
- **Down**: Away from root
- **Objective Function**: Minimize energy, latency, …
- **Rank**: Distance from root using specified objective
- **RPL Instance**: One or more DODAGs. A node may belong to multiple RPL instances.
- **DODAG ID**: IPv6 Adr of the root
- **DODAG Version**: Current version of the DODAG. Every time a new DODAG is computed with the same root, its version incremented.



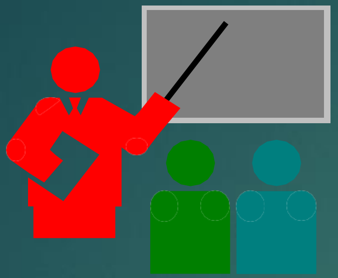
# RPL Concepts (Cont)

- **Goal**: Reachability goal, e.g., connected to database
- **Grounded**: Root can satisfy the goal
- **Floating**: Not grounded. Only in-DODAG communication.
- **Parent**: Immediate successor towards the root
- **Sub-DODAG**: Sub tree rooted at this node
- **Storing**: Nodes keep routing tables for sub-DODAG
- **Non-Storing**: Nodes know only parent. Do not keep a routing table.



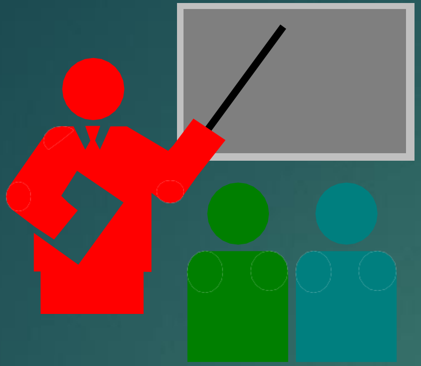
## RPL

Routing Protocol for Low-Power and Lossy Networks (RPL) is distance-vector protocol that can support a variety of datalink protocols, including the ones discussed in the previous section. It builds a Destination Oriented Directed Acyclic Graph (DODAG) that has only one route from each leaf node to the root in which all the traffic from the node will be routed to. At first, each node sends a DODAG Information Object (DIO) advertising itself as the root. This message is propagated in the network and the whole DODAG is gradually built. When communicating, the node sends a Destination Advertisement Object (DAO) to its parents, the DAO is propagated to the root and the root decides where to send it depending on the destination. When a new node wants to join the network, it sends a DODAG Information Solicitation (DIS) request to join the network and the root will reply back with a DAO Acknowledgment (DAO-ACK) confirming the join. RPL nodes can be stateless, which is most common, or stateful. A stateless node keeps tracks of its parents only. Only root has the complete knowledge of the entire DODAG. Hence, all communications go through the root in every case. A stateful node keeps track of its children and parents and hence when communicating inside a sub-tree of the DODAG, it does not have to go through the root [RFC6550].



## RPL Summary

1. An RPL instance consists of one or more DODAGs
2. DIO are broadcast downward,  
   DAOs are requests to join upward  
   DIS are DIO solicitations  
   DAO-ack are responses to DAO
3. Non-storing nodes do not keep any routing table and send everything upwards toward the root



# Summary

1. 6LowPAN is designed for IPv6 over IEEE 802.15.4
   Frame size and address sizes are primary issues
   Header compression is the key mechanism
2. RPL is designed primarily for data collection
   No assumption about IEEE 802.15.4 or wireless or frame size
   Routing is the primary issue
   Forming a spanning tree like DODAG is the solution

## CARP

- Channel-Aware Routing Protocol (CARP) is a distributed routing protocol designed for underwater communication. It can be used for IoT due to its lightweight packets. It considers link quality, which is computed based on historical successful data transmission gathered fromneighboring sensors, to select the forwarding nodes. There are two scenarios: network initialization and data forwarding. In network initialization, a HELLO packet is broadcasted from the sink to all other nodes in the networks. In data forwarding, the packet is routed from sensor to sink in a hop- by-hop fashion. Each next hop is determined independently. The main problem with CARP is that it does not support reusability of previously collected data. In other words, if the application requires sensor data only when it changes significantly, then CARP dataforwarding is not beneficial to that specific application. An enhancement of CARP was done in E-CARP by allowing the sink node to save previously received sensory data. When new data is needed, E-CARP sends a Ping packet which is replied with the data from the sensors nodes. Thus, E-CARP reduces the communication overhead drastically [Shou15].

## Network Layer Encapsulation Protocols

- One problem in IoT applications is that IPv6 addresses are too long and cannot fit in most IoT datalink frames which are relatively much smaller. Hence, IETF is developing a set of standards to encapsulate IPv6 datagrams in different datalink layer frames for use in IoT applications. In this section, we review these mechanisms briefly.

## 6TiSCH

- 6TiSCH working group in IETF is developing standards to allow IPv6 to pass through TimeSlotted Channel Hopping (TSCH) mode of IEEE 802.15.4e datalinks. It defines a Channel Distribution usage matrix consisting of available frequencies in columns and time-slots available for network scheduling operations in rows. This matrix is portioned into chucks where each chunk contains time and frequencies and is globally known to all nodes in the network.

# DHCP: Dynamic Host Configuration Protocol

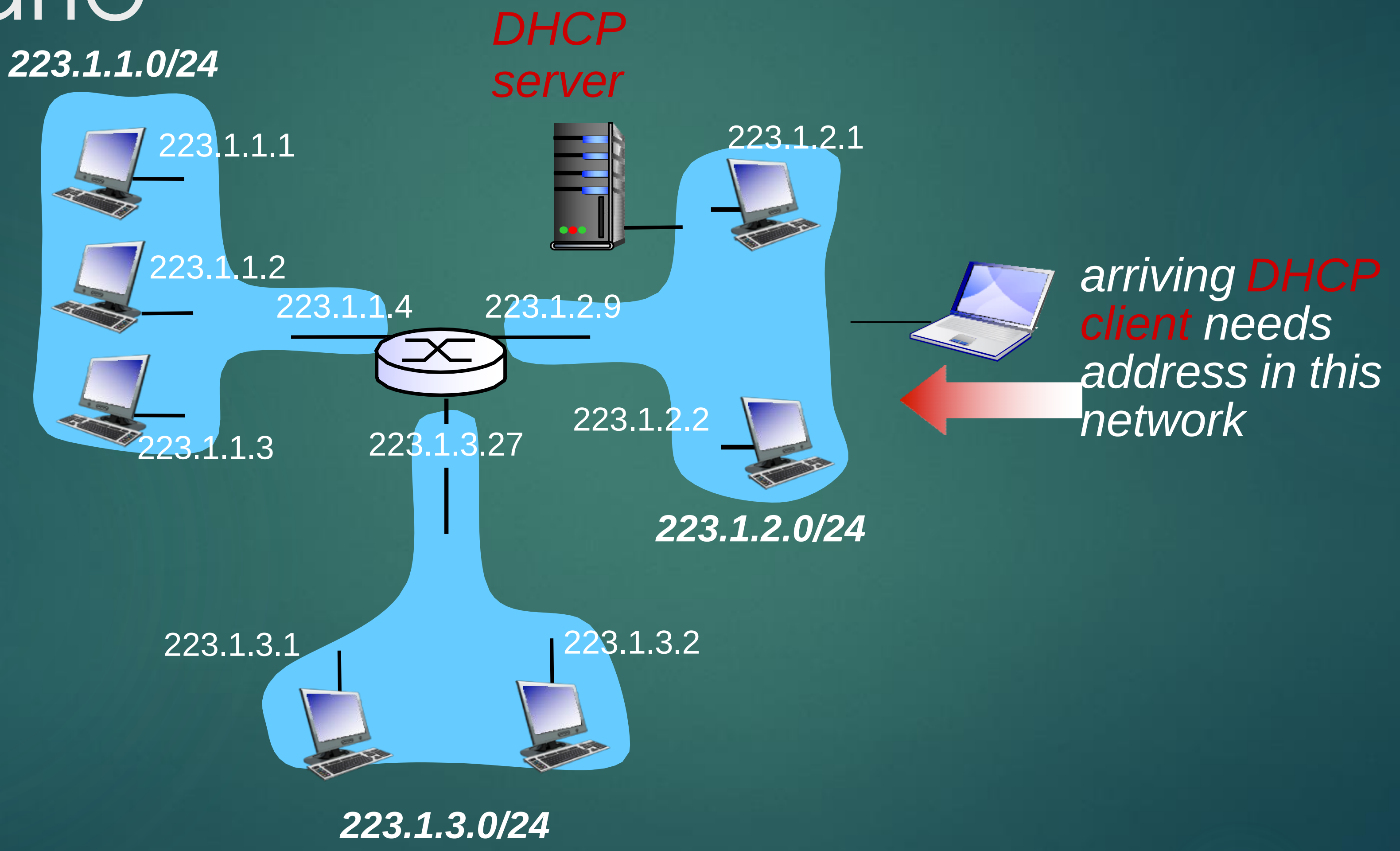
*goal:* allow host to *dynamically* obtain it‘s IP address from network server when it joins network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/“on”)
- support for mobile users who want to join network

*DHCP overview:*

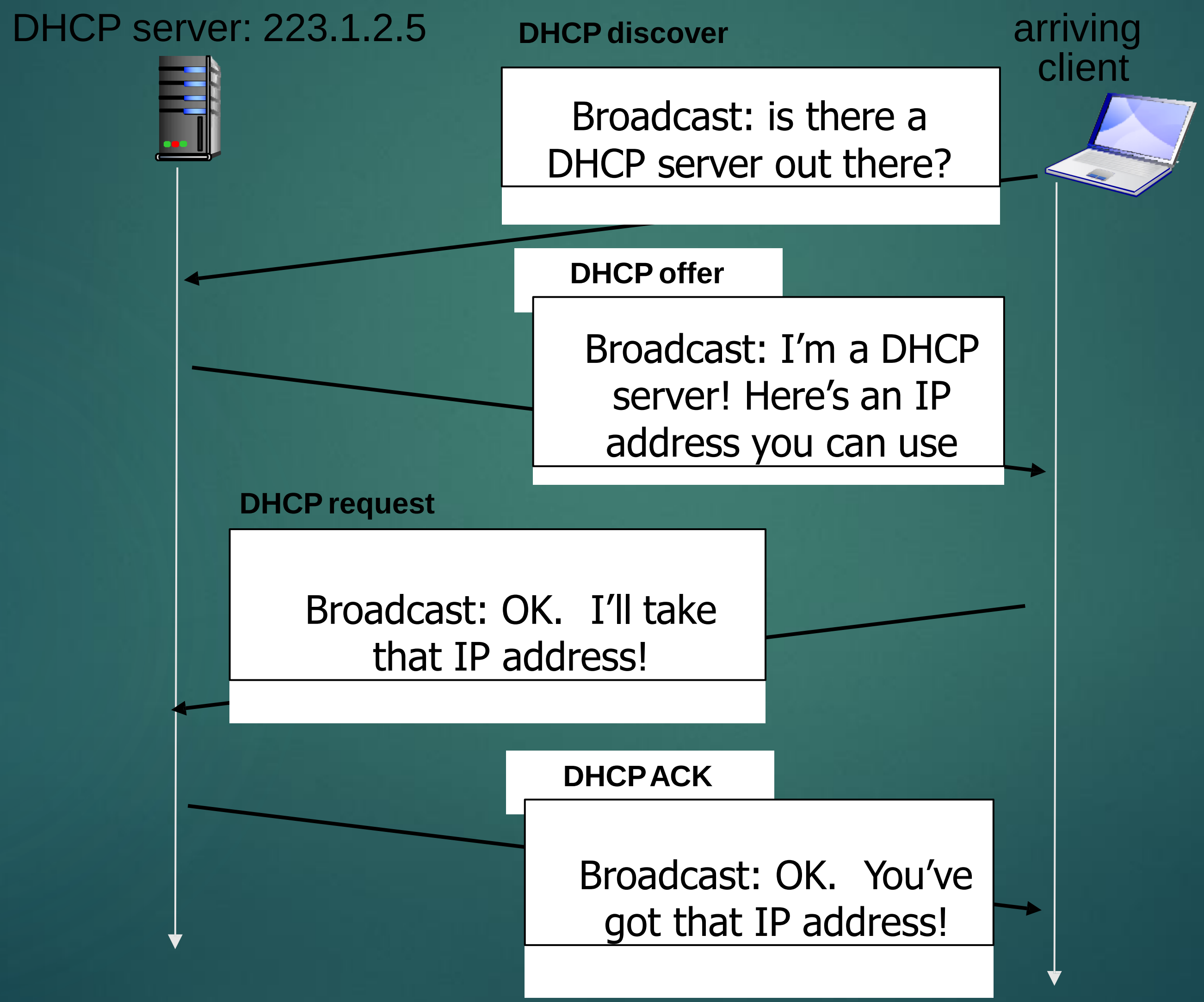
- host broadcasts “DHCP discover” msg [optional]
- DHCP server responds with “DHCP offer” msg [optional]
- host requests IP address: “DHCP request” msg
- DHCP server sends address: “DHCP ack” msg

# DHCP client-server scenario



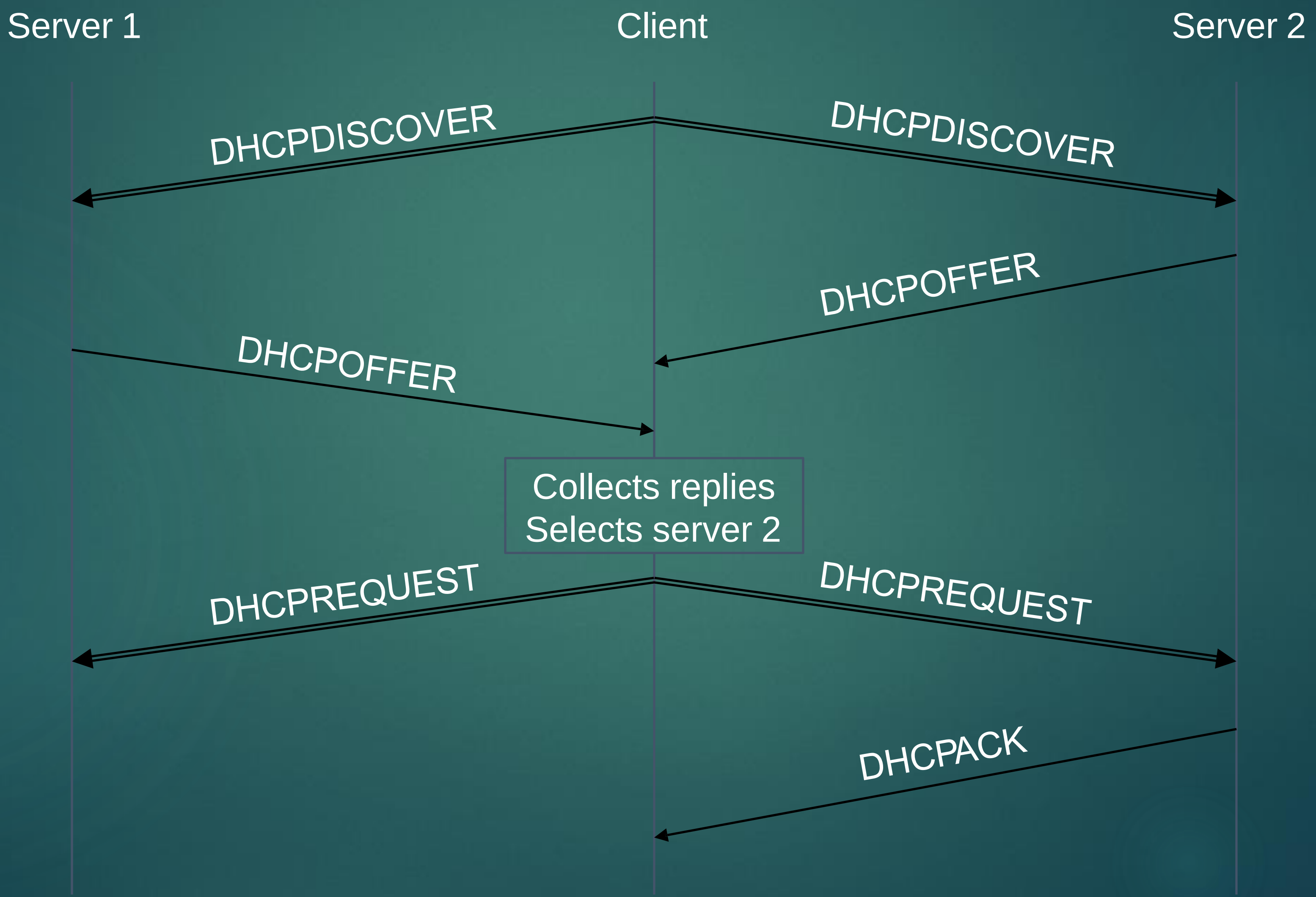
arriving DHCP client needs address in this network

# DHCP client-server scenario



DHCP server: 223.1.2.5
DHCP discover
arriving client
Broadcast: is there a DHCP server out there?
DHCP offer
Broadcast: I'm a DHCP server! Here's an IP address you can use
DHCP request
Broadcast: OK. I'll take that IP address!
DHCP ACK
Broadcast: OK. You've got that IP address!

# DHCP Protocol



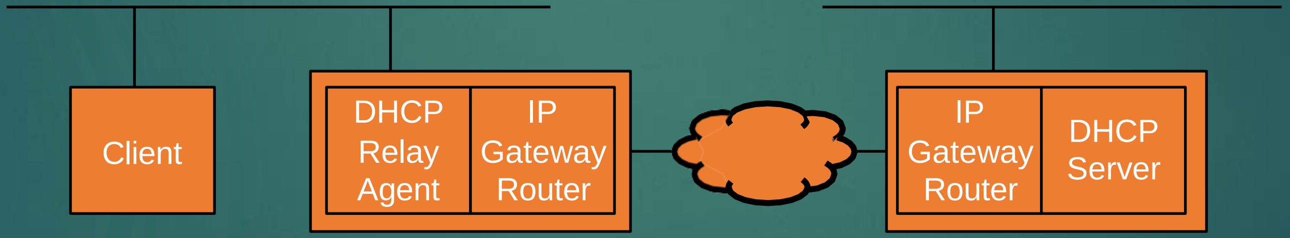
## DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client to reach other subnets
- name and IP address of the local DNS sever
- subnet mask

# DHCP Relay Agents

- DHCP relay agents allow DHCP servers to handle requests from other subnets



## Summary

- IP addresses don’t have to be manually configured into hosts
- DHCP allows “ignorant” hosts to receive IP addresses (and more) at start-up time
- DHCP solves important bootstrapping problems in attaching new hosts to a network

## Internet Control Message Protocol (ICMP)

## ICMP

- Protocol for error detection and reporting
    - tightly coupled with IP, unreliable
- ICMP messages delivered in IP packets
- ICMP functions:
    - Announce reachability and network errors
    - Announce “time exceeded” errors for IP packets
    - Announce network congestion
- ICMP assists network troubleshooting in general

## ICMP message

**IP header**
Source, Destination Address, TTL, ...

38

**ICMP MSG**
Message type, Code, Checksum,
Data

## ICMP: Internet Control Message Protocol

| Type | Code | description |
|---|---|---|
| 0 | 0 | echo reply (ping) |
| 3 | 0 | dest. network unreachable |
| 3 | 1 | dest host unreachable |
| 3 | 2 | dest protocol unreachable |
| 3 | 3 | dest port unreachable |
| 3 | 6 | dest network unknown |
| 3 | 7 | dest host unknown |
| 4 | 0 | source quench (congestion control - not used) |
| 8 | 0 | echo request (ping) |
| 9 | 0 | route advertisement |
| 10 | 0 | router discovery |
| 11 | 0 | TTL expired |
| 12 | 0 | bad IP header |

Time for an activity!

## Specific uses of ICMP

- Echo request reply
  - Can be used to check if a host is alive
- Destination unreachable
  - Invalid address and/or port
- TTL expired
  - Routing loops, or too far away

THANK YOU