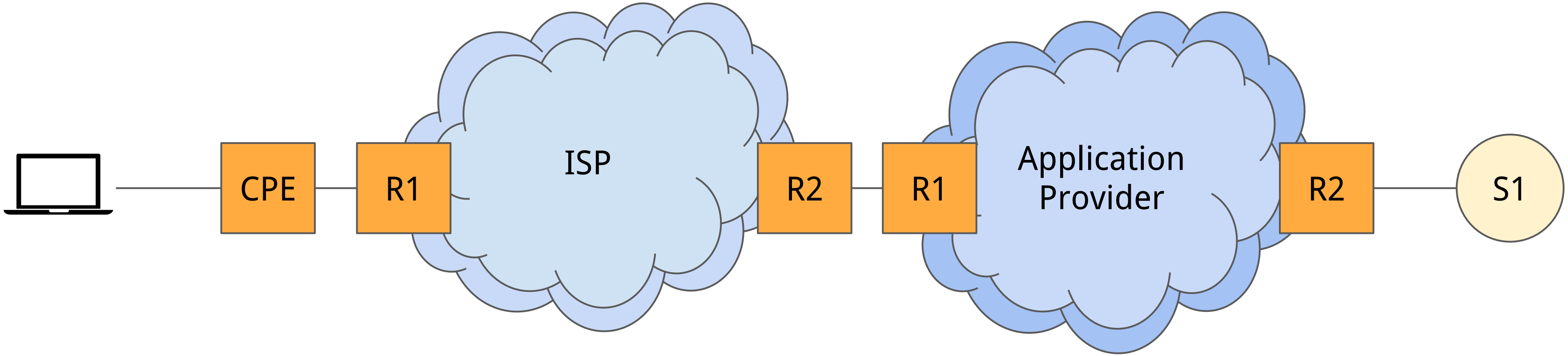
Ethernet, ARP, and DHCP

Spring 2024
[cs168.io](cs168.io)

Rob Shakir

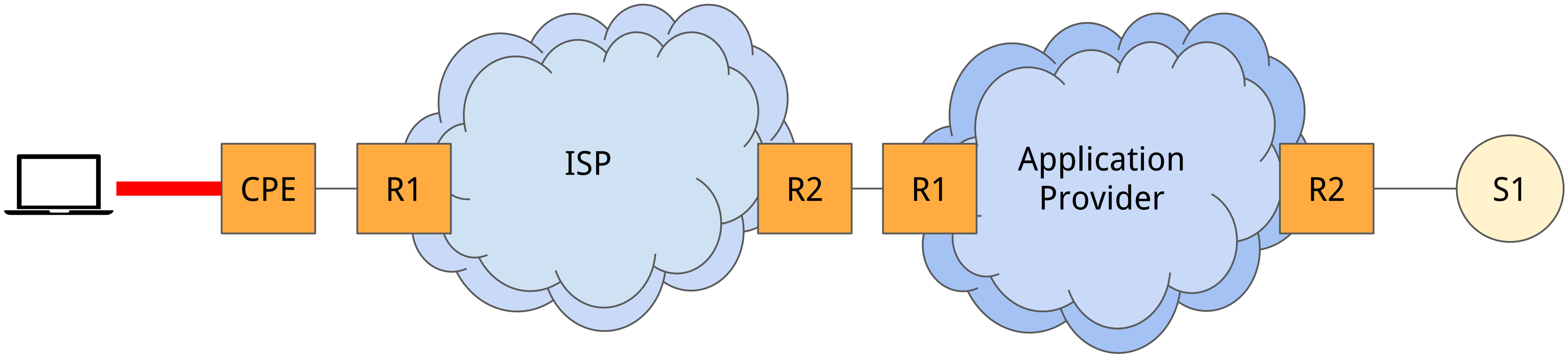
Thanks to Murphy McCauley for some of the material!

Before the break…



This week – we'll think about putting together the end-to-end picture.

Today - **local area networking.**



We've thought about the wide area and datacentre network - but not our local network.

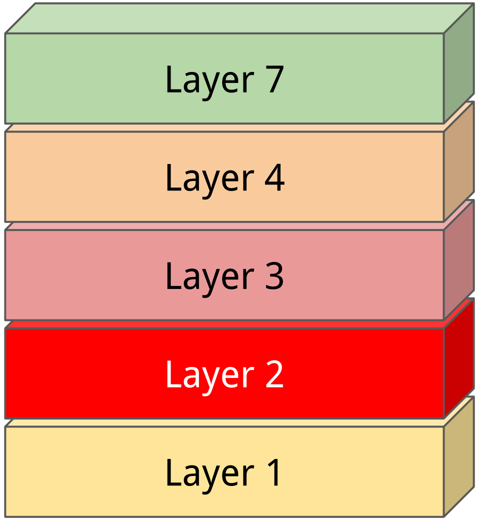
What happens in a LAN?

- Forwarding local hosts → router, and host → host packets.
  - Layer 2 forwarding – focus on Ethernet today.
  - Layer 2 addressing.

- Discovering local IP (Layer 3) addresses and routers.
  - DHCP, SLAAC.

- Fitting different types of addresses together:
  - How do Layer 3 addresses and Layer 2 addresses fit together? (ND, ARP)
  - How do private IP addresses and Internet addresses fit together? (NAT)

Looking at Layers Again



Application → DNS, HTTP, …

Transport → TCP

Network → IP

Data Link → Ethernet

Physical → Optical fibre, copper

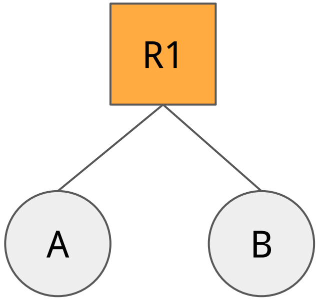
We skipped over this layer previously - but it is vital to our end-to-end picture!

Layer 2 – Ethernet

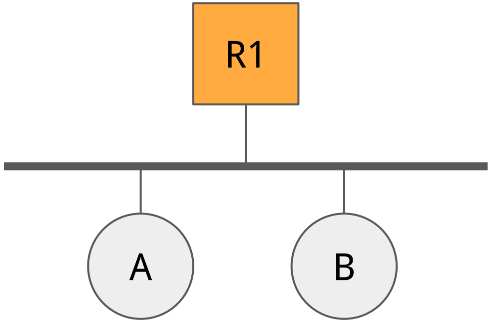
Connecting local hosts together

- We have generally shown one host connected to a router.
  - Or assumed all packets go via a router.

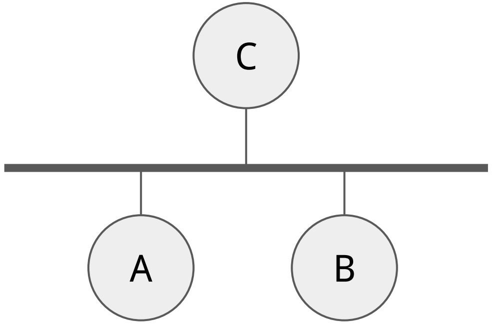
- But in our local network – we might have multiple computers that are connected within the same network.



Our assumed view…

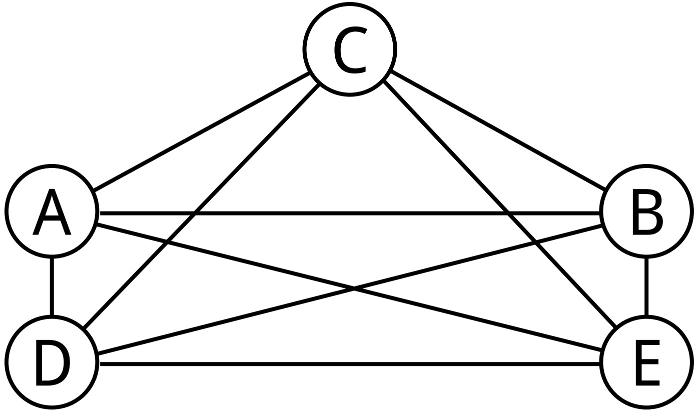


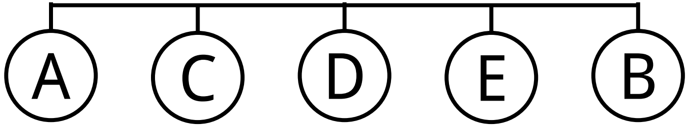
…really looks like this…



…or this!

How could we build this local network?





Meshes – needs a lot of cabling and a lot of ports per host.

A <u>bus</u> approach - introduces a **shared media**.

Some history - ALOHANet

- Norman Abramson had a problem at the University of Hawaii in 1968.
  - How do we allow people on other islands access to the U of H computer?

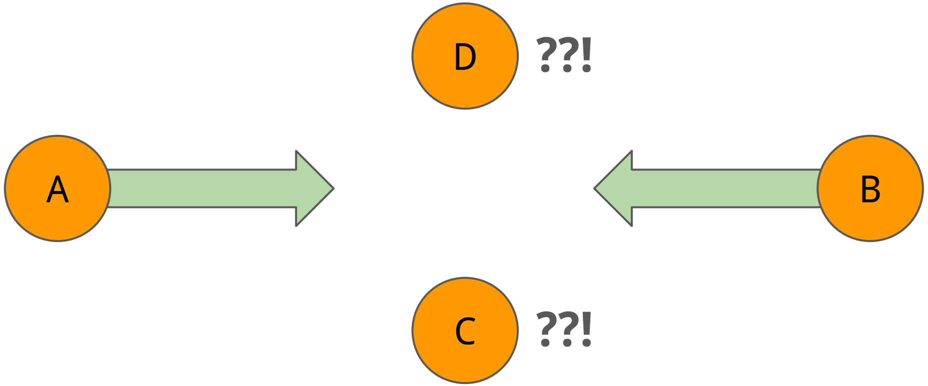
- ALOHANet
  - Additive Links On-line Hawaii Area
  - Wireless communications from terminals (computers) on other islands.
  - Hugely influential.

- ALOHANet was <u>wireless</u> but in these radio networks there is a *shared medium* (the electromagnetic spectrum).
  - Similar problem to the *shared media* in our local bus network.
  - And our bus network *could* have been wireless (we'll come back to WiFi).

Shared Media

- In a network with a *shared medium*, then transmissions from different nodes may interfere or *collide* with each other.

- We need a way to allocate the medium to everyone wanting to use it…
    - A **multiple access protocol**.



Common Multiple Access Protocol approaches

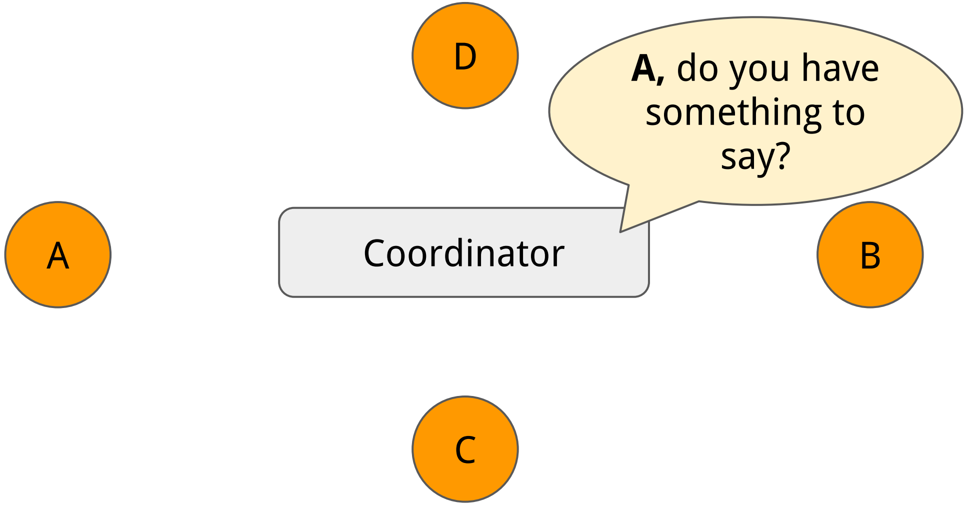
- Divide the medium by frequency – **frequency-division multiplexing**.
    - Give each connected node some slice of frequencies.
    - Can be wasteful – only a specific amount of frequency to allocate.
    - Not everyone has something to say all the time (many frequencies idle).

- Divide the medium by time - **time-division multiplexing.**
    - Divide time into fixed slots and allocate them to each connected node.
    - Same downside – only so much time, many slots are idle.

- Alternative: can connected nodes take turns?

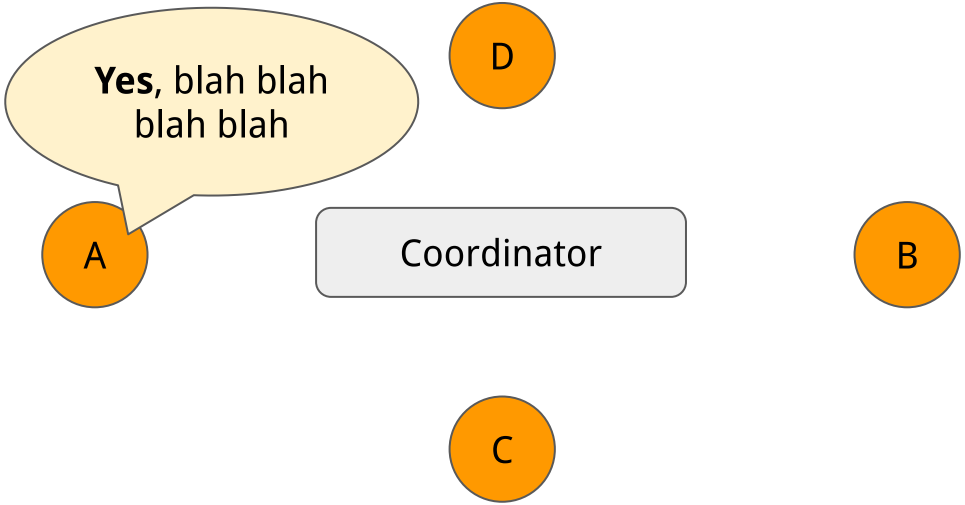
Turn-taking Schemes

- Polling protocols.
  - A coordinator decides when each connected node can speak.
  - e.g., Bluetooth



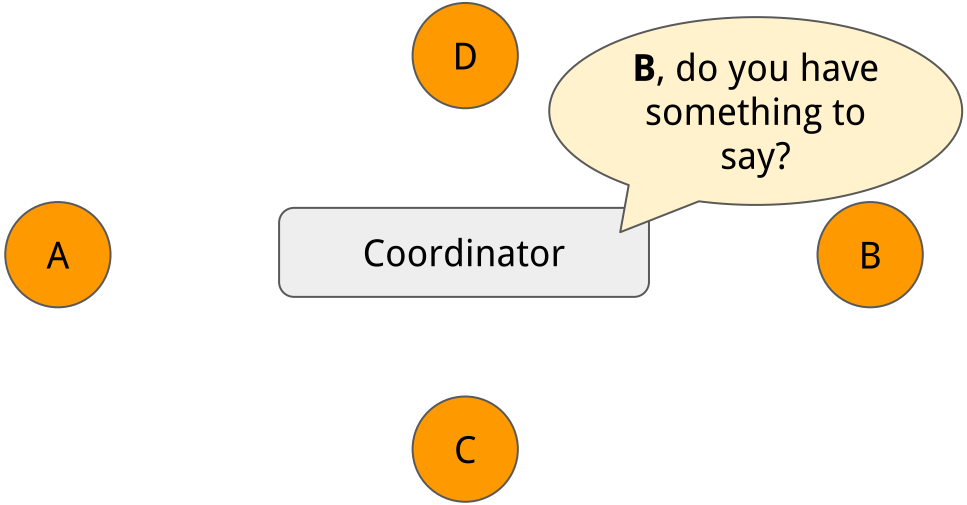
Turn-taking Schemes

- Polling protocols.
  - A coordinator decides when each connected node can speak.
  - e.g., Bluetooth



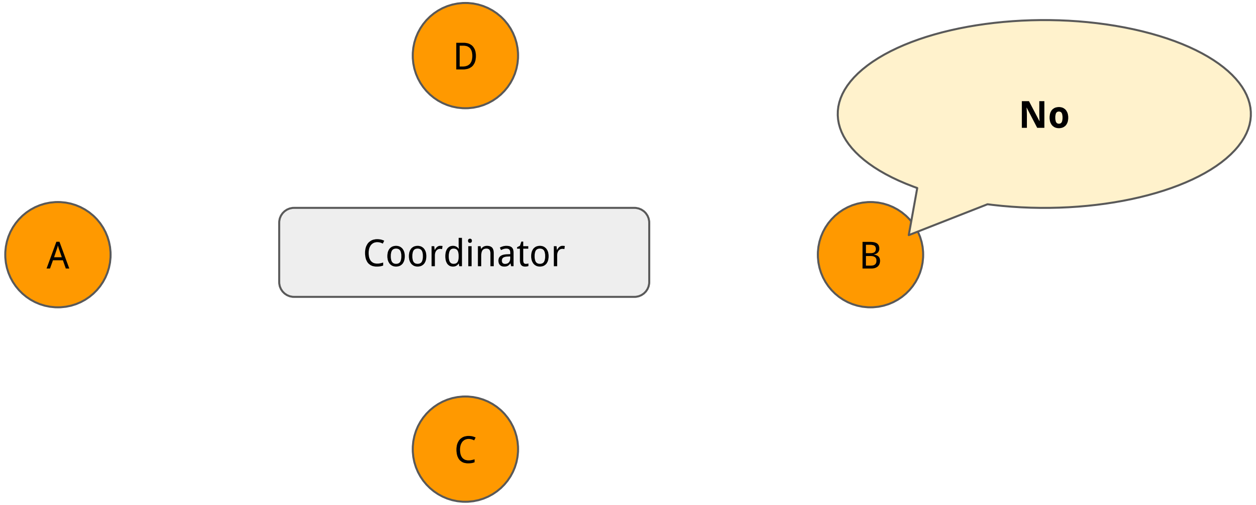
Turn-taking Schemes

- Polling protocols.
  - A coordinator decides when each connected node can speak.
  - e.g., Bluetooth



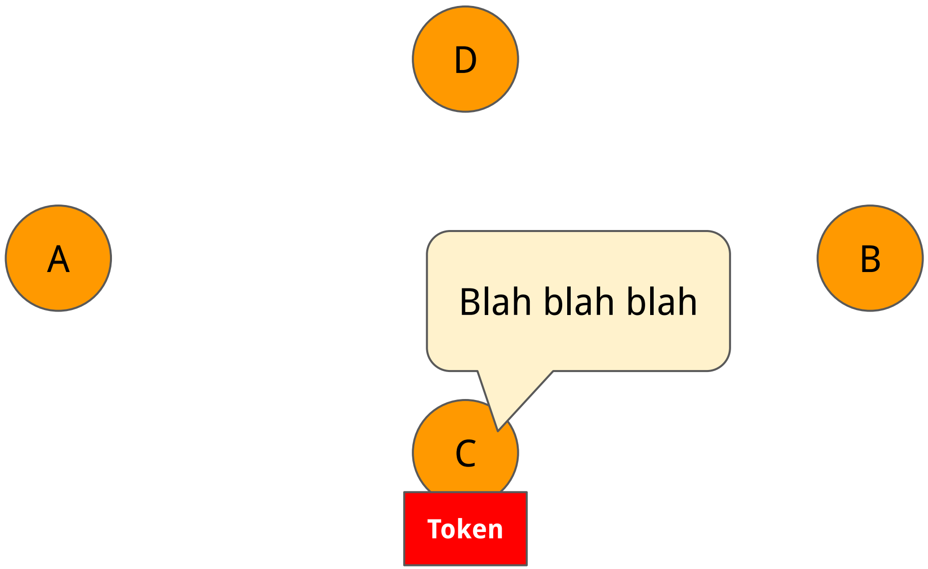
Turn-taking Schemes

- Polling protocols.
  - A coordinator decides when each connected node can speak.
  - e.g., Bluetooth



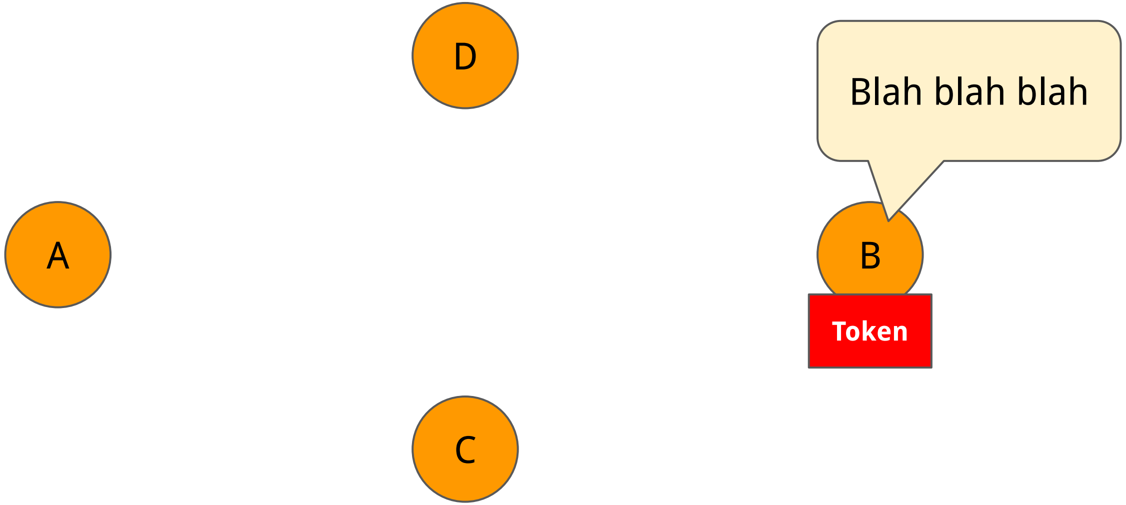
Turn-taking Schemes

- Token-passing
  - Virtual “token” passed around, only the holder can transmit.
  - IBM Token Ring and FDDI.



Turn-taking Schemes

- Token-passing
  - Virtual “token” passed around, only the holder can transmit.
  - IBM Token Ring and FDDI.



Alternative – *Random Access*

- Both of these mechanisms are **partitioning approaches**.
  - Essentially, we are dividing by time – but more dynamically.
  - Require some form of inter-node communication.

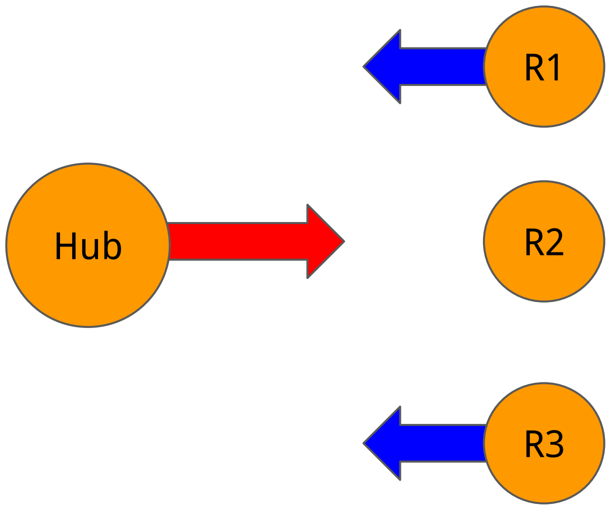
- An alternate idea – just allow for nodes to talk when they have something to say.
  - And deal with collisions when they occur.

- Used by ALOHANet and then later in Ethernet.

ALOHANet's Random Access

- Hub node on Oahu.
- Remote nodes across Hawaii.

- Used two frequencies:
  - **Hub transmits on its own frequency.**
    - Only one sender – no collisions.
    - All remote nodes listen to this frequency.
  - **All remote sites transmit on one frequency.**
    - May collide.
    - Only the hub listens to the remote frequency.



ALOHANet: Pure ALOHA random access scheme

- If remote has a packet – just send it.
  - No *a priori* coordination among remote sites.

- When the hub gets a packet – send ACK.

- If two remote sites transmitted at once, collisions results in a corrupted packet.
  - Hub doesn’t ACK!

- If a remote sender doesn’t get the expected ACK – then:
  - Wait a random amount of time.
  - Then resend, probably avoiding collisions this time.

Questions?

Ethernet and CSMA

- Ethernet – used as the most common wired *Data Link* protocol.

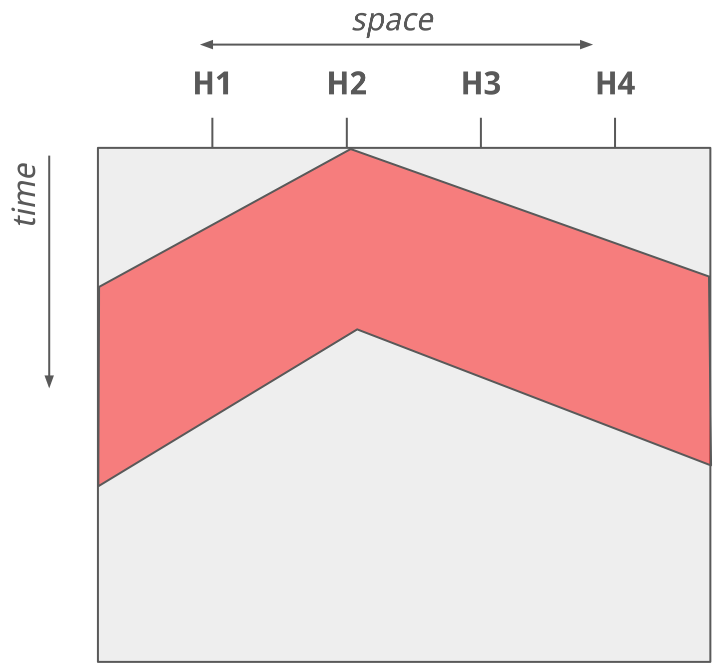
- Refined the ALOHA multiple access protocol to allow access to a shared Ethernet bus resulting in ***Carrier Sense Multiple Access*** **(CSMA).**

- Where ALOHA is rude, CSMA is polite.
  - Rather than just starting talking, and dealing with collisions…
  - CSMA listens first, and then starts to talk when it is quiet.
  - “Listen” means sensing the signal (carrier) on the shared medium.

Ethernet: CSMA and propagation delay

- CSMA does not necessarily avoid collisions – because of **propagation delay**.

- t=0:
  - H2 transmits.
  - Signal propagates through the shared media.



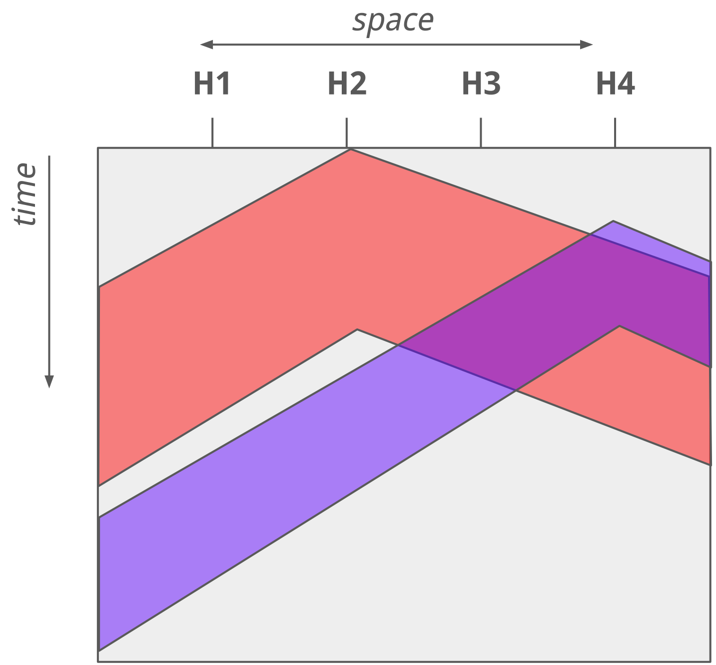
Ethernet: CSMA and propagation delay

- CSMA does not necessarily avoid collisions – because of **propagation delay**.

- t=0:
    - H2 transmits.
    - Signal propagates through the shared media.

- t=2:
    - H3 has heard, won't transmit.
    - H4 has not heard – it's safe to transmit!
        - Signal propagates as time goes by
        - …and collides with H2's signal.

- Solution: CSMA/CD.



Ethernet: CSMA/CD

- *Carrier Sense Multiple Access with Collision Detection* (CSMA/CD).

- Modification to the previous approach:
  - Listen <u>whilst</u> you talk.

  - If you start hearing something whilst you are still transmitting **– stop!**
    - Hence - detect the collision.

- Some additional complexities – but this is the core idea.

- What do we do after detecting a collision?

Ethernet: CSMA/CD

- After collision – wait a random amount of time and retransmit.
- If the link has many senders who want to talk (has high contention) we may keep colliding.
- Use randomised *binary exponential backoff…*
  - If retransmit after collision also collides, wait up to twice as long.
  - Continue doubling for every subsequent collision.
  - Retransmits fast when possible, slowing down where necessary.

Questions?

Ethernet as the LAN network protocol

- Local Area Networks are generally Ethernet.
- Gives us a way to have different machines send signals to each other directly.
    - Which gives us a good way communicate with other local computers!
- Analogy: many people in the same room – we can talk to each other locally without going via the postal system.

Ethernet Addressing

- If I send a signal (shout in this room) – everyone gets the message.

- But we do want some way to be able to identify the destination of a particular message.
  - E.g., just talk to one person in the room – not talk to everyone!

- We therefore need some form of addressing to be able to identify different hosts connected to the same medium.
  - Like we would use a *name* within this room to talk to one another.

Ethernet: Addresses

- Ethernet has Media Access Control (MAC) addresses.
  - These are Layer 2 addresses – we don't need to know anything about what is inside the Ethernet *Frame* (i.e., it doesn't matter whether it's IPv4, IPv6, or even IP at all!)
  - Remember - our layers build on top of one another.

- MAC addresses are 48-bits.
  - Usually shown as six two-digit hex numbers with colons.
  - Sometimes referred to as `ether` or `link` addresses.

```
▸ ifconfig en0
en0:
flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST>
mtu 1500
        options=400<CHANNEL_IO>
        ether f8:ff:c2:2b:36:16
```

```
rjs@jumphost:~$ ip link show ens4
2: ens4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1460
qdisc mq state UP mode DEFAULT group default qlen 1000
    link/ether 42:01:0a:8a:00:03 brd ff:ff:ff:ff:ff:ff
    altname enp0s4
```

Ethernet: MAC Addresses

- MAC addresses are allocated according to organisation.
  - Usually the manufacturer of the Ethernet network interface card (NIC).
- Typically stored permanently in the NIC (“burned in”) .
  - Often can be overridden by software.
- Structure:
  - Two bits of flags (we won’t discuss this)
  - 22-bits identifying company/organisation (e.g., device manufacturer)
  - 24-bits of identifying space.
- Usually supposed to be globally unique.
  - You might plug your computer in *anywhere*…

Ethernet: Types of communication

- We have typically talked about **unicast.**
  - Send to any one recipient.

- We've mentioned other types:
  - Anycast – send to any <u>one</u> member of a group.

- There are other models that we might care about:
  - Speak to everyone in the room – **broadcast**.
  - Speak to everyone who has joined a group in the room – **multicast**.

- Ethernet supports both multicast and broadcast.
  - And generally they are not distinguished from each other at the Ethernet level.

Ethernet: Unicast

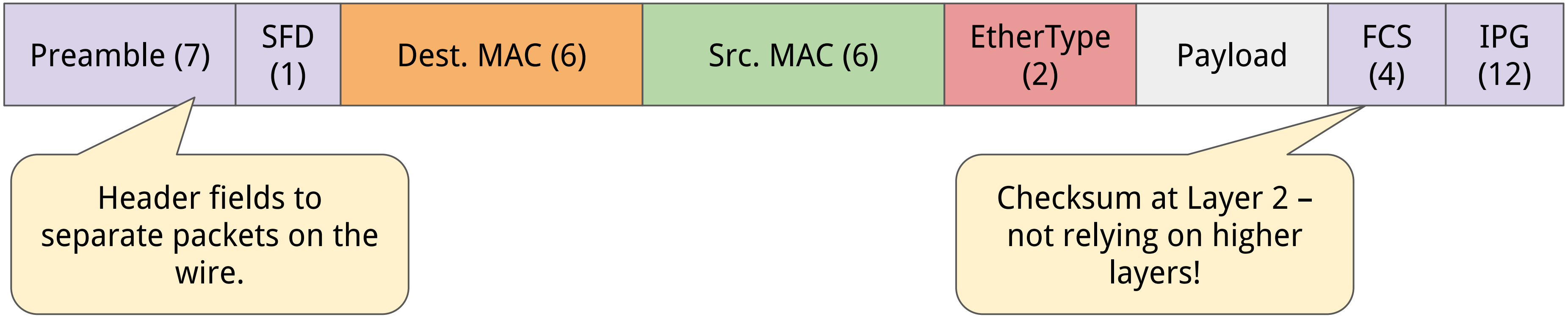
- Unicast is the typical type of communication we have talked about.
  - A source host wants to talk to a specific destination host.

- The Ethernet header has the same types of fields as those that we talked about in IP for this purpose.
  - A data packet in Ethernet is referred to as a *frame*.

Preamble (7)	SFD (1)	Dest. MAC (6)	Src. MAC (6)	EtherType (2)	Payload	FCS (4)	IPG (12)

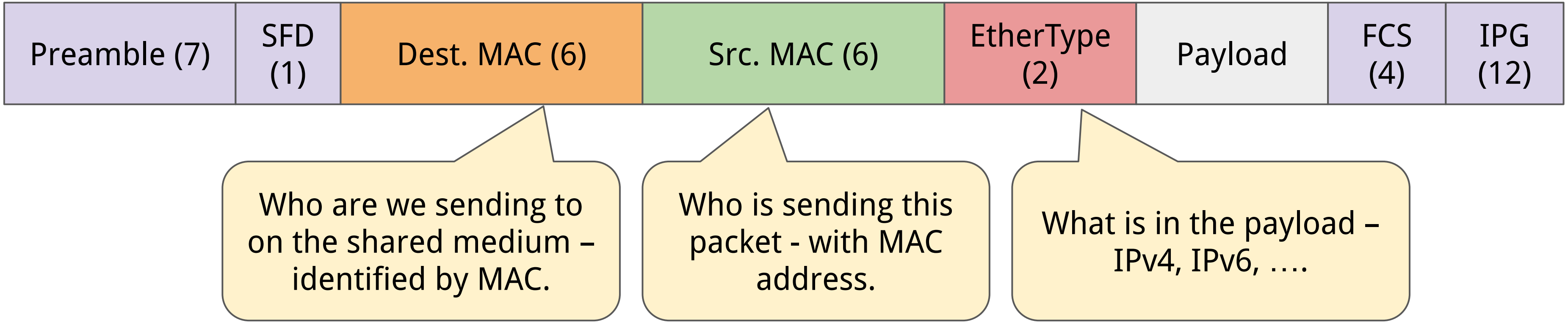
Ethernet: Unicast

- Unicast is the typical type of communication we have talked about.
  - A source host wants to talk to a specific destination host.
- The Ethernet header has the same types of fields as those that we talked about in IP for this purpose.
  - A data packet in Ethernet is referred to as a *frame*.



Ethernet: Unicast

- Unicast is the typical type of communication we have talked about.
  - A source host wants to talk to a specific destination host.
- The Ethernet header has the same types of fields as those that we talked about in IP for this purpose.
  - A data packet in Ethernet is referred to as a *frame*.



Ethernet: Unicast

- To send a packet to a specific destination host - we set the destination MAC to a specific remote machine’s MAC address.

- Packets go to everyone on the shared medium (wire).

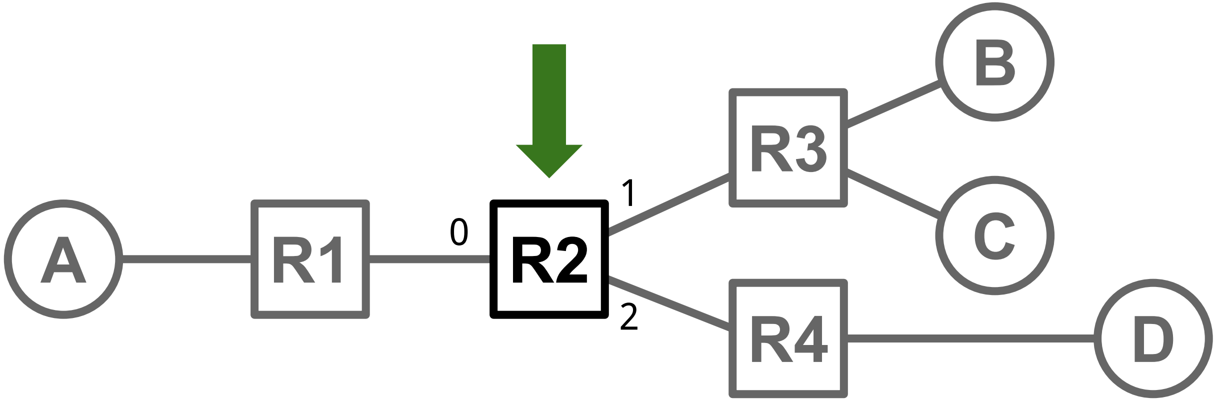
- Receivers check the destination MAC to determine whether the packet is destined to them.
  - “Is dst MAC == `42:01:0a:8a:00:03`? It’s for me!”

Ethernet: L2 Networks

- We have not explored Layer 2 networks – but given that we have a source and destination address, we can build networks at Layer 2.

- Our host ID when talking about routing was an IP address – but it could just be the MAC address.

Thinking back to routing…

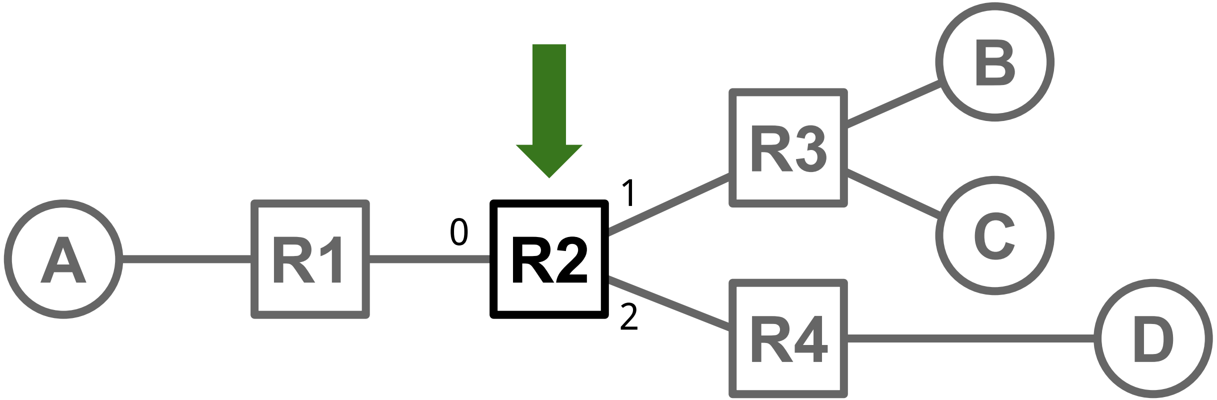


R2's Table	
Dst	*NextHop*
A	R1
B	R3
C	R3
D	R4

.. or ..

R2's Table	
Dst	*Port*
A	0
B	1
C	1
D	2

Thinking back to routing…



Destination could be our MAC addresses.

*R2's **Forwarding** Table*	
Dst	***NextHop***
A	R1
B	R3
C	R3
D	R4

*R2's **Routing** Table*	
Dst	***Port***
A	0
B	1
C	1
D	2

MAC addresses are not aggregatable – allocated by manufacturer.

Ethernet: Broadcast

- Broadcast – send to everyone!
  - Specifically, everyone on the specific Ethernet network…
  - …everyone on the same cable.
- The packet already reaches everyone – they are connected to the *shared media.*
  - We need receivers to listen.
- Broadcast is implemented using the all ones address.
  - `FF:FF:FF:FF:FF:FF`
- Any Layer 2 *switch* needs to know to send this address to all ports.

Ethernet: Multicast

- Multicast – send to all members of a group.
  - Trivial on classic Ethernet – since everyone gets the packet.

- Implemented by having specific addresses – one of the flags in the address set to 1.
  - `01:00:00:00:00:00`
  - Normal addresses all have an even first byte.
  - This 1 is the first bit on the wire – bytes are sent low bit first.

- Broadcast is just a special case of multicast – where everyone is in a group.
- Layer 2 networking for <u>multicast</u> gets more complicated.
  - Need to know who is in the group.
  - Similarly complicated at Layer 3.

Why do we need multicast in a LAN?

- Apple invention: Bonjour/mDNS.

- iPhone wants to discover any Apple TV, or HomePod that it can play music on.
  - It can actively discover this “hey local Apple products, are there any speakers?”.
  - Sends to a multicast group that all Apple products join by default.
  - Equally, HomePod/Apple TV can advertise “I am an Apple TV!”.

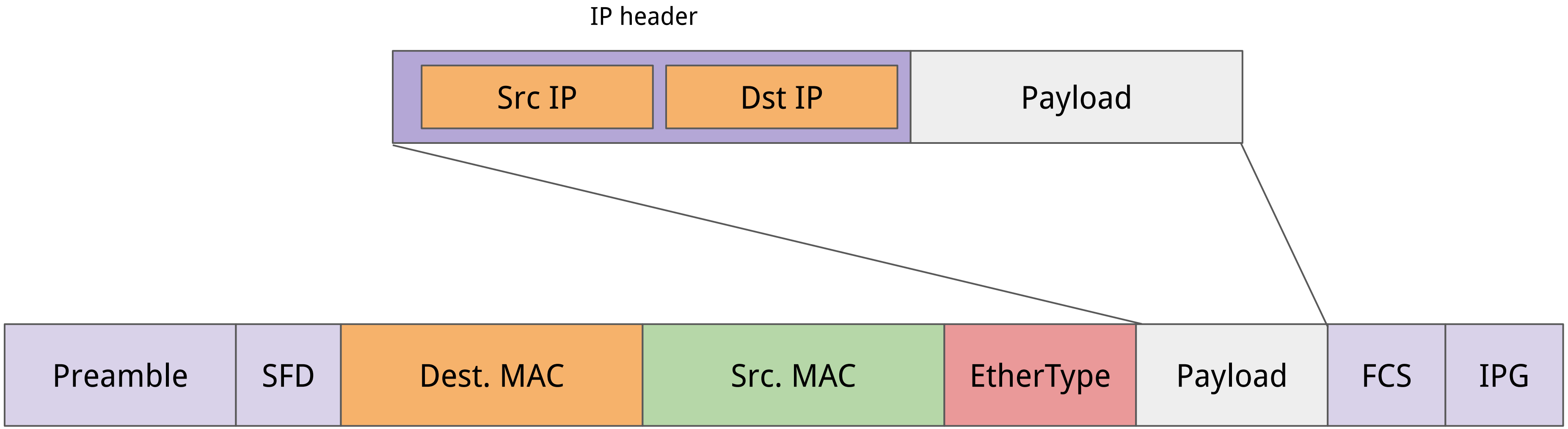
- Actually uses DNS advertisements that are sent to multicast addresses.
  - Using specific types of records – e.g., SRV – to advertise capabilities.

Questions?

How do Layer 2 and Layer 3 work together?

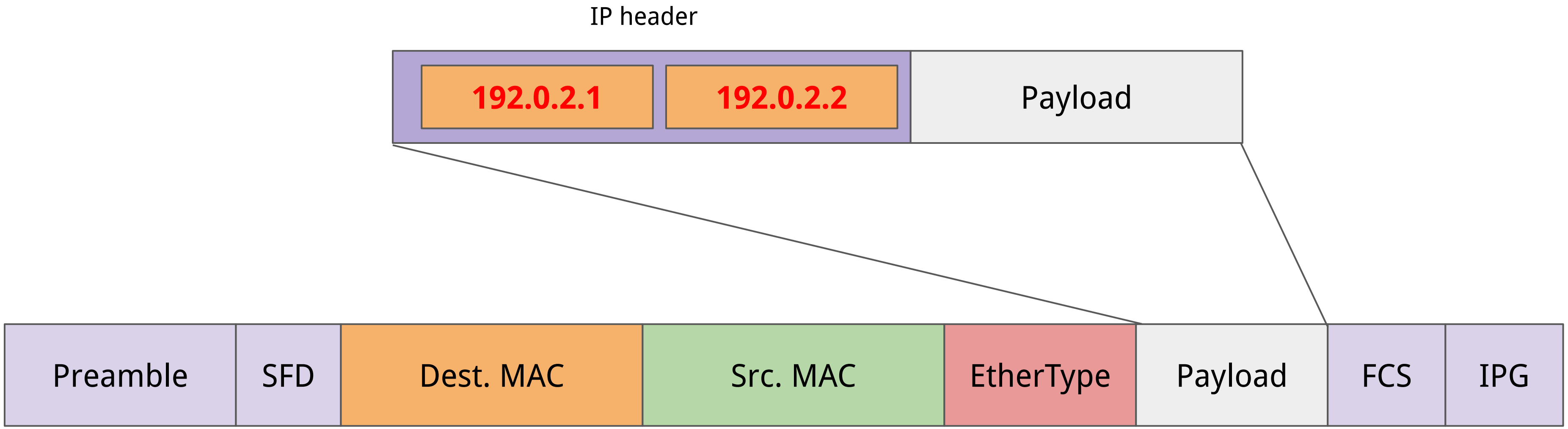
Sending an IP packet to a local host

- Local routing table says that our subnet is local.
    - `192.0.2.0/24` means that anything in `192.0.2.X` is connected to the same network as us.



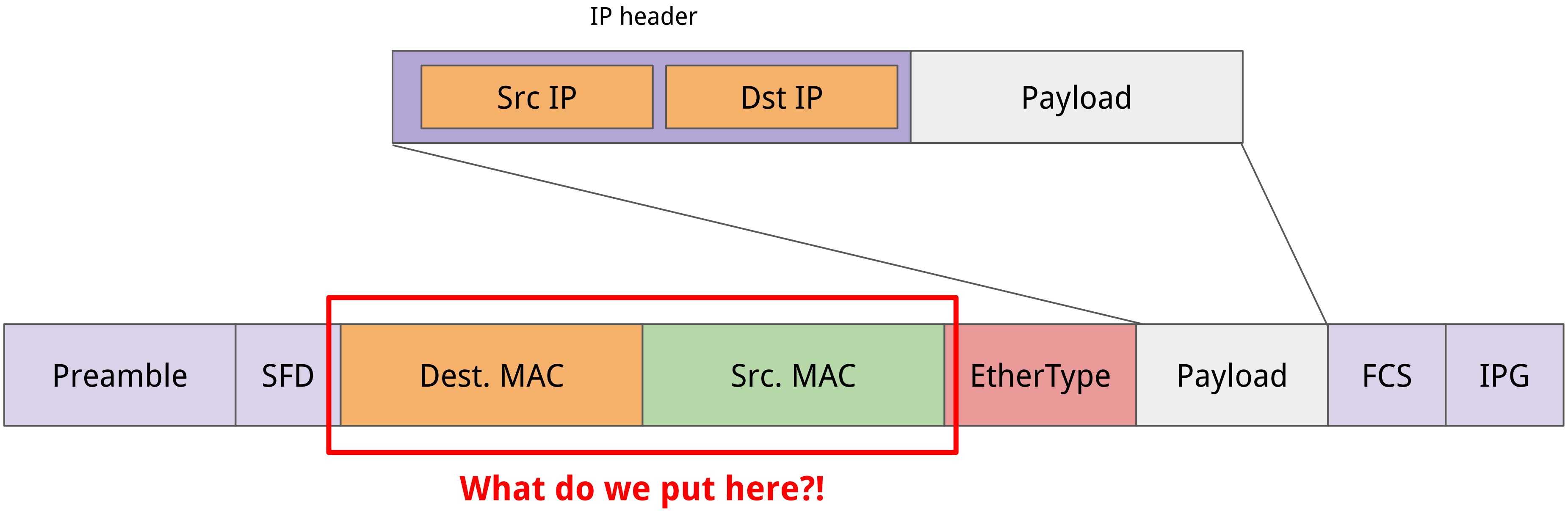
Sending an IP packet to a local host

- Local routing table says that our <u>subnet</u> is local.
  - `192.0.2.0/24` means that anything in `192.0.2.X` is connected to the same network as us.



Sending an IP packet to a local host

- Local routing table says that our <u>subnet</u> is local.
  - `192.0.2.0/24` means that anything in `192.0.2.X` is connected to the same network as us.



Options for Sending L2 Packets

- We could just send our packet to everyone – `FF:FF:FF:FF:FF:FF`.

- Every station connected to the Ethernet network needs to receive and process the frame.
- Any Layer 2 network has to use bandwidth to carry the frame to every host.

- We really want to unicast this frame to the right MAC address that corresponds to the destination IP address.

- **We need some mechanism for us to discover the mapping between IP address and MAC address**.

Overview: Resolving L2 addresses.

- The high-level concept is generally the same across IPv4 and IPv6.
    - Some of the implementation details are a little different.
- High level conversation flow – solicited and advertised.
- Solicitation:
    - “I’m host A, who has IP address 192.0.2.1?”
    - “Hi, I’m host B, I own 192.0.2.1!”
- Advertisement:
    - “I’m Host B, I own 192.0.2.1”
    - (even though no-one asked :-))
- **What do “host A” and “host B” mean?**

Overview: Resolving L2 addresses.

- The high-level concept is generally the same across IPv4 and IPv6.
  - Some of the implementation details are a little different.

- High level conversation flow – solicited and advertisement.

- Solicitation:
  - “I’m **`42:01:0a:8a:00:03`** who has IP address 192.0.2.1?”
  - “Hi, I’m **`44:01:0c:8d:02:05`**, I own 192.0.2.1!”

- Advertisement:
  - “I’m **`44:01:0c:8d:02:05`**, I own 192.0.2.1”
  - (even though no-one asked :-))

- **This provides a link between Layer 2 (MAC) addresses and Layer 3 (IP) addresses.**

ARP: Address Resolution Protocol

- For IPv4, we know an IP address, and we want the corresponding MAC address.

- ARP runs directly on top of L2 (not IP).

- Basic underlying protocol:
  - Request (“who has”)
  - Response (“I am”)

- Requests need to reach everyone within the Ethernet network to find a specific *Target Hardware Address*.
  - Send it to the broadcast address (`FF:FF:FF:FF:FF:FF`).
- Responses go back to the original requester – we can use their MAC address from the request.
- Unsolicited responses are announcements that go to everyone.
  - Can be sent to the broadcast address.

ARP: Examples

We (previously) asked for 192.168.86.1

```
▶ sudo tshark -n arp

Capturing on 'Wi-Fi: en0'

    1   0.000000 3c:28:6d:67:7f:18 → f8:ff:c2:2b:36:16 ARP 42 192.168.86.1 is at 3c:28:6d:67:7f:18

    2   5.643721 68:72:c3:c5:b8:86 → f8:ff:c2:2b:36:16 ARP 42 Who has 192.168.86.38? Tell
192.168.86.51

    3   5.643811 f8:ff:c2:2b:36:16 → 68:72:c3:c5:b8:86 ARP 42 192.168.86.38 is at f8:ff:c2:2b:36:16

    4   6.689294 ec:b5:fa:1d:58:16 → f8:ff:c2:2b:36:16 ARP 60 Who has 192.168.86.38? Tell
192.168.86.47

    5   6.689367 f8:ff:c2:2b:36:16 → ec:b5:fa:1d:58:16 ARP 42 192.168.86.38 is at f8:ff:c2:2b:36:16

5 packets captured
```

ARP: Examples

```
▸ sudo tshark -n arp

Capturing on 'Wi-Fi: en0'

    1   0.000000 3c:28:6d:67:7f:18 → f8:ff:c2:2b:36:16 ARP 42 1  .168.86.1 is at 3c:28:6d:67:7f:18

    2   5.643721 68:72:c3:c5:b8:86 → f8:ff:c2:2b:36:16 ARP 42 Who has 192.168.86.38? Tell
192.168.86.51

    3   5.643811 f8:ff:c2:2b:36:16 → 68:72:c3:c5:b8:86 ARP 42 192.168.86.38 is at f8:ff:c2:2b:36:16

    4   6.689294 ec:b5:fa:1d:58:16 → f8:ff:c2:2b:36:16 ARP 60     has 192.168.86.38? Tell
192.168.86.47

    5   6.689367 f8:ff:c2:2b:36:16 →                             2.168.86.38 is at f8:ff:c2:2b:36:16

5 packets captured
```

Someone is asking us for our address.

So we respond.

ARP: How do we know what to ARP for?

- We need to ARP for **local** IP addresses.

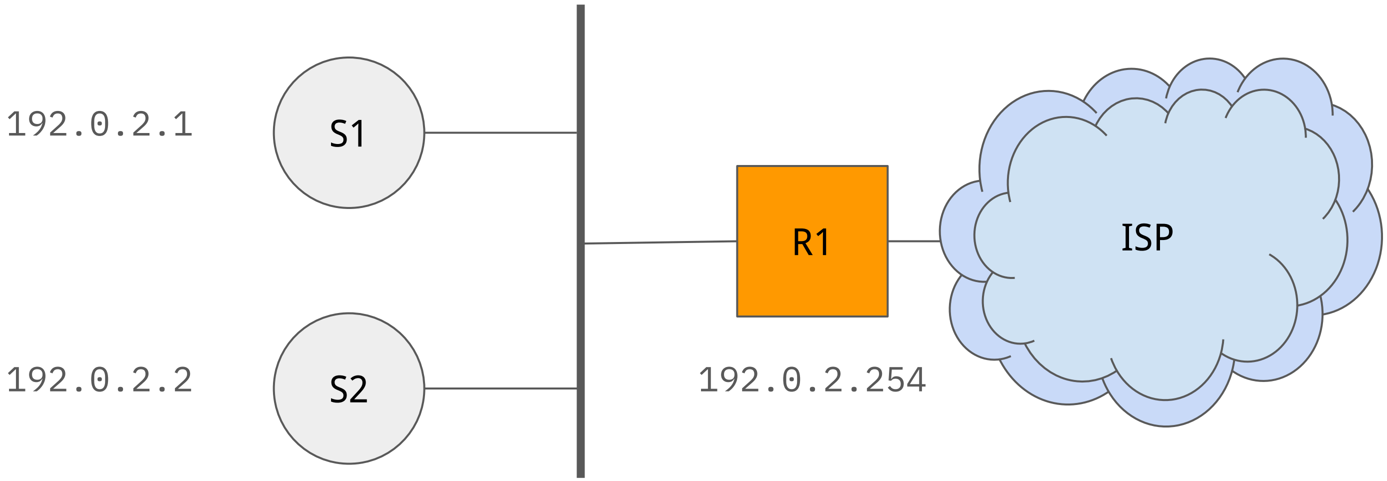
```
▸ netstat -rn -f inet
Routing tables
Internet:
Destination        Gateway            Flags            Netif Expire
default            192.168.86.1       UGScg              en0
192.168.86         link#6             UCS                en0       !
```

We referred to these kind of entries as "direct".

Questions?

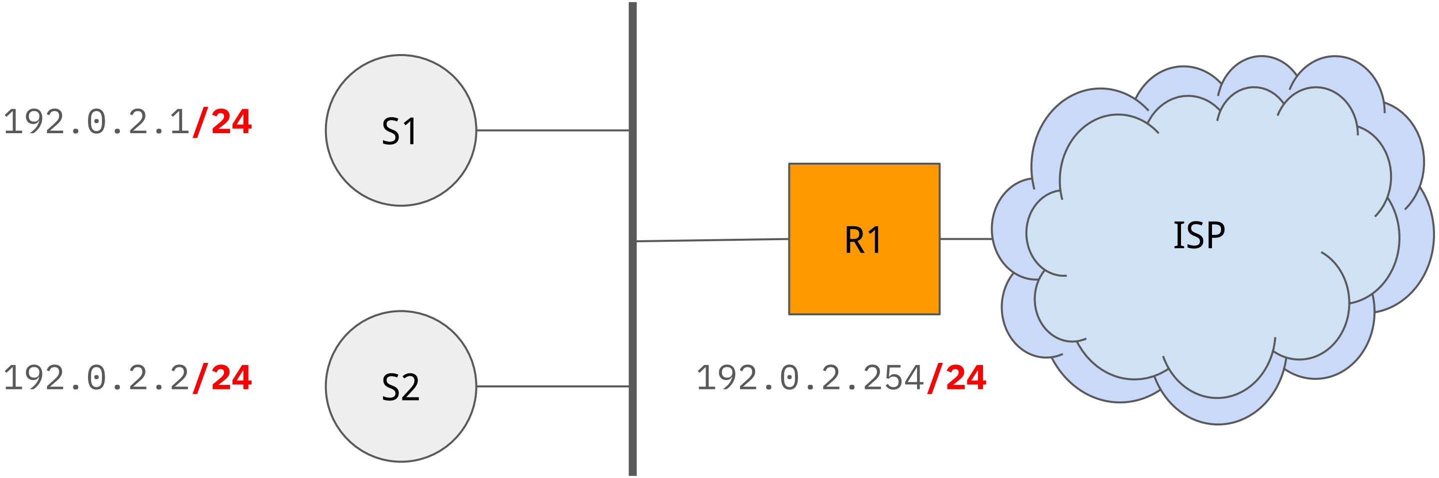
Note: Local IP addresses & Routing

- What is local to a particular host?
    - A **range** of IP addresses – i.e., an IP prefix.



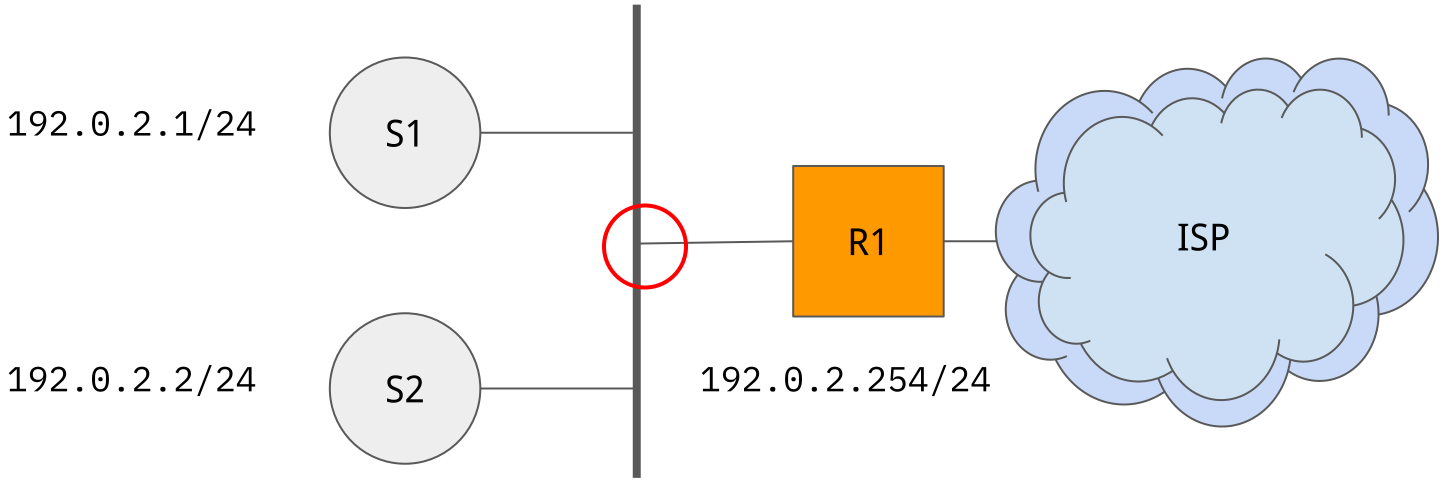
Note: Local IP addresses & Routing

- What is local to a particular host?
  - A **range** of IP addresses – i.e., an IP prefix.
  - **192.0.2.0/24 → Direct**



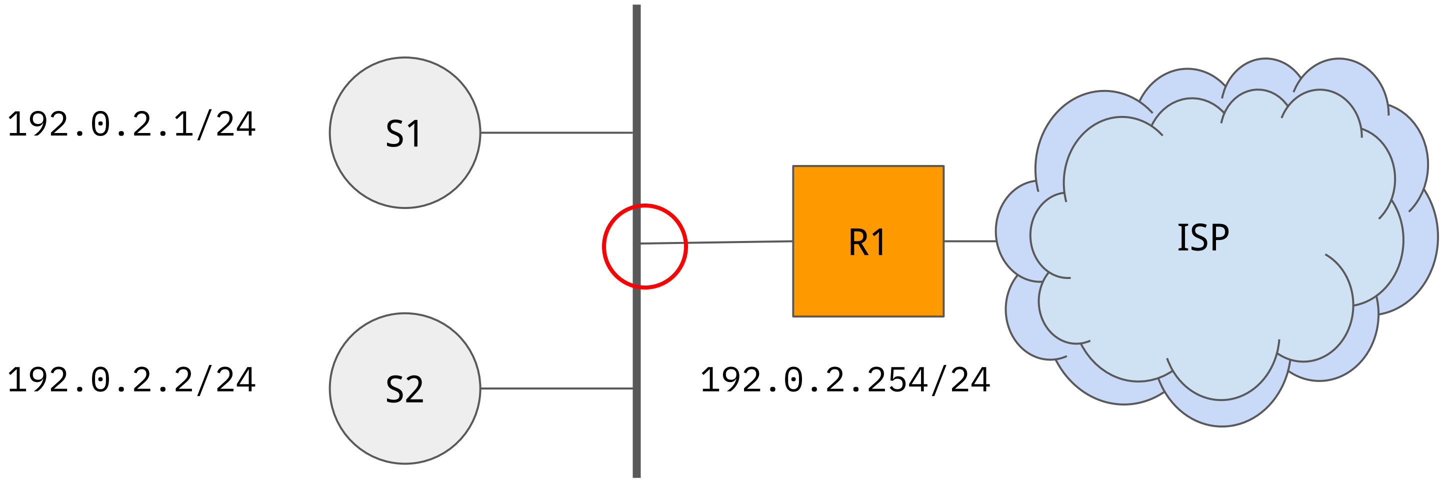
Aside: What did `Direct` actually mean?

- Something that we don't have a next hop for.
  - We've said it just means "send to port 1".
  - What happens if there are multiple hosts on the same port?



Aside: What did `Direct` actually mean?

- Something that we don't have a next hop for.
  - We've said it just means "send to port 1".
  - What happens if there are multiple hosts on the same port?
  - Direct actually meant "just send to the right Layer 2 address for the destination IP address"



What do we ARP for?

- Things that are directly connected to us in our routing table.

- Convert our address to binary, and consider our netmask.
    - Netmask expressed as CIDR (/24) or as a dotted quad – `255.255.255.0`.

- This allows us then to form an ARP `who-has` request towards the Ethernet broadcast address.
    - The `FF:FF:FF:FF:FF:FF` broadcast address is independent of the Ethernet network we are on!

- Subsequently, we can update our local **ARP Table** to store the mapping between the destination IPv4 address and MAC address.
    - And just send unicast Ethernet frames to that destination MAC address.

The ARP Table

- Stores mappings between IPv4 addresses and Ethernet MAC addresses.
- Format:
  - `IPv4 address`, `MAC address`, `Interface`, `Expiry`
- We need to know the address on a specific interface that we have an IP address assignment on.
- Must be an expiry time (time to live) since IP addresses might change owners.
  - A new machine could connect to the network and use the same IPv4 address.

The ARP Table

MAC address associated with the address

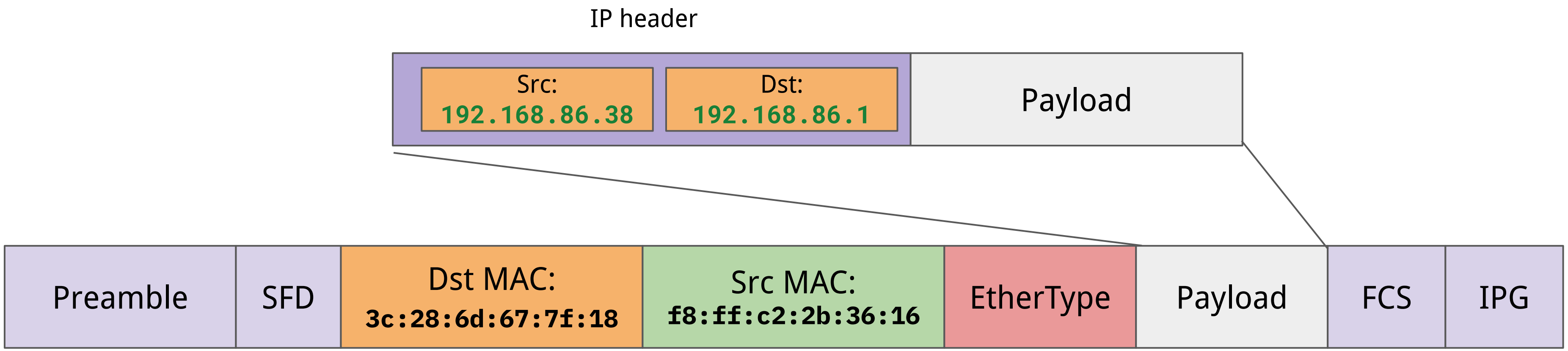
```
▶ arp -n -a
? (192.168.86.1) at 3c:28:6d:67:7f:18 on en0 ifscope [ethernet]
? (192.168.86.21) at f4:f5:d8:2f:8c:a8 on en0 ifscope [ethernet]
? (192.168.86.23) at c0:95:6d:7e:69:bd on en0 ifscope [ethernet]
? (192.168.86.29) at 3c:28:6d:67:f9:c5 on en0 ifscope [ethernet]
```

Remote IPv4 address

Interface name

Sending an IP packet

- We are 192.168.86.38 sending to 192.168.86.1.
  - Look at routing table 192.168.86.0/24 is direct.
  - Look at ARP table – retrieve destination MAC address of 3c:28:6d:67:7f:18.



Layer 2 and Layer 3 destinations are different.

- Our routing table might say we have a *default route*.
  - `0.0.0.0/0` is via `192.0.2.1`
  - We are connected to `192.0.2.0/24`.
- We want to send a packet to `10.0.0.1`.
  - This isn’t local!
- Our routing table says send to `192.0.2.1`.
  - We build an Ethernet frame that has a destination MAC address corresponding to the MAC for `192.0.2.1`.
  - The IP header (L3) is still `10.0.0.1`.
- Each hop that forwards changes the L2 destination but not the L3 destination.

IPv6: Neighbour Discovery

- IPv6 uses a similar mechanism to ARP: Neighbour Discovery.
  - Uses ICMPv6 towards **well-defined multicast addresses**.
  - These multicast addresses are programmatically associated with multicast Ethernet MAC addresses.

- Neighbour Solicitation allows a node to ask for the association between an IPv6 address and MAC address.
  - Uses the IPv6 address to determine the Ethernet multicast MAC.
  - Local IPv6 address is used by a node to determine which multicast groups to join.

- Neighbour Advertisement allows a node to reply with the association between IPv6 and MAC address.

Questions?

Learning about our local network.

What does a host need to know?

- If we connect to a new Ethernet network – we know our own MAC address (burnt into our hardware).

- We need to know the IP address we should use for this network.
- **And** the subnet mask (network size) that tells us which other addresses are directly connected to us.

- We need to know where to send packets for IP addresses that are **not** directly connected to us.

- We might need to know where the resolving DNS server for this network is.

How could we learn this additional information?

- Option 1: **Manually**.

- When a machine connects to a new network:
  - Your address is `192.168.86.38`.
  - The network size is `/24`.
  - Your *default gateway* is `192.168.86.254`.
  - Your resolving DNS servers are `8.8.8.8` and `1.1.1.1`.

- Humans go and configure these details.

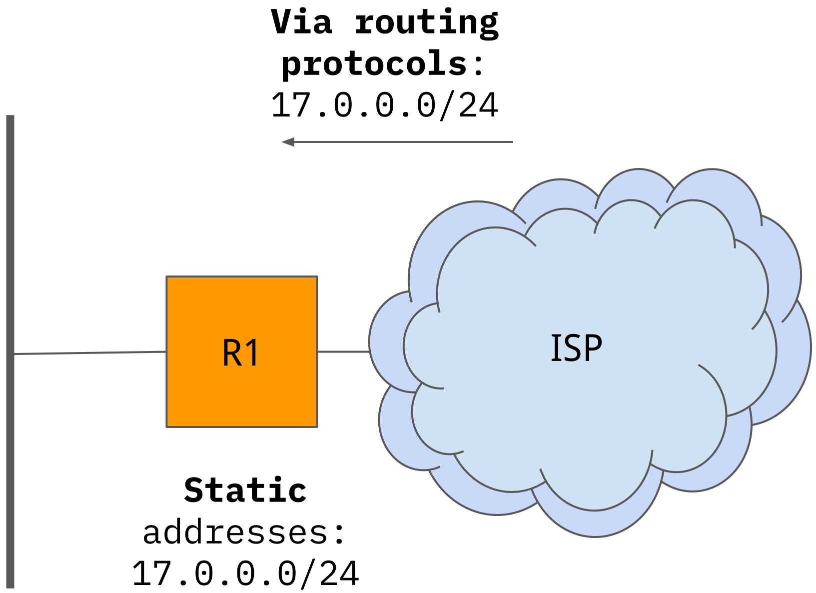
- What happens when we move location?
  - Addresses were hierarchically allocated based on the network.
  - Your home network has different addresses and details to the Berkeley network.

How could we learn this additional information?

- Option 2: **Automatically**.

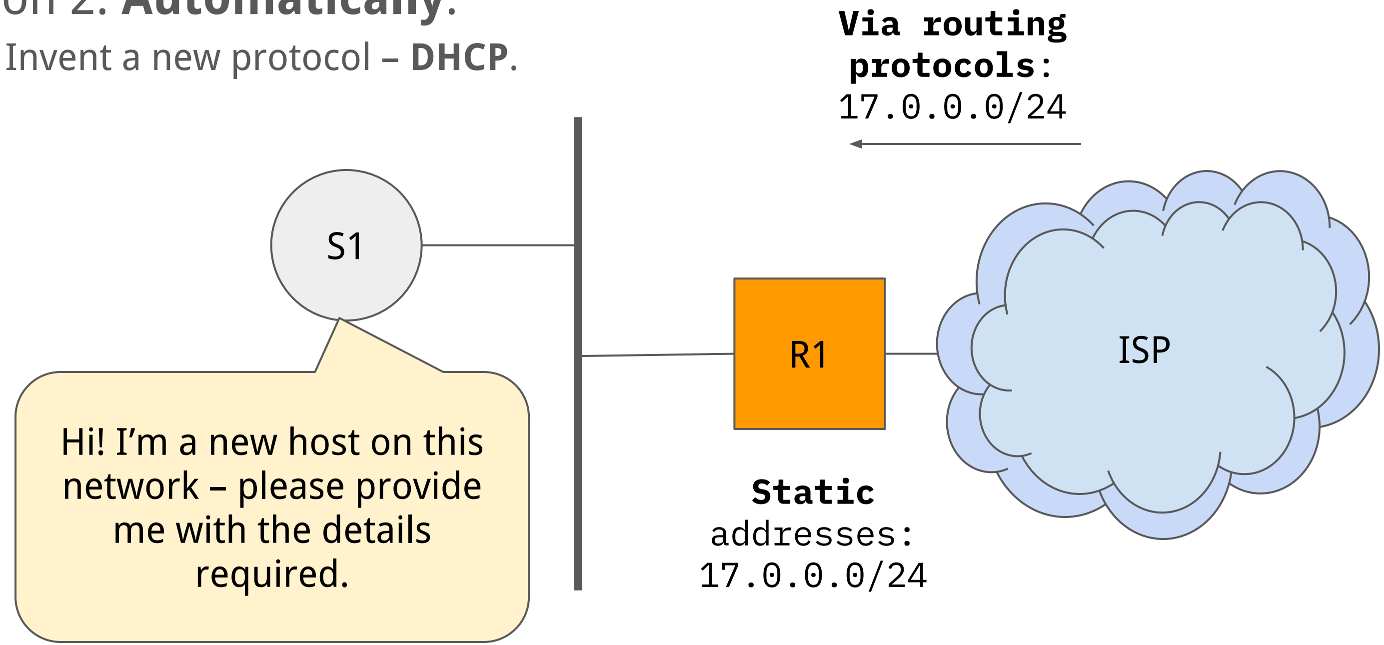
Routers are static – based on their location in the network.

Manually configuring addresses is acceptable – but hosts move more often.



How could we learn this additional information?

- Option 2: **Automatically**.
  - Invent a new protocol – **DHCP**.



DHCP

- DHCP is the *Dynamic Host Configuration Protocol*.

- Provides a way for a new host to query information from “the network” about the local environment that it is in.

- DHCP carries:
  - IP address assigned to the host
  - Netmask
  - “Default gateway” – the first-hop (directly connected router) that non-local packets can be sent.
    - This is where `0.0.0.0/0` is sent towards.
  - Additional information:
    - Local DNS resolving server.
  - Extensible to carry other information.

DHCP: Servers

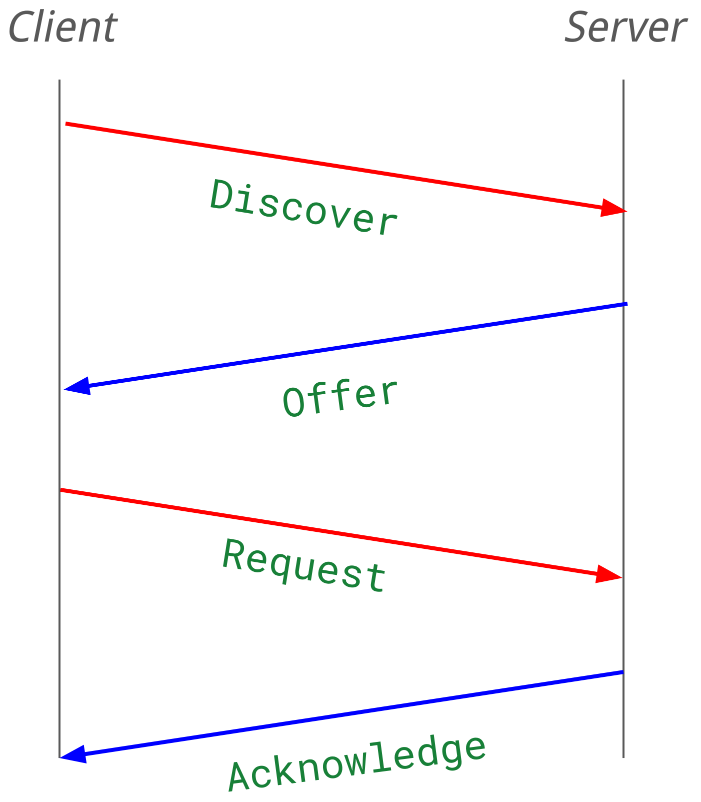
- DHCP servers are added to the network.
  - Either on the local router or a separate machine that acts as the server.
- These servers listen on UDP port 67.
- DHCP servers are configured with required information:
  - First hop router address, local DNS servers.
  - A **pool** of usable IP addresses.

DHCP: IP Assignments

- Servers *lease* hosts IP addresses.
- Leases are only valid for a limited time (on the order of hours or days).
  - Hosts must renew the lease if they want to keep the address.
- Servers don't offer the same address to other clients if leased.
  - Avoids conflicts of IP addresses.
- No need for any static configuration – just connected to the network and ask for the local details!

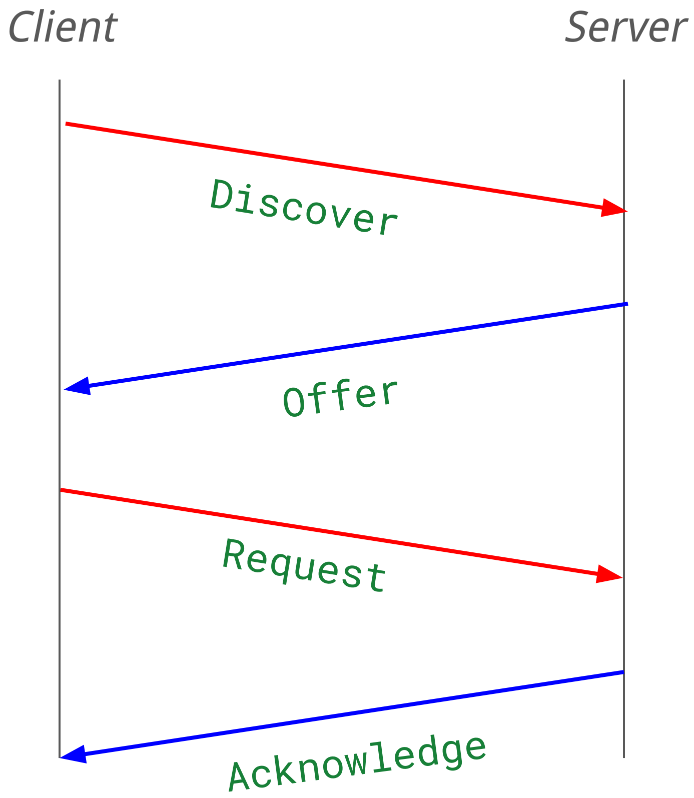
Questions?

DHCP: Message Flow



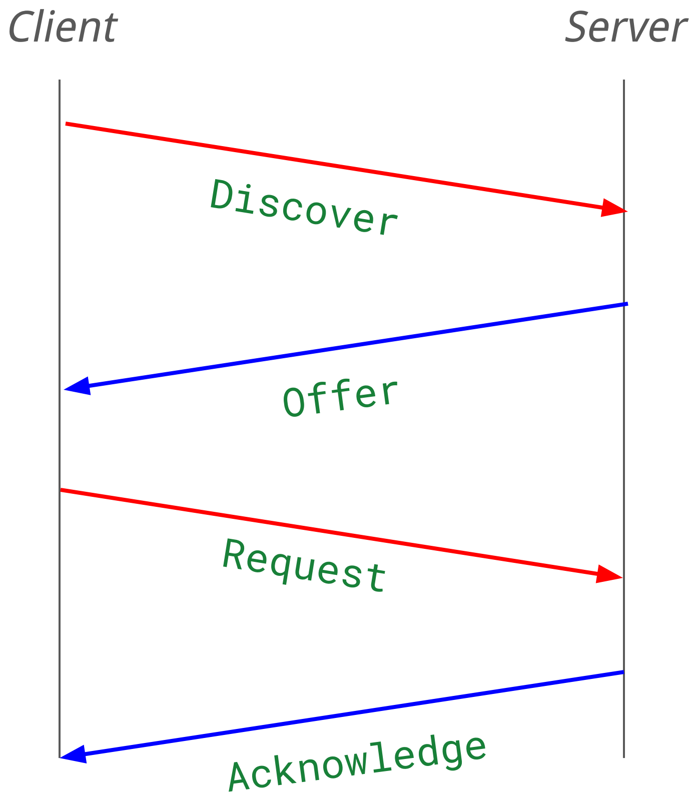
- Client sends a `discover` message – asks for configuration information from the local DHCP server.

DHCP: Message Flow



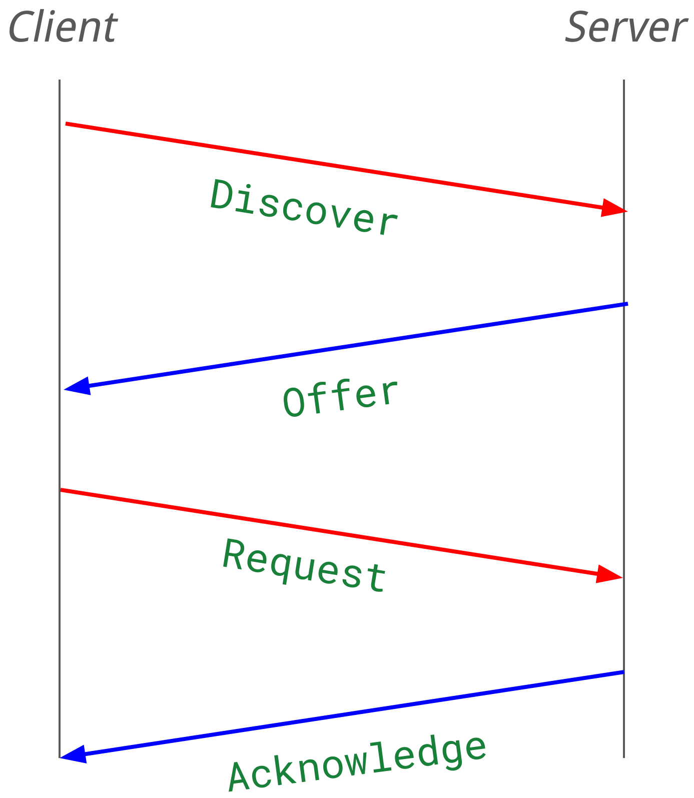
- Client sends a `discover` message – asks for configuration information from the local DHCP server.
- Local DHCP server(s) send `offer` messages with configuration information (e.g., particular IP).

DHCP: Message Flow



- Client sends a `discover` message – asks for configuration information from the local DHCP server.
- Local DHCP server(s) send `offer` messages with configuration information (e.g., particular IP).
- Client sends a `request` message to accept a particular offer.

DHCP: Message Flow



- Client sends a `discover` message – asks for configuration information from the local DHCP server.
- Local DHCP server(s) send `offer` messages with configuration information (e.g., particular IP).
- Client sends a `request` message to accept a particular offer.
- Server sends an `acknowledge` message to confirm the request was granted.

DHCP

- DHCP is based on UDP – which runs on top of IP.
- How does the client know the server IP?
    - It doesn't!
- It sends a message to an IPv4 broadcast address - `255.255.255.255`.
    - This maps to an Ethernet broadcast address - `FF:FF:FF:FF:FF:FF`.
- Thus, the client does not need to understand anything about the local network.
- Wait, what source IP address does it use?
    - `0.0.0.0`.
- What destination MAC address that the server sends to?
    - Could be broadcast – `FF:FF:FF:FF:FF:FF` is received by all hosts on the network.
    - Could be the MAC address that the DHCP request came from.

Where does the DHCP server live?

- Must be within the same Ethernet network – so broadcast frames reach it.

- Running on the local router – especially in home networks – is therefore very convenient.

- In larger networks, we might not want a DHCP server at every router.
  - Therefore we can **relay** requests from one router to a remote DHCP server.

Questions?

IPv6: Autoconfiguration

- DHCP also exists for IPv6 networks.
- But, remember, IPv6 neighbour discovery used IPv6 multicast addresses.
- MAC addresses are 48-bits, and IPv6 addresses are 128-bits.
- Thus, we can encode a MAC address and put it into an IPv6 address.
  - `fe80::/10` is assigned for “link local” IPv6 addresses.
  - We can configure an address in `fe80::/64` using our local MAC address.
  - Encoded using “Extended Unique Identifier” – EUI – as a 64-bit value.
- IPv6 nodes generate an `fe80::<64bit unique ID>` for each interface.
  - `fe80::/64` addresses are **scoped** per link - `fe80::1%interface`, since the same address can exist on multiple links.

IPv6: Extending Neighbour Discovery Protocol

- We can therefore get a <u>local</u> IPv6 address with no additional protocol in IPv6.

- We still need to have ways to discover the local router and DNS servers.

- This is done through additional messages in the Neighbour Discovery Protocol.

- `Router Solicitation` and `Router Advertisement` allow information to be sent from a router to hosts.

IPv6: StateLess Address Auto Configuration (SLAAC)

- Hosts *solicit* for routers on the local link – using a new message in IPv6 NDP.
- Routers *advertise* information about the local network to hosts.
- The local IPv6 network has a fixed prefix length of `/64`.
  - The router can just advertise the prefix that is being used – `2001:db8::/64`.
  - Hosts use EUI-64 to configure their own address – does not need to be allocated (since MAC addresses are ~unique).
  - Additional mechanisms are needed for duplicate address detection (just in case!)
- Router Advertisement messages allow the default gateway and DNS servers to be communicated between a router and host.
- <u>Most</u> IPv6 networks do not run DHCPv6.

Questions?

Recap

- Ethernet is a Layer 2 networking technology that provides connectivity between *local* nodes (routers and hosts).
- Ethernet addresses allow for packets to be sent between machines before they have IP addresses (or even when they don’t have IP addresses).
- ARP and Neighbour Discovery allow machines to map IP addresses to MAC addresses.
- DHCP provides a mechanism for dynamic configuration of Layer 3 information on hosts.