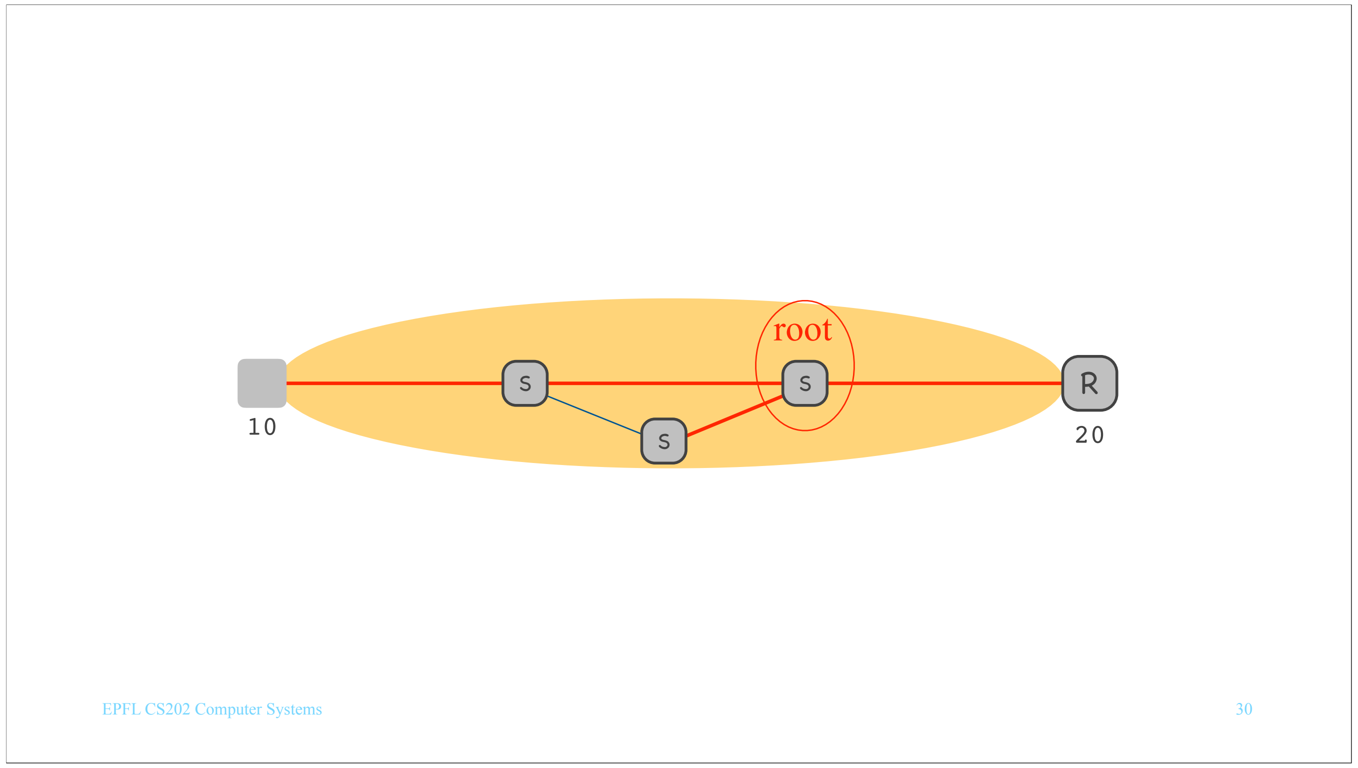
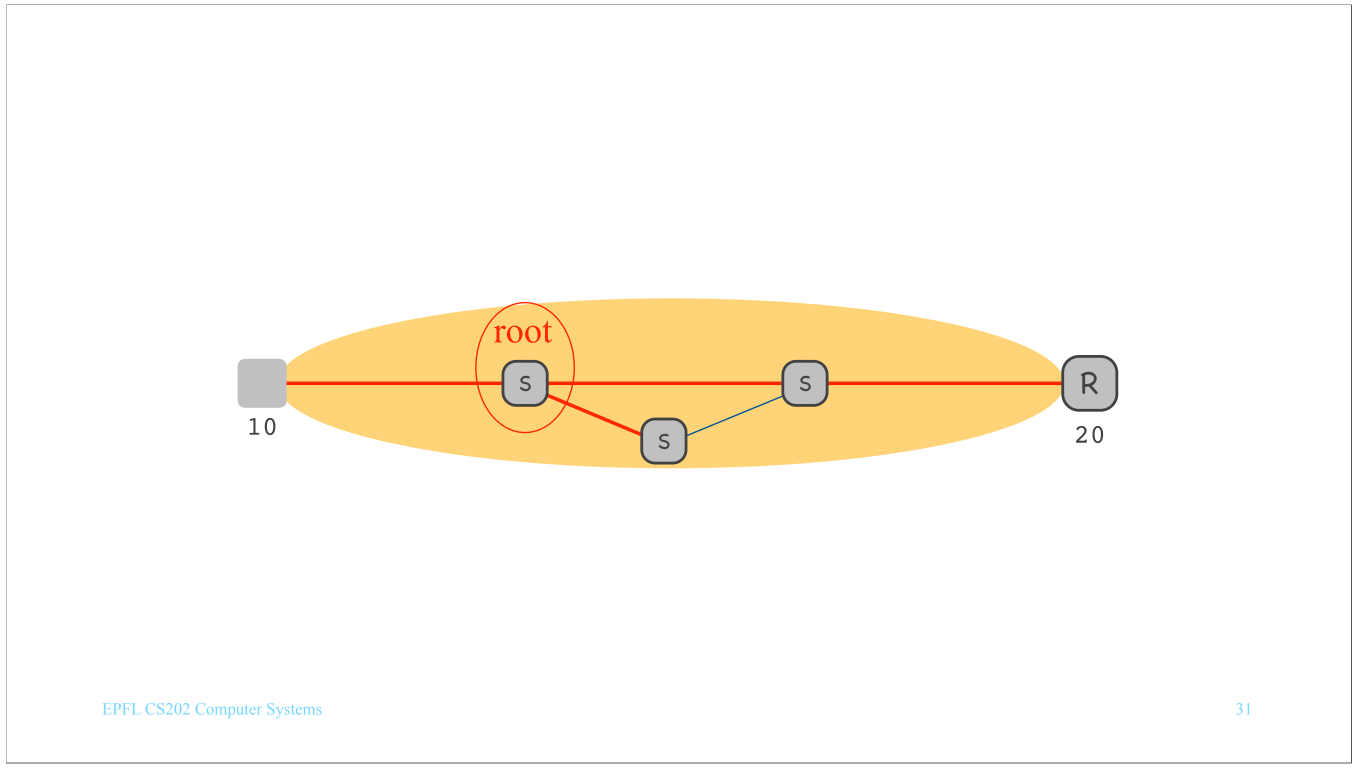


To avoid this, all the switches in an IP subnet participate in a protocol that creates a “spanning tree.”

This is a subgraph of the IP subnet, which includes all the end-systems, routers, and switches of the subnet, but only a subset of the links. It includes just enough links to reach all the end-systems and network devices.

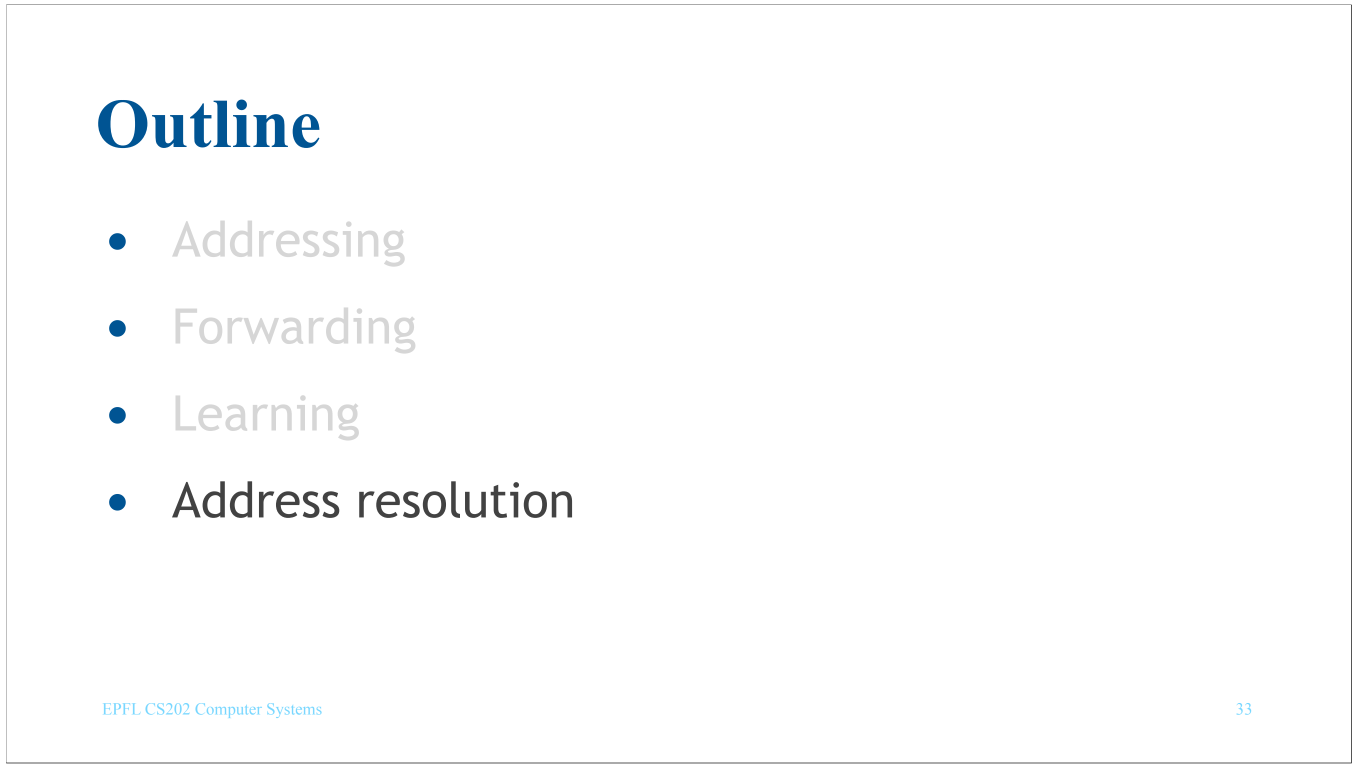
There may exist many spanning trees in an IP subnet.



Here is another example.

Spanning tree

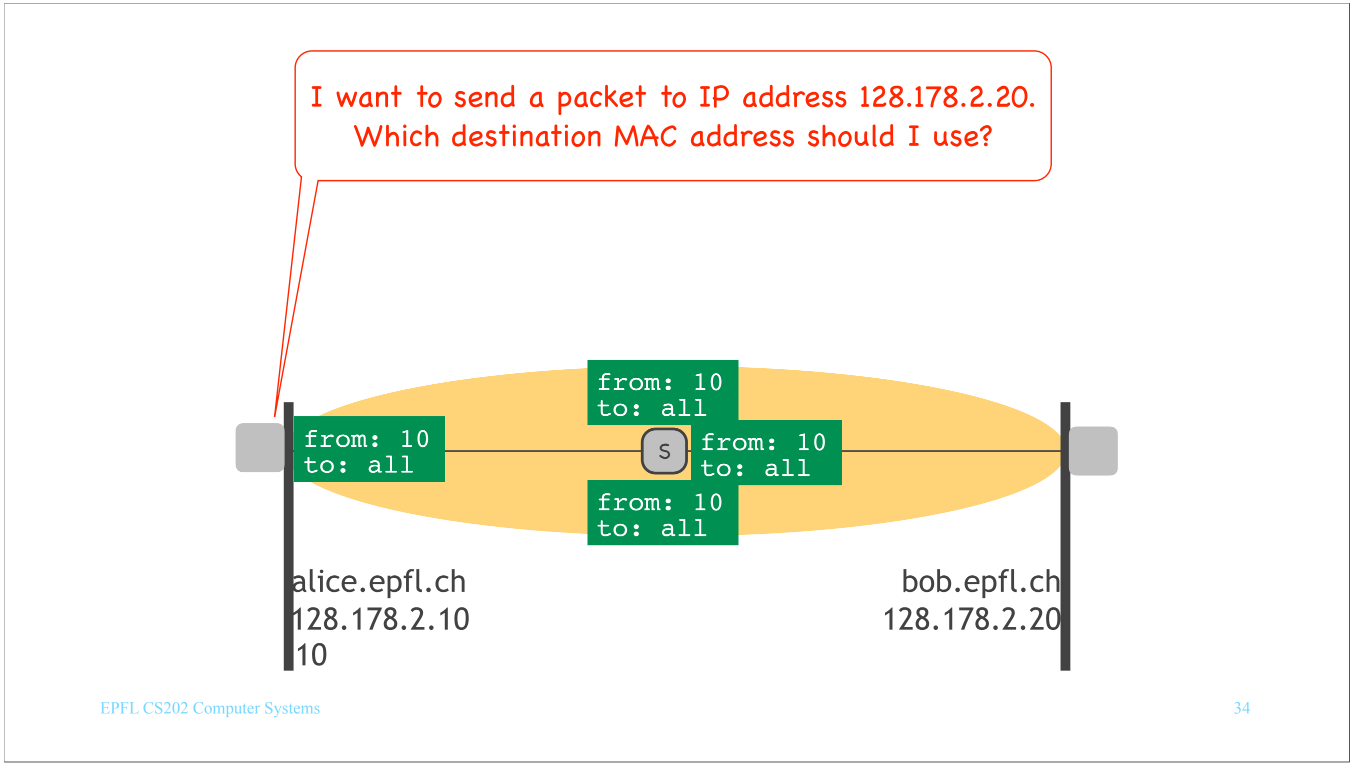
- A subgraph with special properties
  - *includes all nodes + some edges*
  - *cannot remove an edge without disconnecting a node*
- Useful for loop-free broadcasting
  - *broadcast traffic propagated only along tree*
  - *prevents forwarding loops*



Outline

- Addressing
- Forwarding
- Learning
- Address resolution

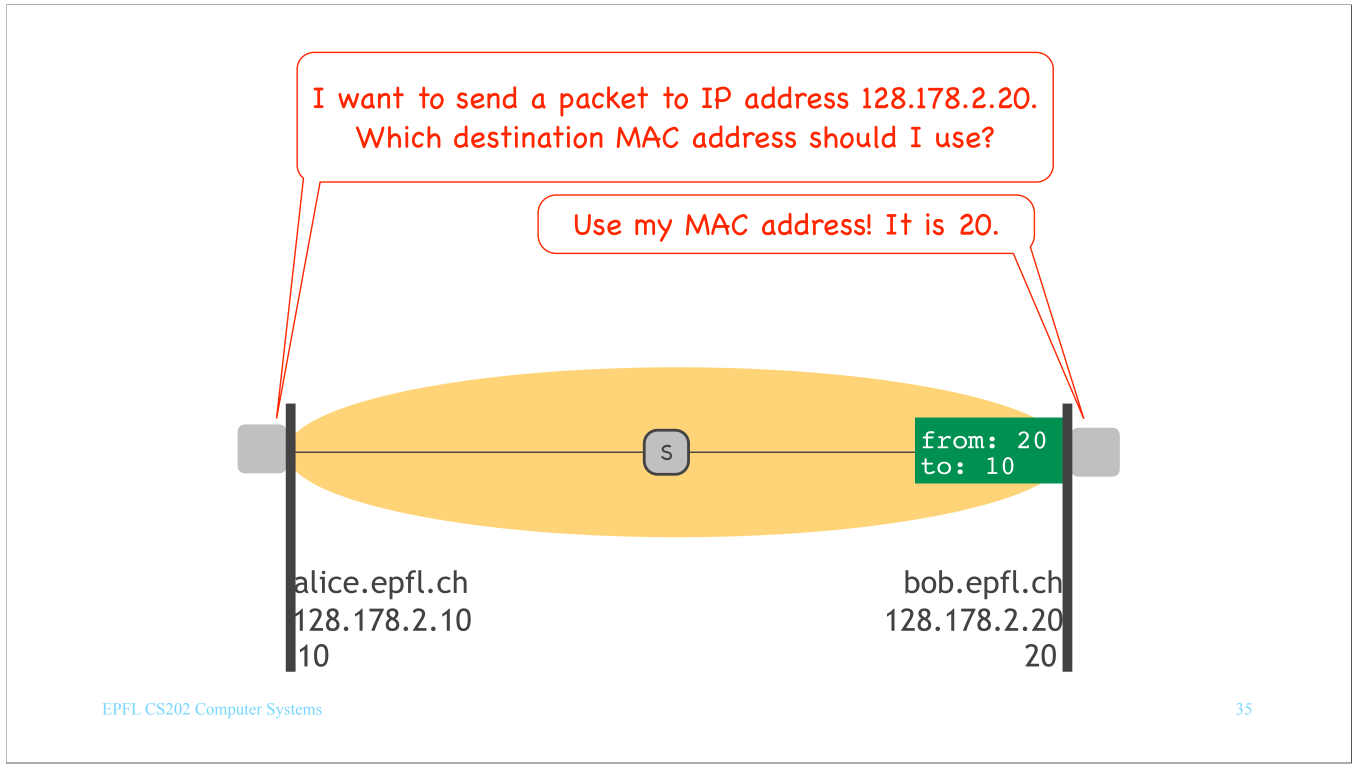
One last but crucial question remains: in an Ethernet, how does an end-system learn the destination MAC address that it should write on a packet that it wants to send?


I want to send a packet to IP address 128.178.2.20.
Which destination MAC address should I use?
from: 10
to: all
from: 10
to: all
S
from: 10
to: all
from: 10
to: all
alice.epfl.ch
128.178.2.10
10
bob.epfl.ch
128.178.2.20
EPFL CS202 Computer Systems
34

Suppose Alice wants to send a packet to Bob, who happens to be in the same IP subnet as Alice.

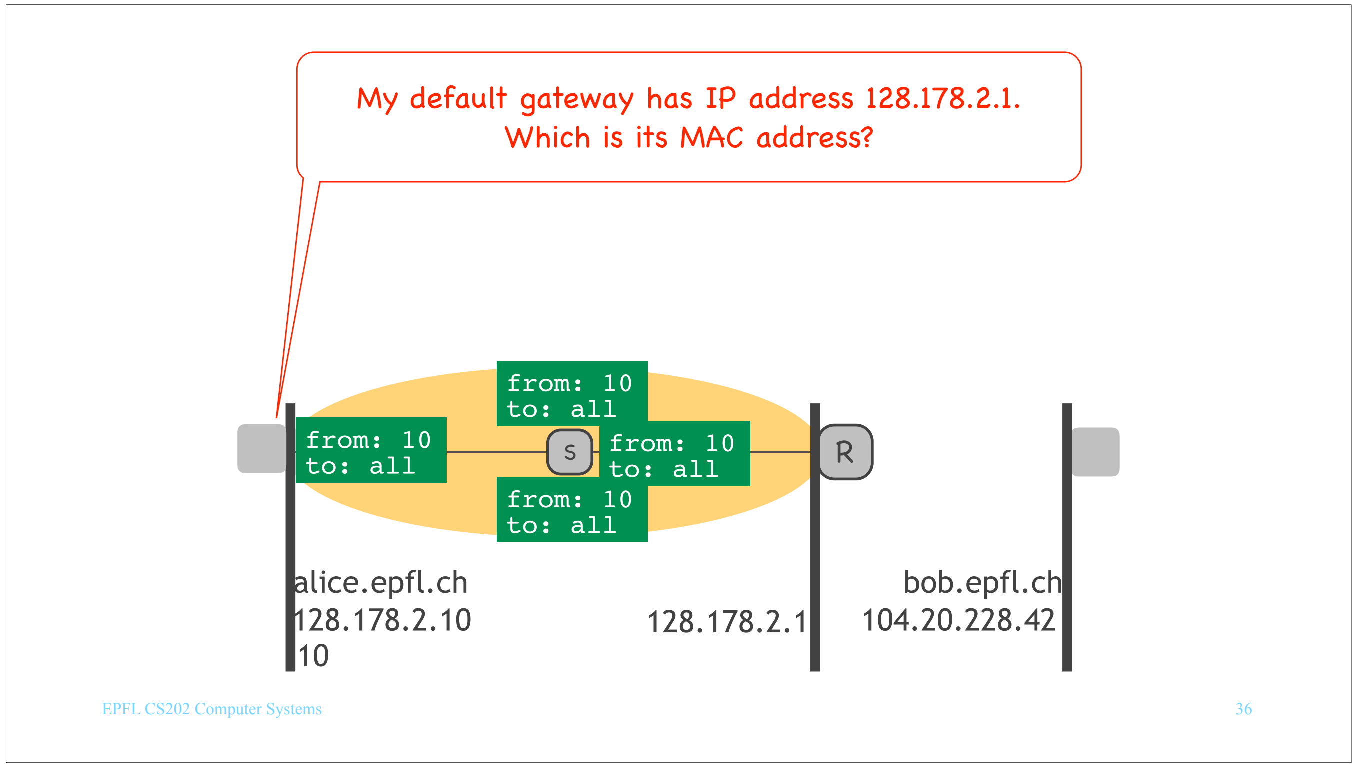
The first thing she needs is Bob's IP address, which will be the packet's destination IP address.
We know how she gets this: she asks a local DNS server.

The second thing she needs is Bob's MAC address, which will be the packet's destination MAC address.
To get this, she sends out a special request that states Bob's IP address.
This request (called an "ARP request," as we will see later) has a special destination MAC address, which is called a "broadcast address," meaning that the request is addressed to all the end-systems and routers in the local IP subnet.
So, every switch that receives this request broadcasts it.
Eventually, the request reaches Bob.


I want to send a packet to IP address 128.178.2.20. Which destination MAC address should I use?
Use my MAC address! It is 20.
S
from: 20
to: 10
alice.epfl.ch
128.178.2.10
10
bob.epfl.ch
128.178.2.20
20
EPFL CS202 Computer Systems
35

When Bob receives the request, he recognises his IP address and sends a special response (called an “ARP response,” as we will see later), which states his MAC address.

When Alice receives Bob’s response, she learns his MAC address and is finally ready to send him a packet
(because she now knows both what destination IP address and what destination MAC address to write on the packet).



Now suppose Alice wants to send a packet to Bob, who happens to be in a different IP subnet from Alice.

As in the previous scenario, the first thing she needs is Bob's IP address, which will be the packet's destination IP address.
We know how she gets this: she asks a local DNS server.

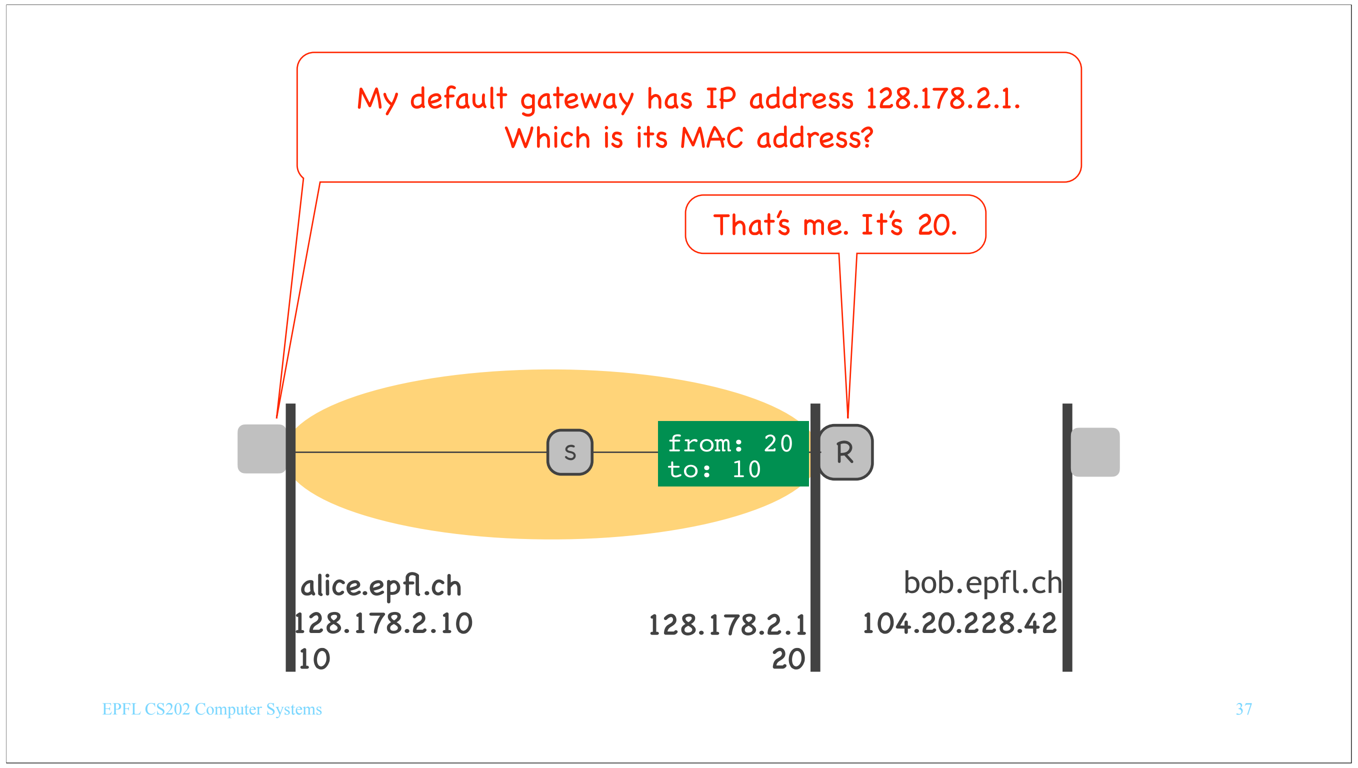
Again, as in the previous scenario, the second thing Alice needs is the correct destination MAC address to write on her packet.
That's the MAC address of router R's network interface that belongs to the local IP subnet.

There are two ways to get this MAC address: **Default gateway** and **Proxy ARP**. We will look only at the first one —which is the most common today — but you know all that is needed to lookup the second one if you are interested.

So, here is how default gateway works:

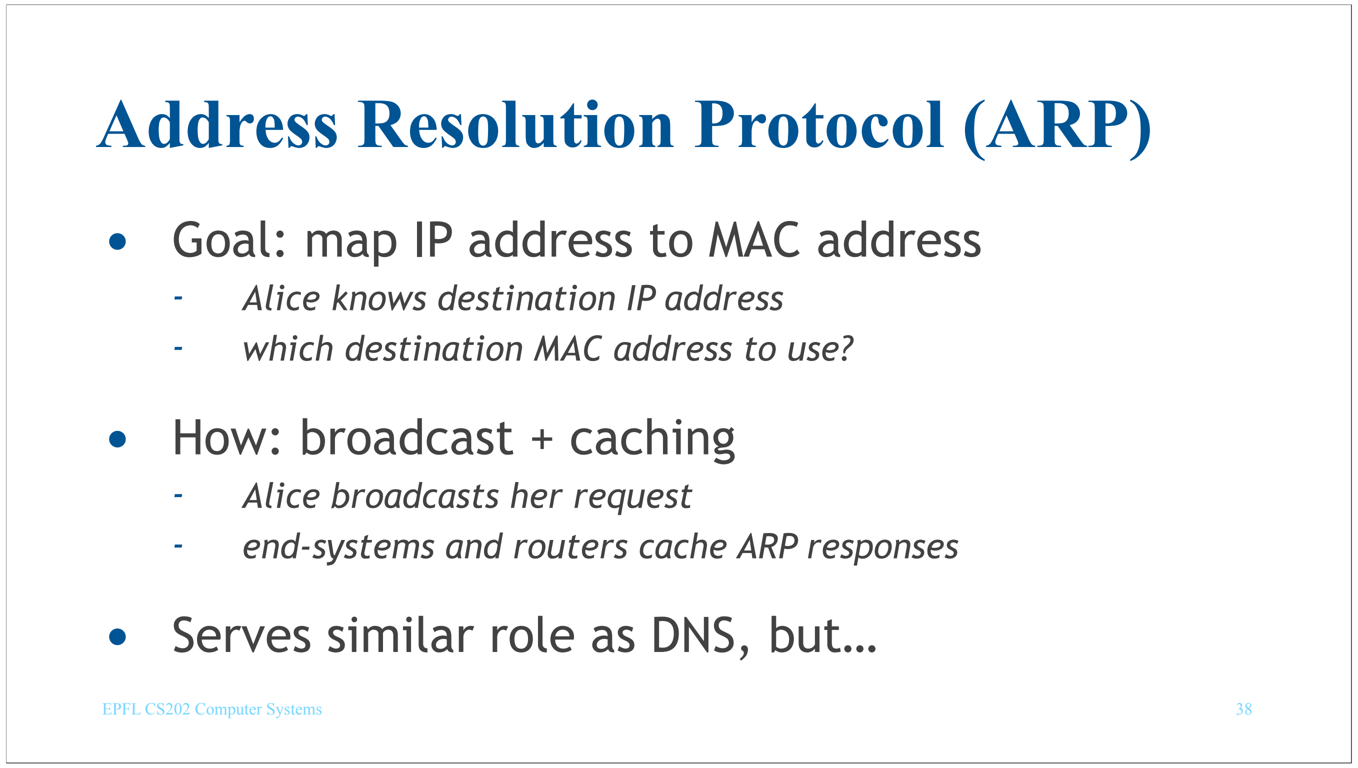
By configuration, Alice knows that R is her "default gateway," meaning that any traffic that Alice sends outside the local IP subnet will exit the subnet through R. More precisely, Alice knows R's IP address.

Hence, Alice broadcasts an ARP request that states R's IP address.
Eventually, the ARP request reaches R.



When R receives the ARP request, it recognizes its IP address and sends an ARP response that states its MAC address.

When Alice receives R's ARP response, she learns R's MAC address,
and is finally ready to send her packet to Bob.



Address Resolution Protocol (ARP)

- Goal: map IP address to MAC address
  - *Alice knows destination IP address*
  - *which destination MAC address to use?*
- How: broadcast + caching
  - *Alice broadcasts her request*
  - *end-systems and routers cache ARP responses*
- Serves similar role as DNS, but…

ARP requests and responses are part of the Address Resolution Protocol (ARP)…

Broadcasting

- Alice sends request to special, broadcast destination MAC address
  - *FF-FF-FF-FF-FF-FF*
- Reaches every end-system and router in the local IP subnet

ARP vs. DNS

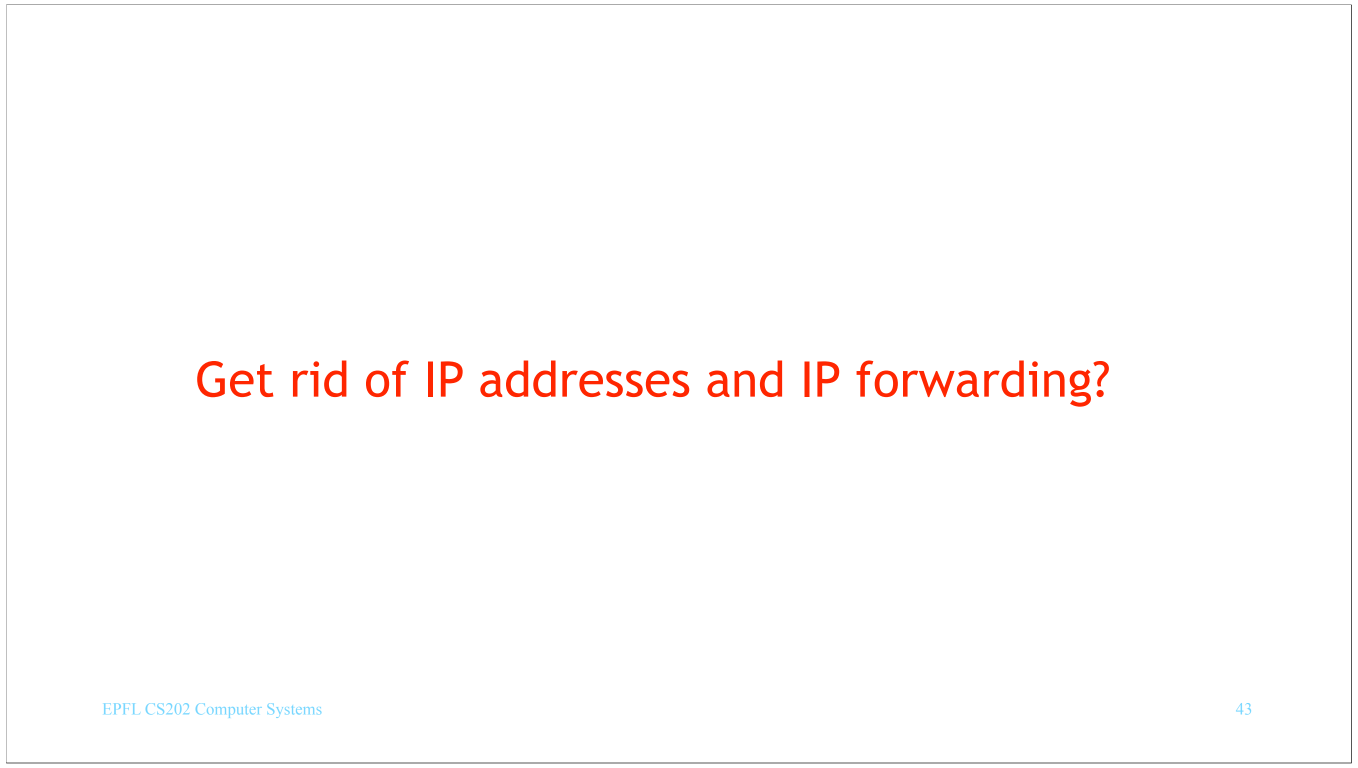
- ARP: relies on broadcasting
  - *no logically centralized map*
  - *each entity knows its own MAC address and knows which requests to respond to*
- DNS: relies on DNS infrastructure
  - *logically centralized map*
  - *stored in DNS servers*

Basic Ethernet elements

- Address Resolution Protocol
  - *resolves IP address to MAC address*
- L2 forwarding
  - *based on MAC addresses (which are flat)*
- L2 learning
  - *populates switch forwarding table*

Basic Ethernet elements

- Address Resolution Protocol [rel. to DNS]
  - *resolves IP address to MAC address*
- L2 forwarding [rel. to IP forwarding]
  - *based on MAC addresses (which are flat)*
- L2 learning [rel. to IP routing]
  - *populates switch forwarding table*

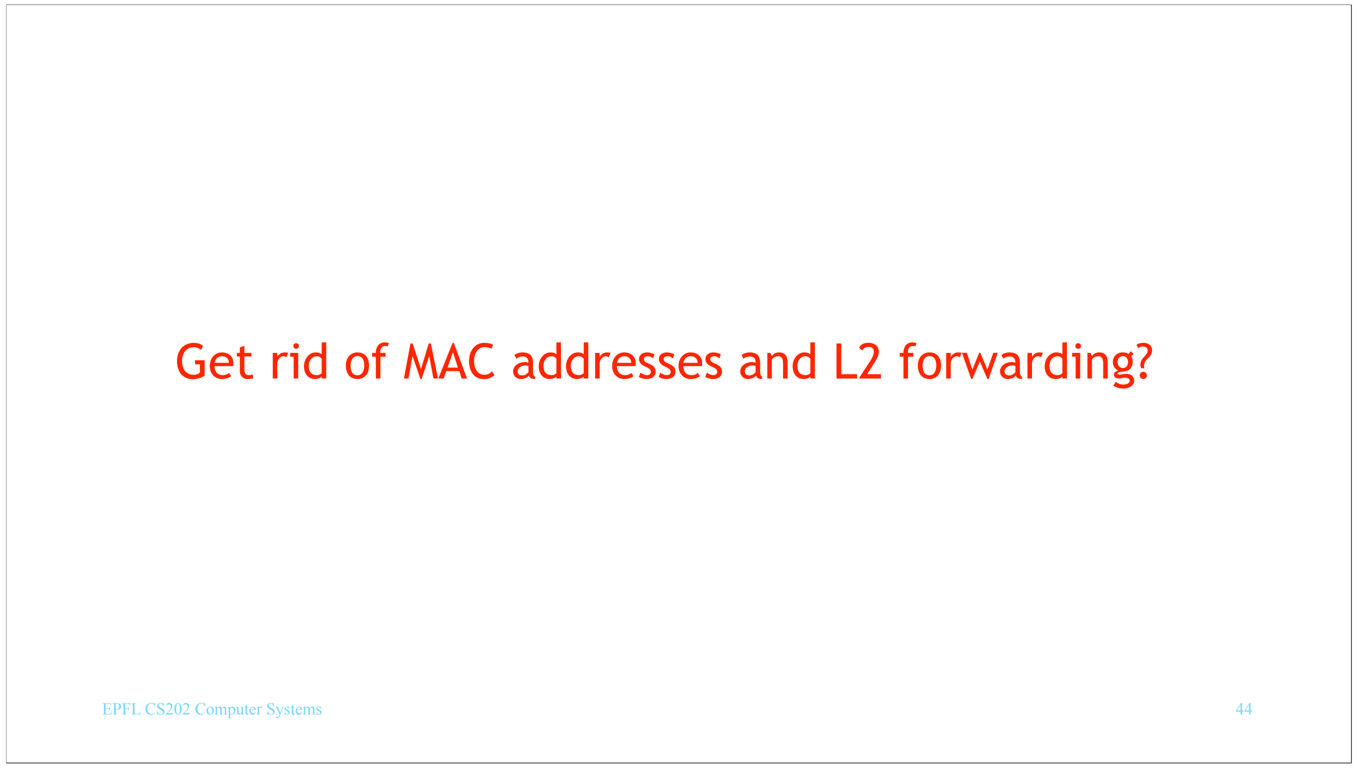


Get rid of IP addresses and IP forwarding?

Could we get rid of IP addresses and IP routers/forwarding? Could all Internet end-systems have only MAC addresses and be interconnected through link-layer switches? In other words, could the Internet be one big IP subnet?

No, this would not scale:

- A network device located in the middle of the Internet may see traffic from millions of end-systems. The forwarding table would need to be large enough to fit individual entries for each active end-system.
- Imagine what would happen if every DNS request was broadcast to everyone on the Internet in the hope that the right entity will eventually answer.



Get rid of MAC addresses and L2 forwarding?

Could all Internet end-systems have only IP addresses and be interconnected through IP routers?

Yes, that would scale.

But, it would remove flexibility.

The Internet was fundamentally designed as a network of networks (IP subnets), not a single network. In this lecture, we saw only one type of IP subnet (Ethernet), but there are many others, and we want to be able to interconnect all of them. If we only used IP addresses and IP routers, then we would need to replace all the network devices in all these different types of IP subnets with IP routers.