CSCI 466: Networks

IP Addresses, IP Protocol, Subnets, NAT

Reese Pearsall

Fall 2024

*All images are stolen from the internet

Announcements

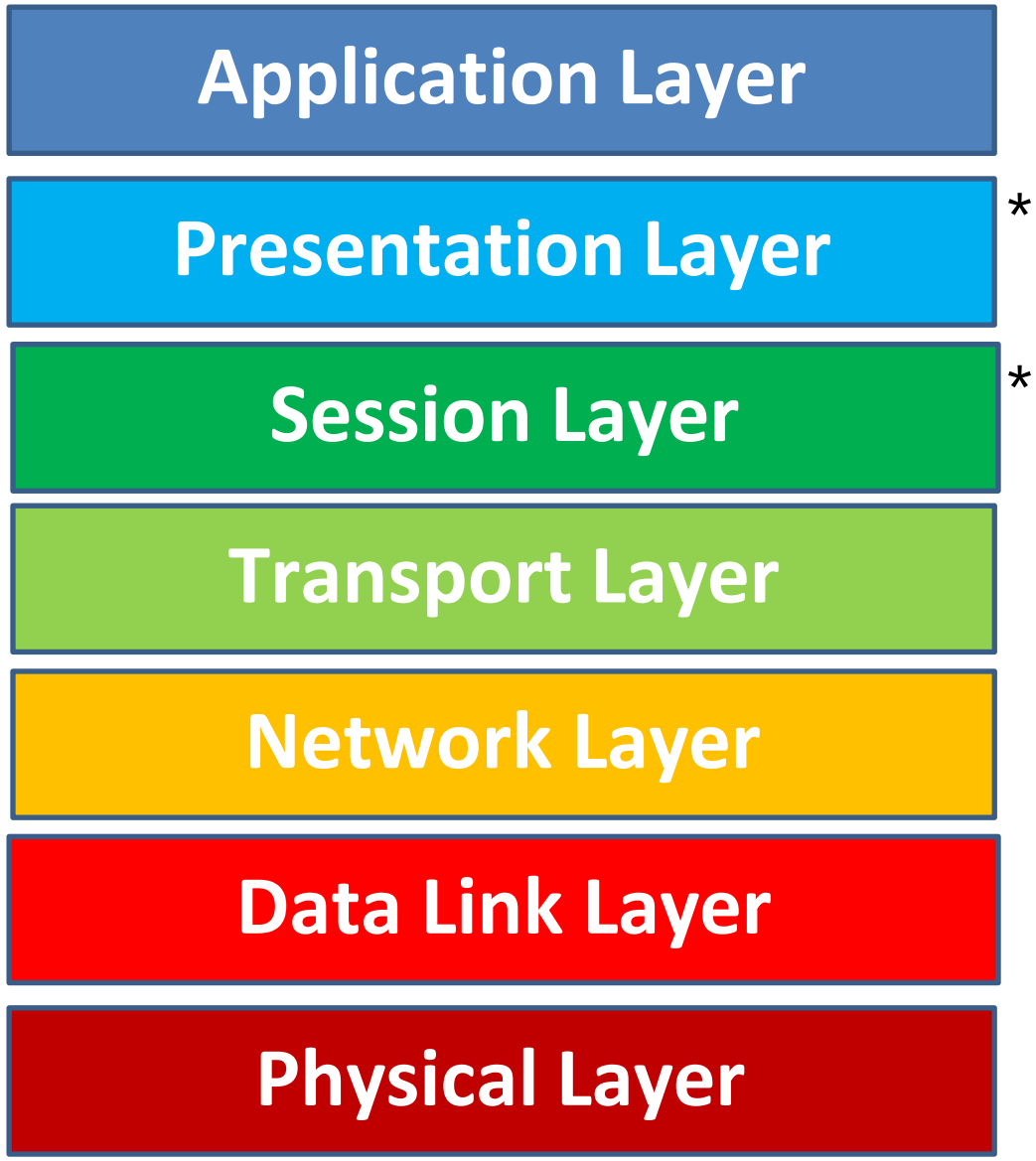
No lecture on Friday

Wireshark Lab due **tonight** at 11:59 PM

Quiz 3 on Friday

- Pipelining
- TCP/UDP
- Congestion Control
- Network Layer, Routers, Forwarding
- IP Addresses
- Subnets, NAT

OSI Model



Application Layer

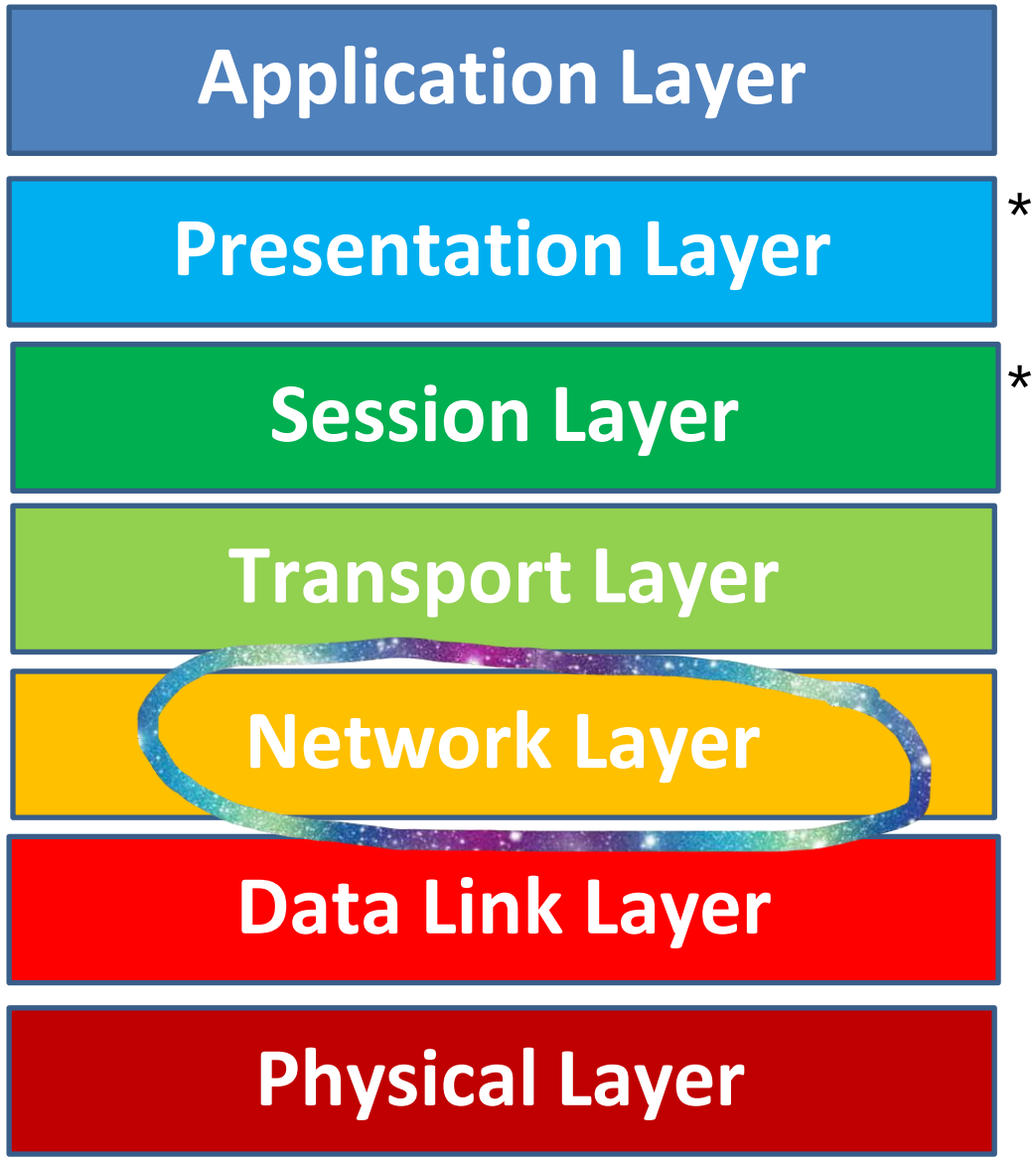
Messages from Network Applications

Physical Layer

Bits being transmitted over a medium

**In the textbook, they condense it to a 5-layer model, but 7 layers is what is most used*

OSI Model



Application Layer

Messages from Network Applications

Physical Layer

Bits being transmitted over a medium

**In the textbook, they condense it to a 5-layer model, but 7 layers is what is most used*

Application Layer

Transport Layer

Our packet of information so far…

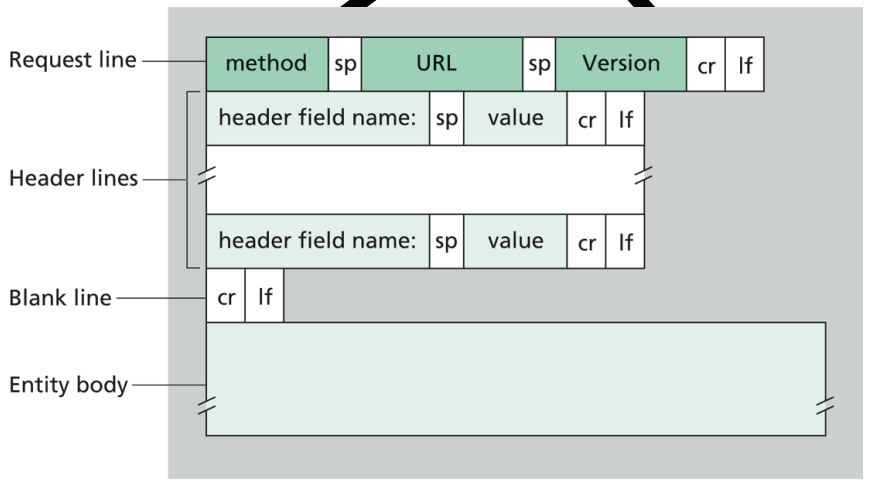
"Segment"

Message

Transport Layer Header

0	8	16	24	32
Source Port			Destination Port	
Sequence Number				
Acknowledgment Number				
Data Offset	Reserved	CWR ECE URG ACK PSH RST SYN FIN	Window Size	
Checksum			Urgent Pointer	
Options (if any)				

Transport Layer Protocol (ex. **TCP**)

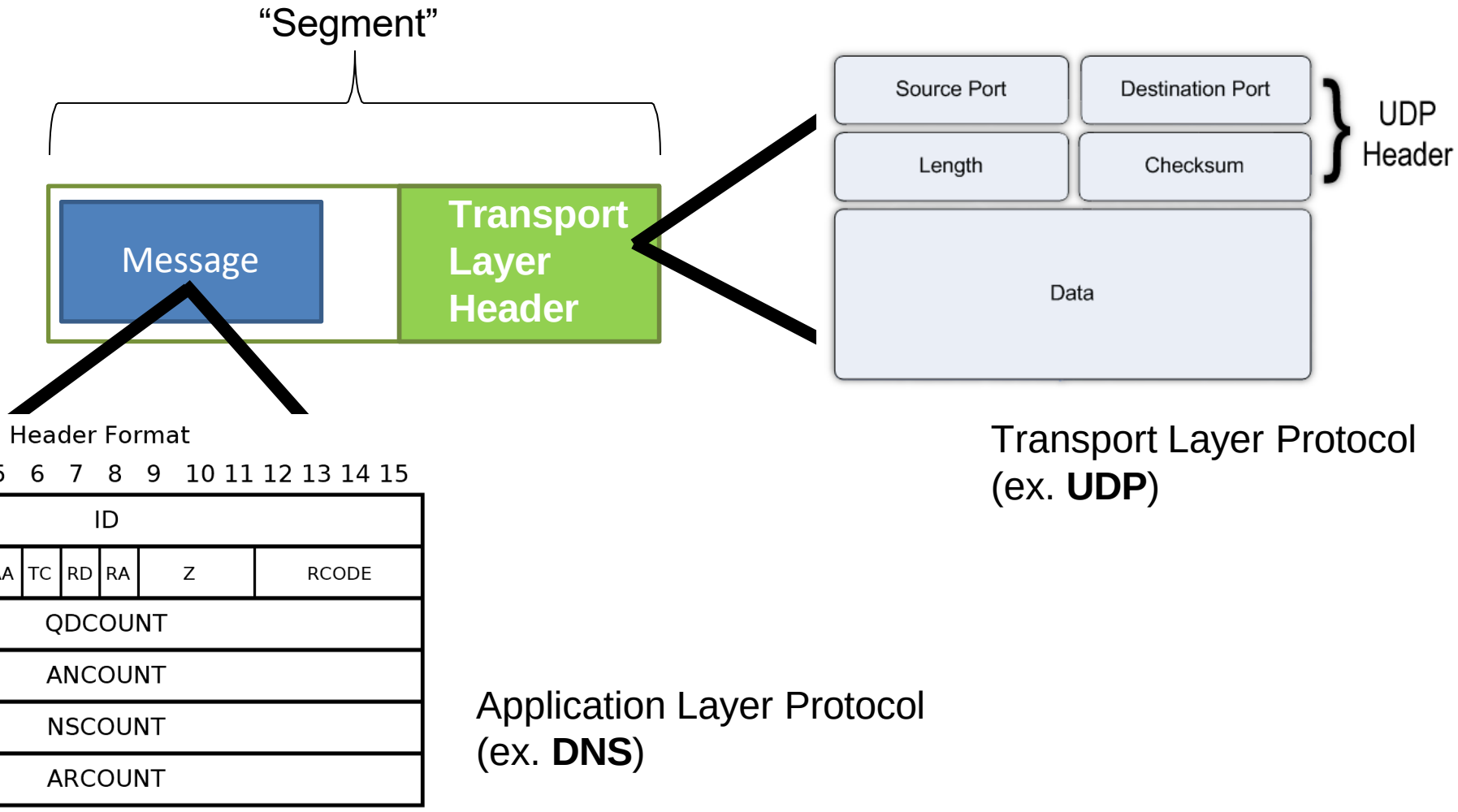


Application Layer Protocol (ex. **HTPP**)

Application Layer

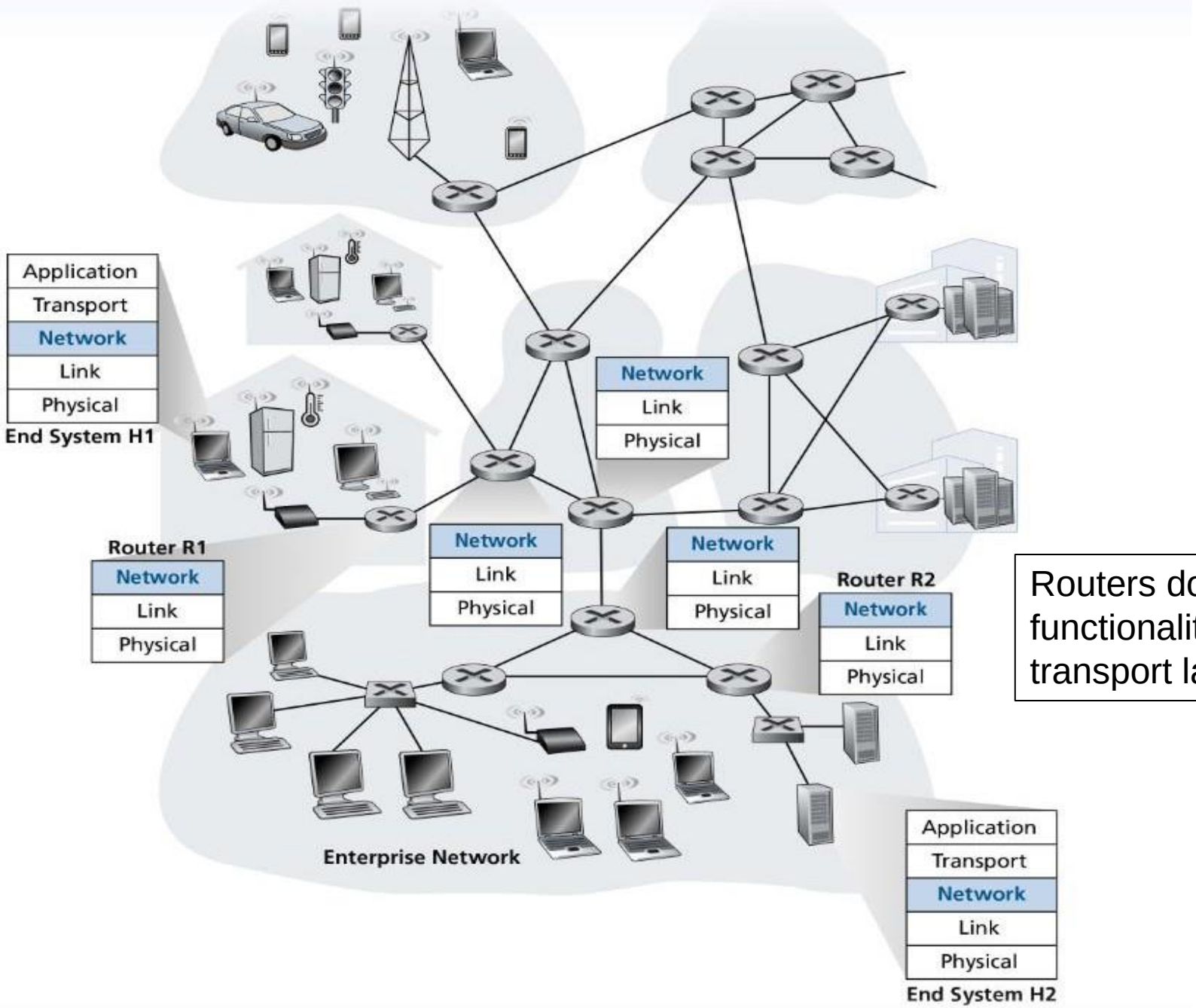
Transport Layer

Our packet of information so far…

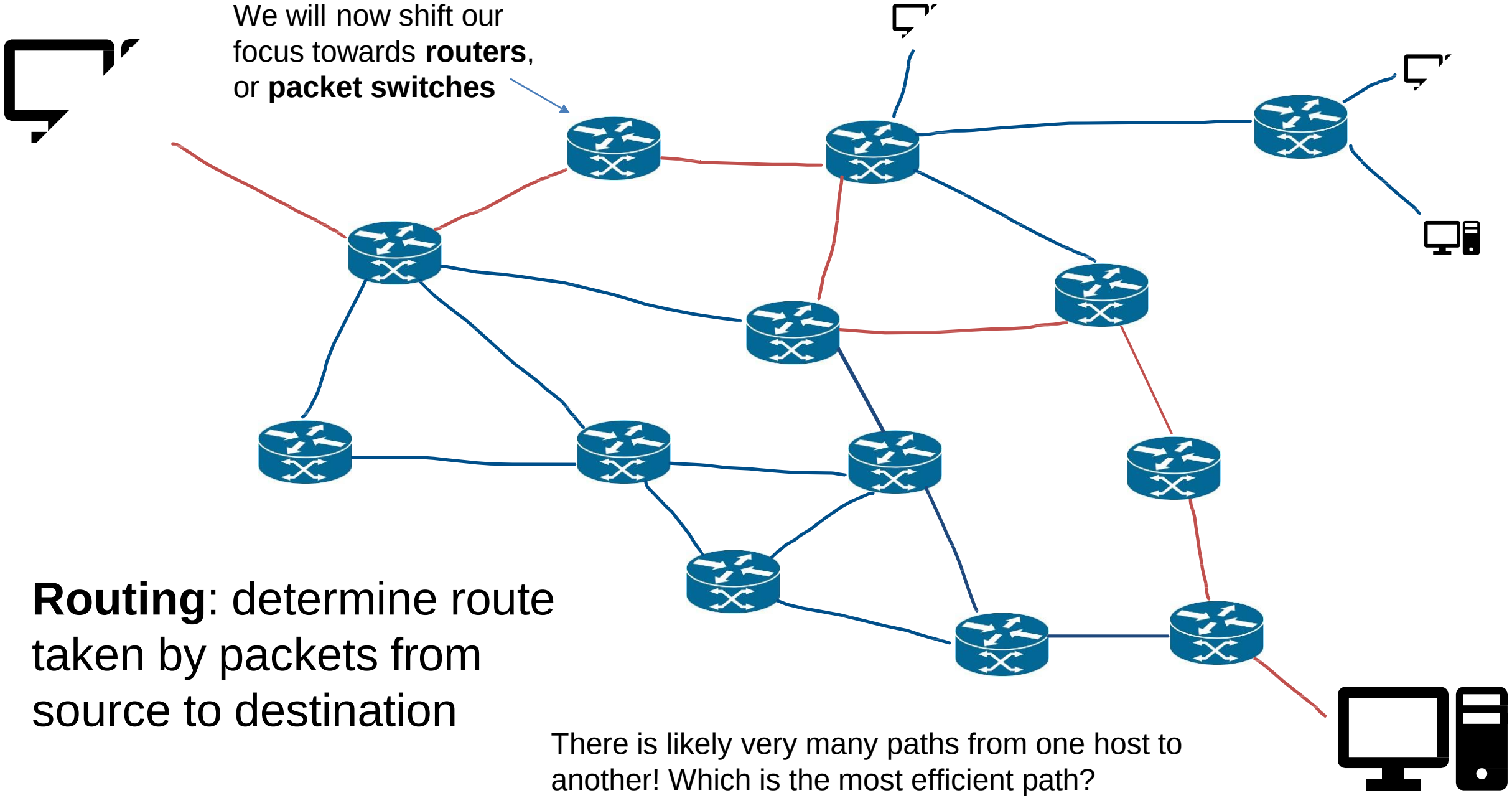


Transport Layer Protocol (ex. **UDP**)

Application Layer Protocol (ex. **DNS**)



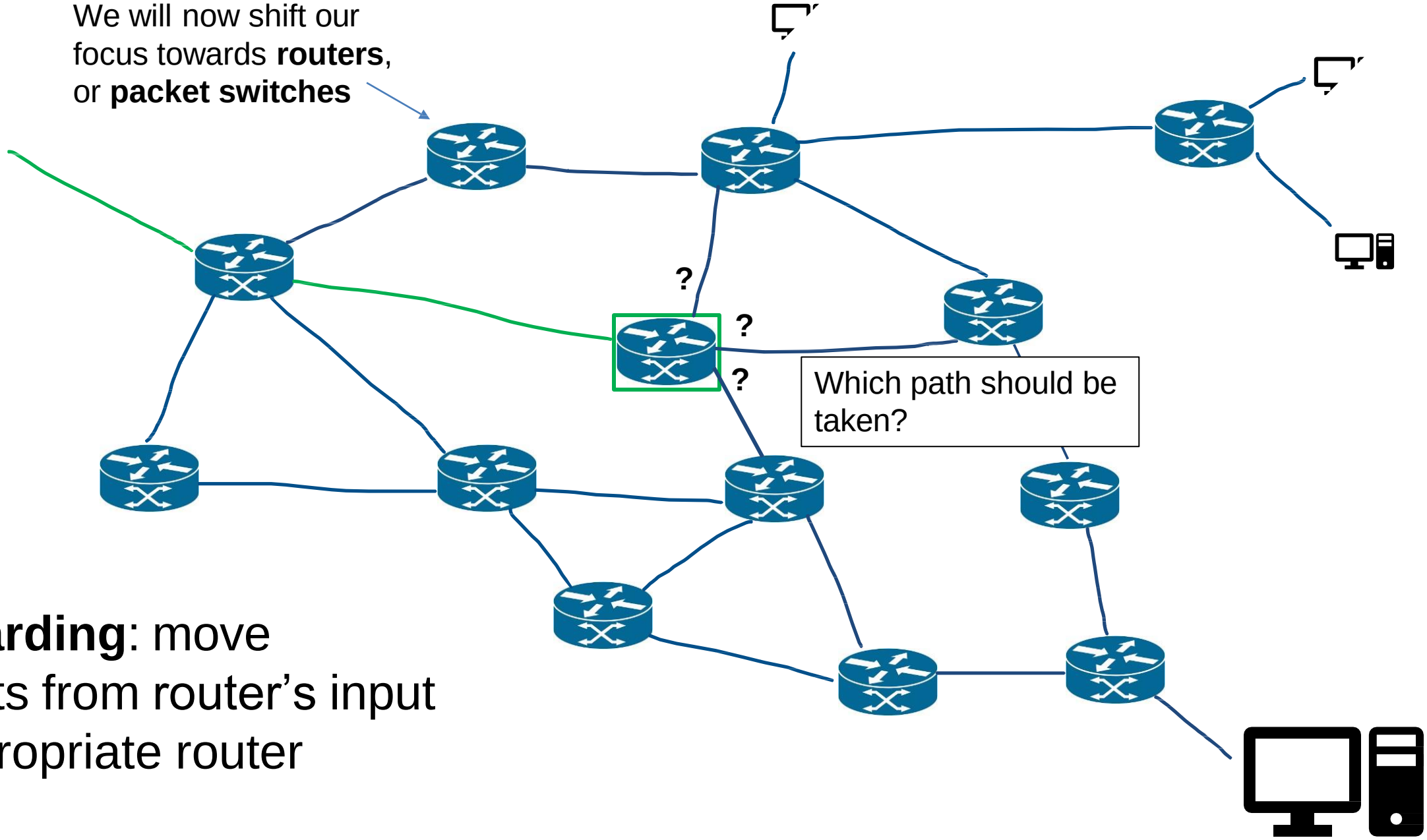
Routers do not have functionality for application & transport layer protocols



Routing: determine route taken by packets from source to destination

There is likely very many paths from one host to another! Which is the most efficient path?

We will now shift our focus towards **routers**, or **packet switches**



Forwarding: move packets from router's input to appropriate router output

Network Layer

Responsible for the delivery of data through a network

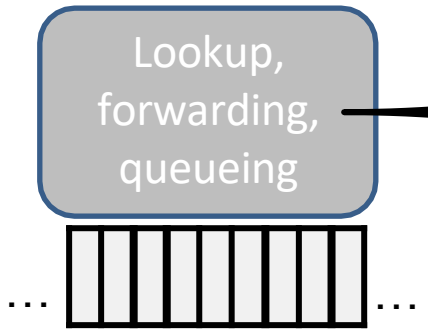
Forwarding

Data Plane

Routing

Control Plane

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

examples:

DA: 11001000 00010111 000**10110 10100001** which interface?

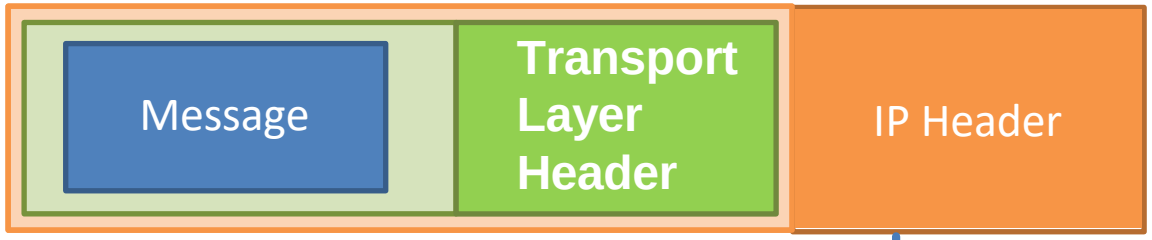
DA: 11001000 00010111 000**11000 10101010** which interface?

IP PROTOCOL

IP addressing, IPv4, and IPv6

https://www.rfc-editor.org/rfc/rfc791

Packets traversing through the network layer are referred to as a **datagram**. Each packet gets an IPv4/IPv6 header



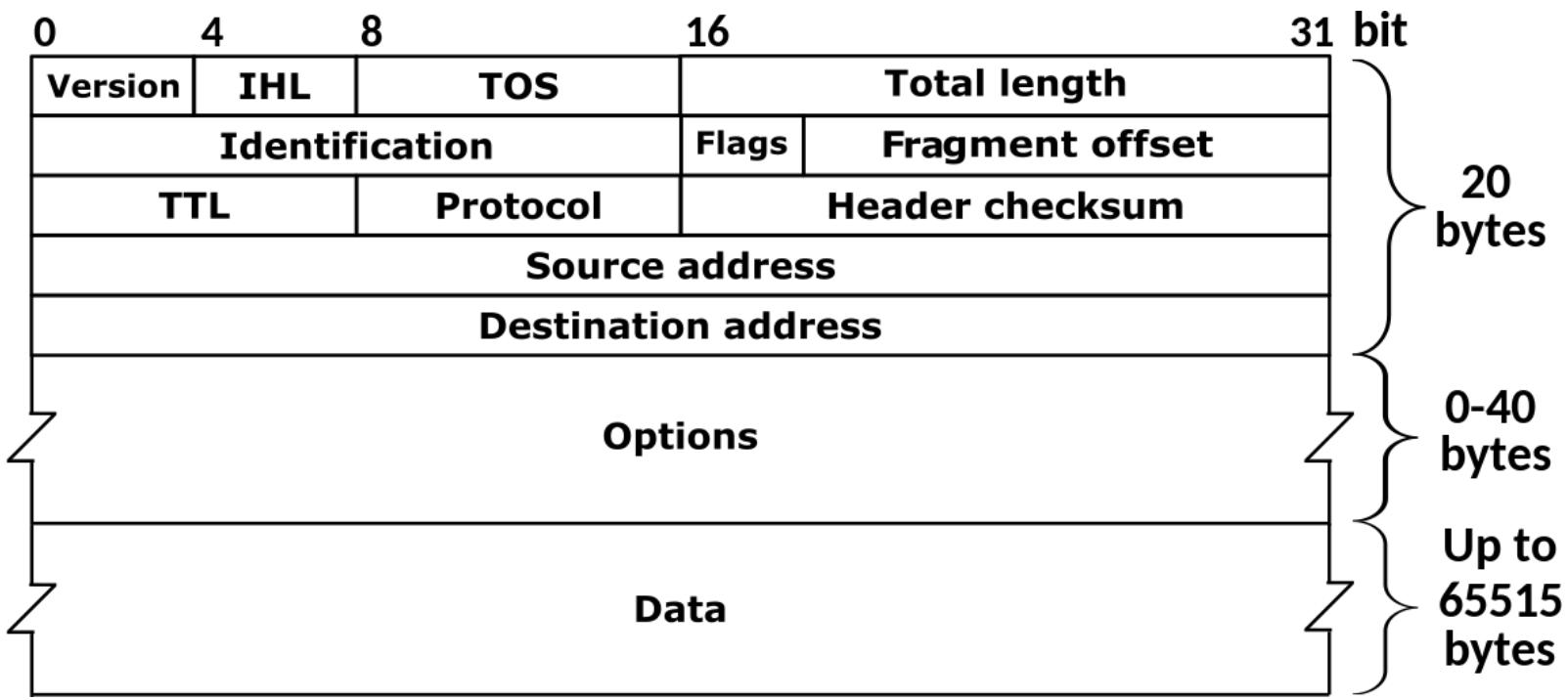
IPv4: 32-bit addresses (decimal)

192.149.252.76

IPv6: 128-bit addresses (hexademical)

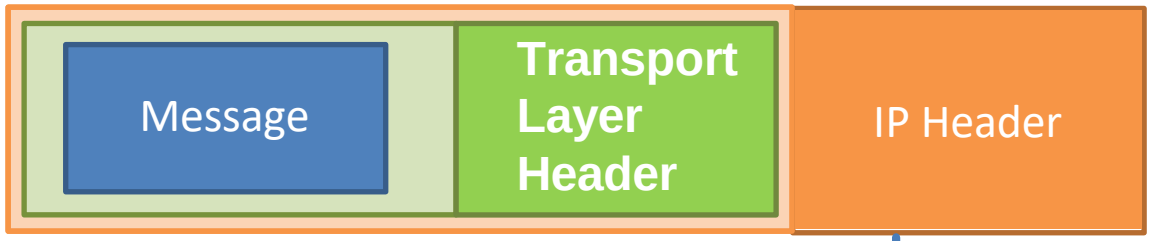
3ffe:1900:fe21:4545::

Why do we need IPv6?

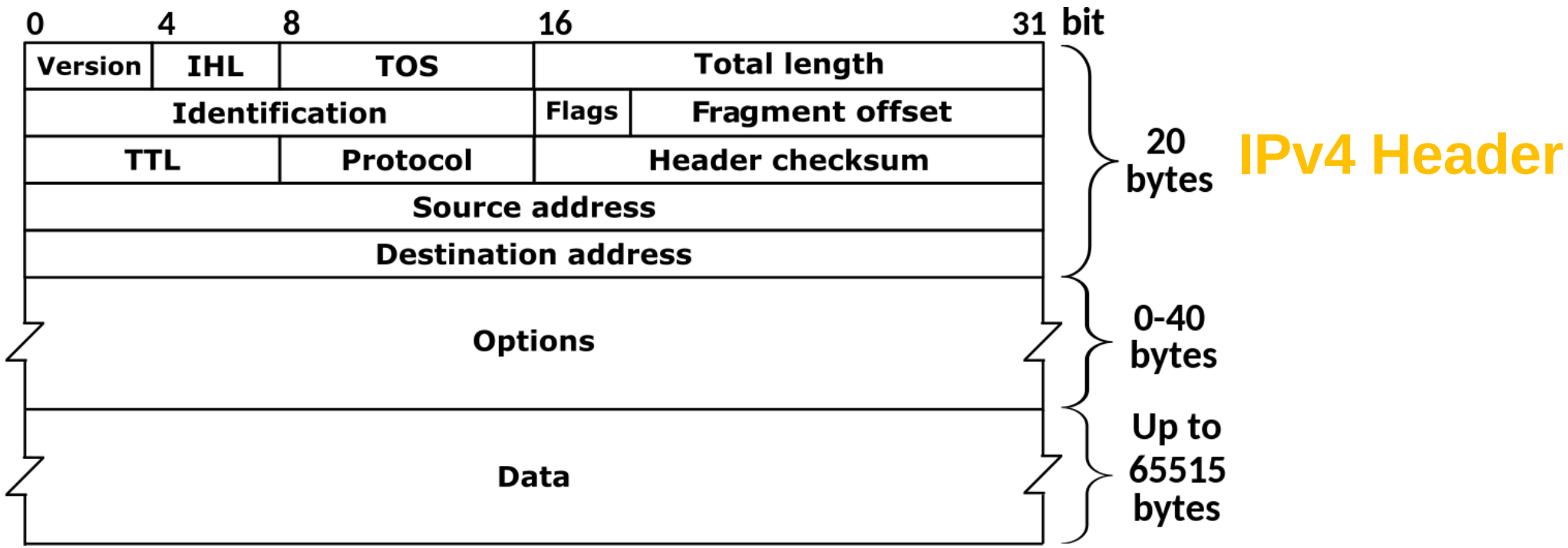


IPv4 Header

Packets traversing through the network layer are referred to as a **datagram**. Each packet gets an IPv4/IPv6 header



TCP header = 20 bytes
IP header = 20 bytes



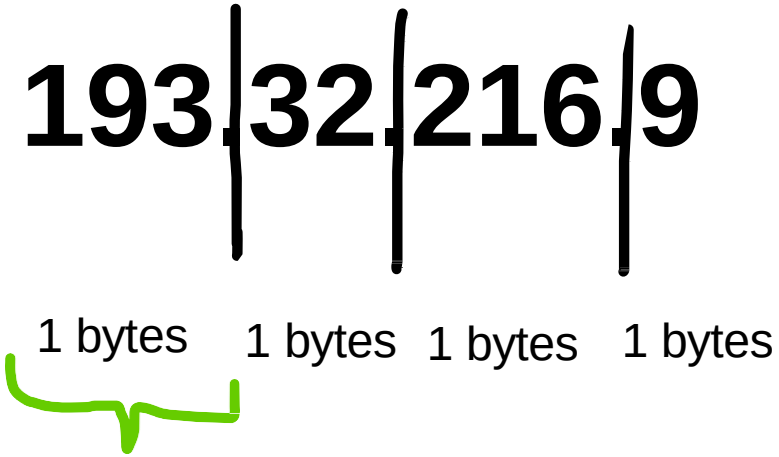
IPv4 Header

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

193.32.216.9

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

(1 byte = 8 bits)



128	64	32	16	8	4	2	1
1	**1**	0	0	0	0	0	**1**

128 + 64 + 1 = **193**

193 = `11000001` in binary

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

(1 byte = 8 bits)

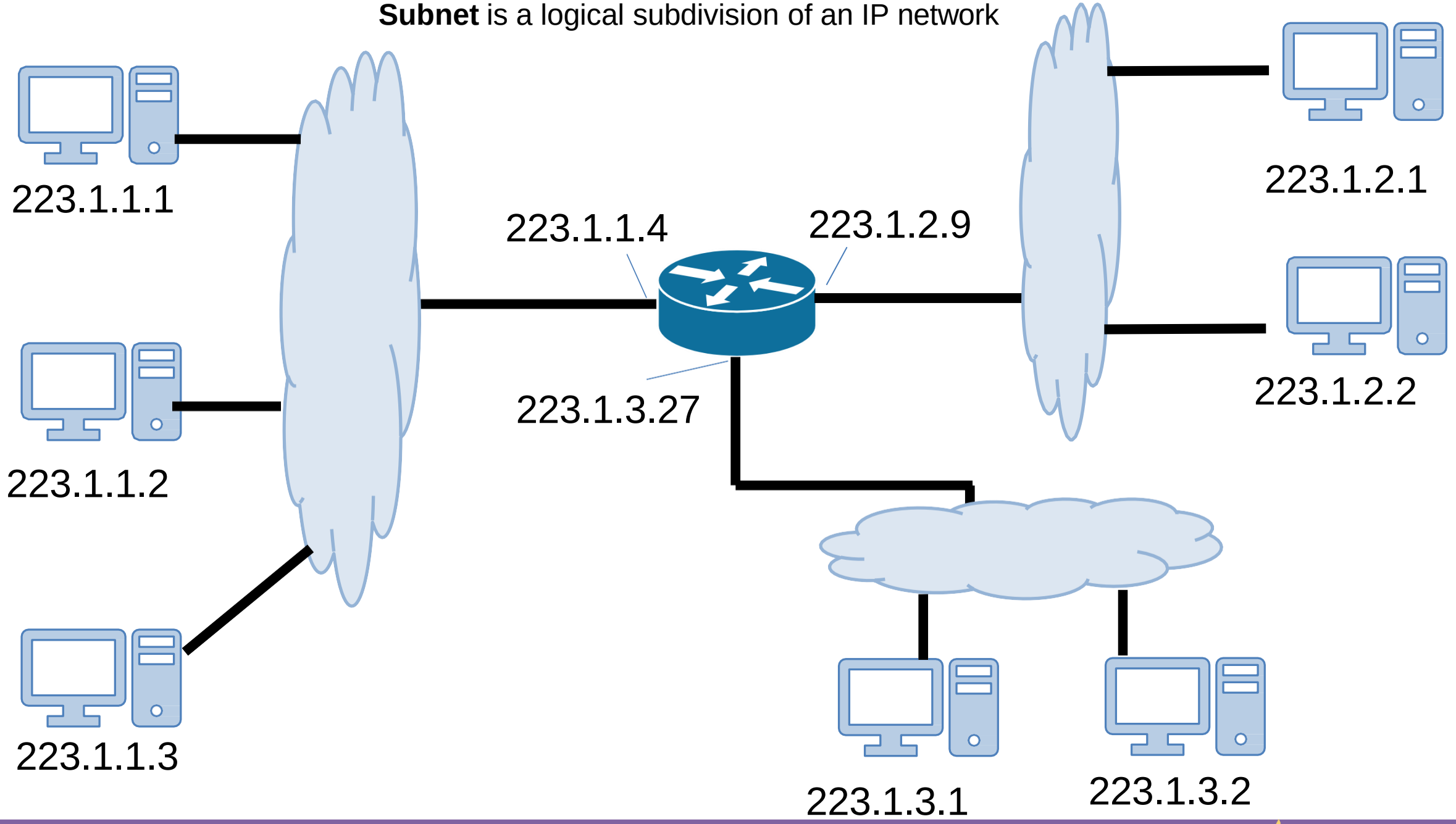
193.32.216.9

=

11000001 00100000 11011000 00001001

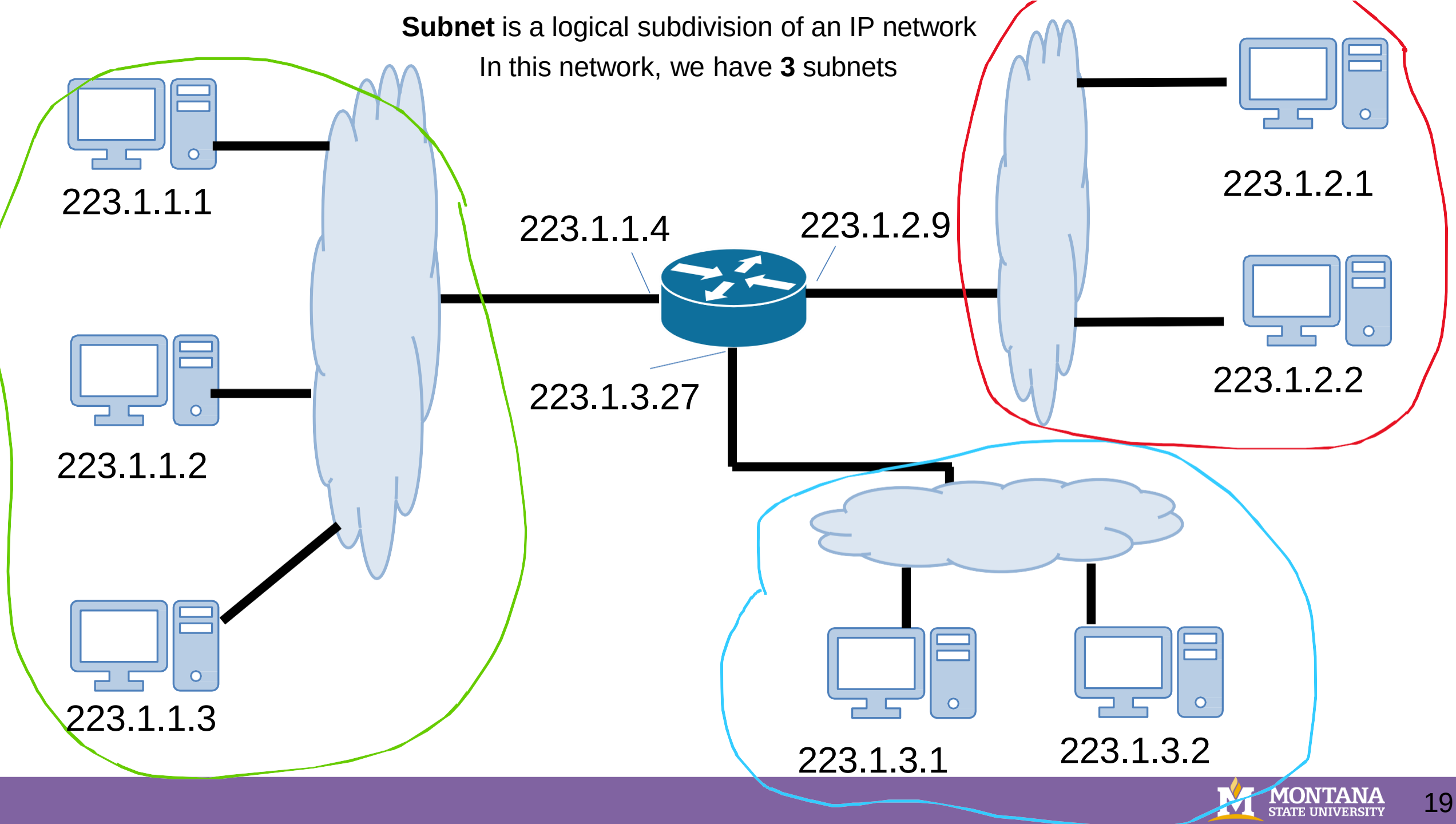
How many addresses are possible with a 32 bit number? **~4 billion possible IPv4 addresses**

Subnet is a logical subdivision of an IP network



Subnet is a logical subdivision of an IP network

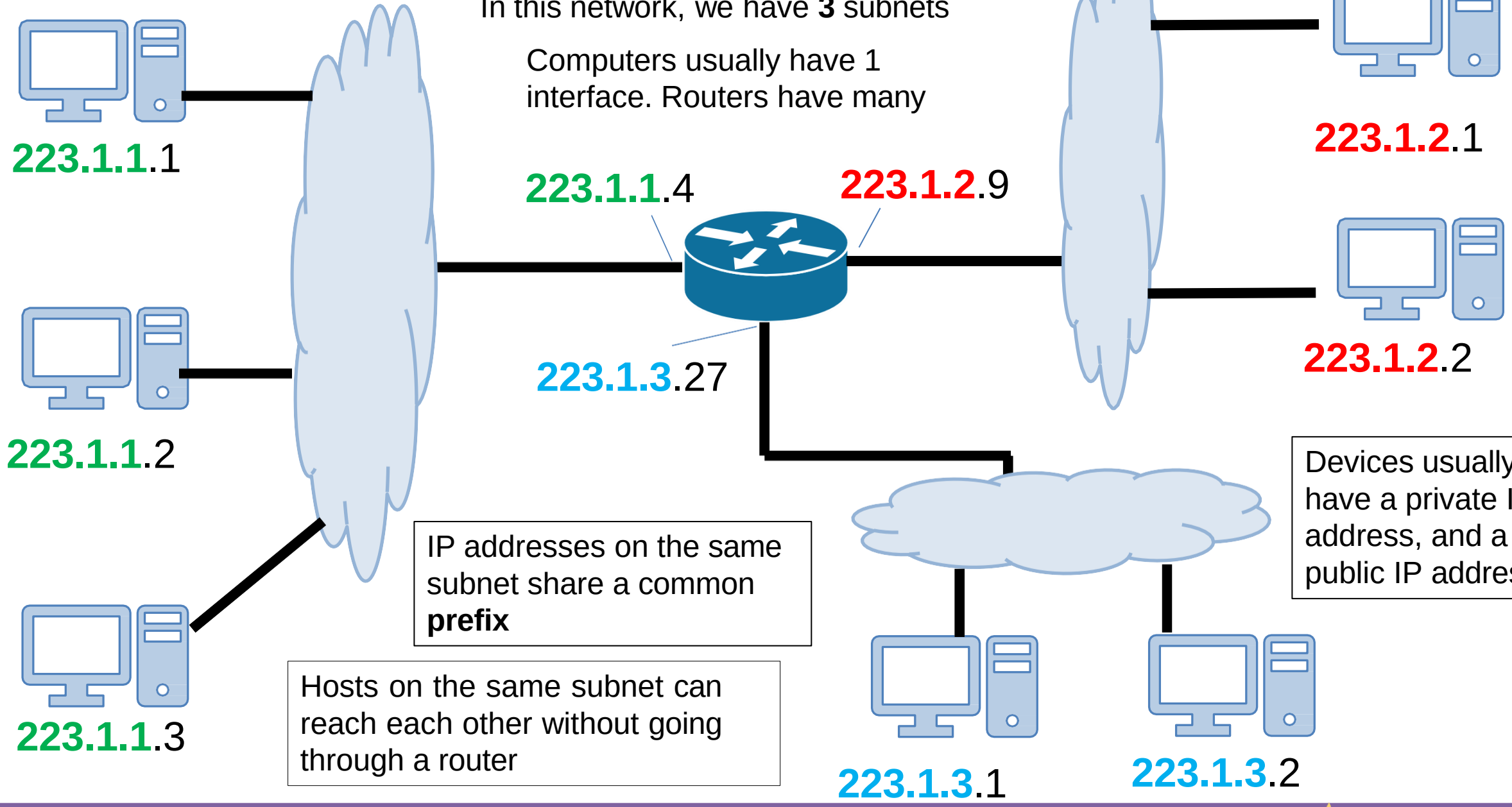
In this network, we have **3** subnets



Subnet is a logical subdivision of an IP network

In this network, we have **3** subnets

Computers usually have 1 interface. Routers have many

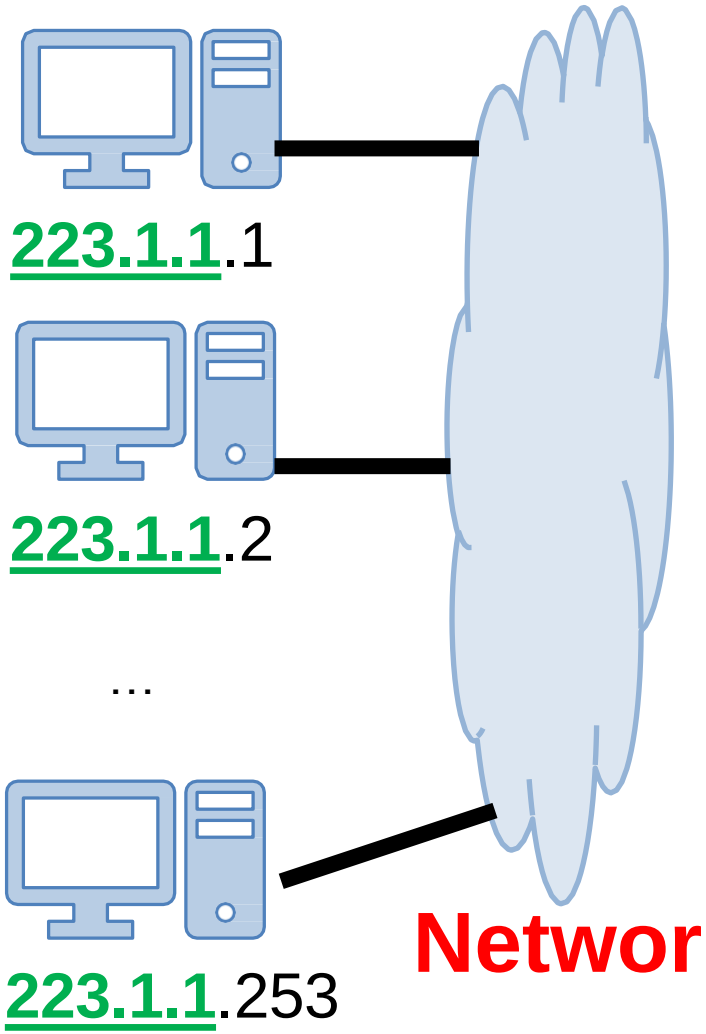


IP addresses on the same subnet share a common **prefix**

Hosts on the same subnet can reach each other without going through a router

Devices usually have a private IP address, and a public IP address

It is very common to have a **range** of IP addresses assigned to you (random assignment would be chaos)



Subnet mask

223.1.1.0/24

The leftmost 24 bits represent the prefix of the subnet

11111111 11111111 11111111 XXXXXXXX = **255.255.255**.0

223.1.1.67 ✓

223.2.1.67 ✗

223.2.1.255 ✗

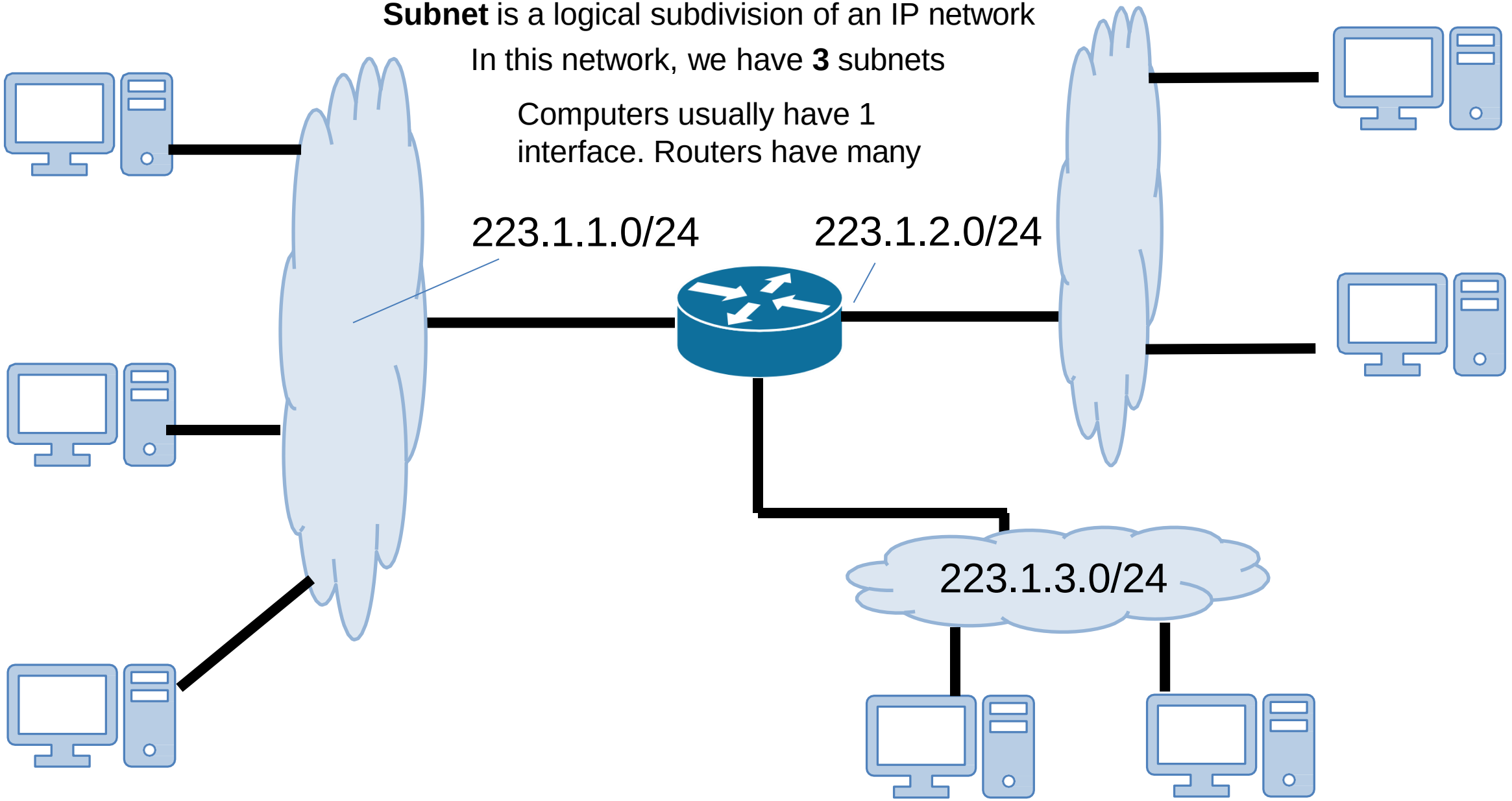
Network bits **193.32.216.9** **Host bits**

11000001 00100000 11011000 00001001

Subnet is a logical subdivision of an IP network

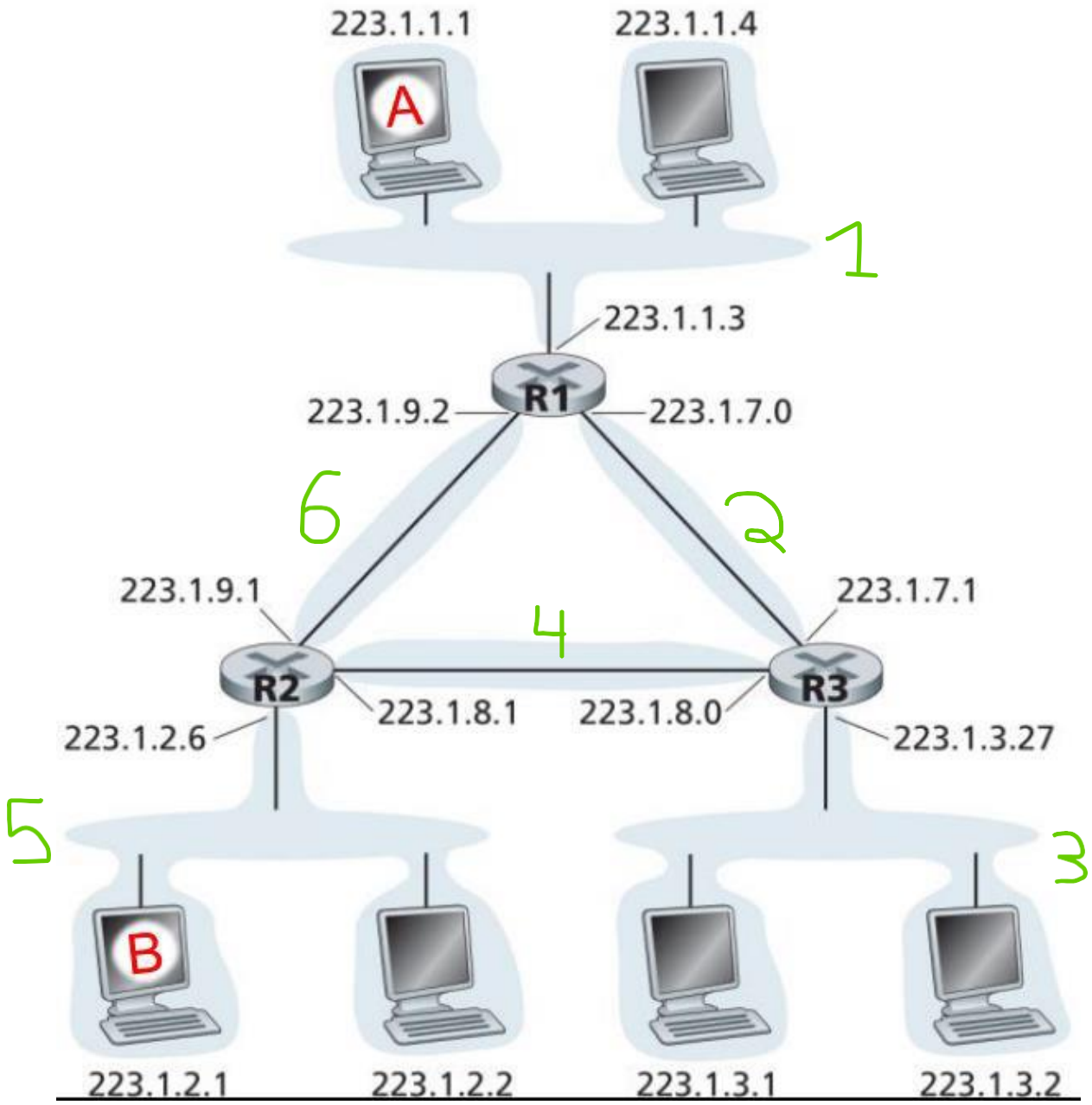
In this network, we have **3** subnets

Computers usually have 1 interface. Routers have many



6 subnets

1. 223.1.1.0/24
2. 223.1.7.0/24
3. 223.1.3.0/24
4. 223.1.8.0/24
5. 223.1.2.0/24
6. 223.1.9.0/23



Managing and Assigning Subnets

The subnet `223.1.1.0/24` contains the following addresses

```
223.1.1.1
223.1.1.2
223.1.1.3
223.1.1.4
223.1.1.5
…
223.1.1.252
223.1.1.253
223.1.1.254
```

How many addresses does a /24 subnet provide?

0 and 255 are reserved for special services (?)

Managing and Assigning Subnets

The subnet `223.1.1.0/24` contains the following addresses

```
223.1.1.1
223.1.1.2
223.1.1.3
223.1.1.4
223.1.1.5
…
223.1.1.252
223.1.1.253
223.1.1.254
```

How many addresses does a /24 subnet provide?

2^8 – 2 = **253**

Class C network

0 and 255 are reserved for special services (?)

Managing and Assigning Subnets

Subnet Type	Slash	Subnet Mask	Available Addresses
Class A	/8	255.0.0.0	2^24 – 2 = **16777216**
Class B	/16	255.255.0.0	2^16 – 2 = **65,634**
Class C	/24	255.255.255.0	2^8 – 2 = **254**

Issues with this type of assignment?

Managing and Assigning Subnets

Subnet Type	Slash	Subnet Mask	Available Addresses
Class A	/8	255.0.0.0	2^24 – 2 = **16777216**
Class B	/16	255.255.0.0	2^16 – 2 = **65,634**
Class C	/24	255.255.255.0	2^8 – 2 = **254**

If I have 2000 devices, I will either need 8 class C subnets or 1 class B subnet (but waste 63,000ish IP addresses)

We need a better way to subnet!

Managing and Assigning Subnets

Our only available subnet masks have been:

255.255.255.0	11111111 11111111 11111111 00000000	/24
255.255.0.0	11111111 11111111 00000000 00000000	/16
255.0.0.0	11111111 00000000 00000000 00000000	/8

CIDR introduces a more flexible way for subnetting

We can any number of bits for our mask

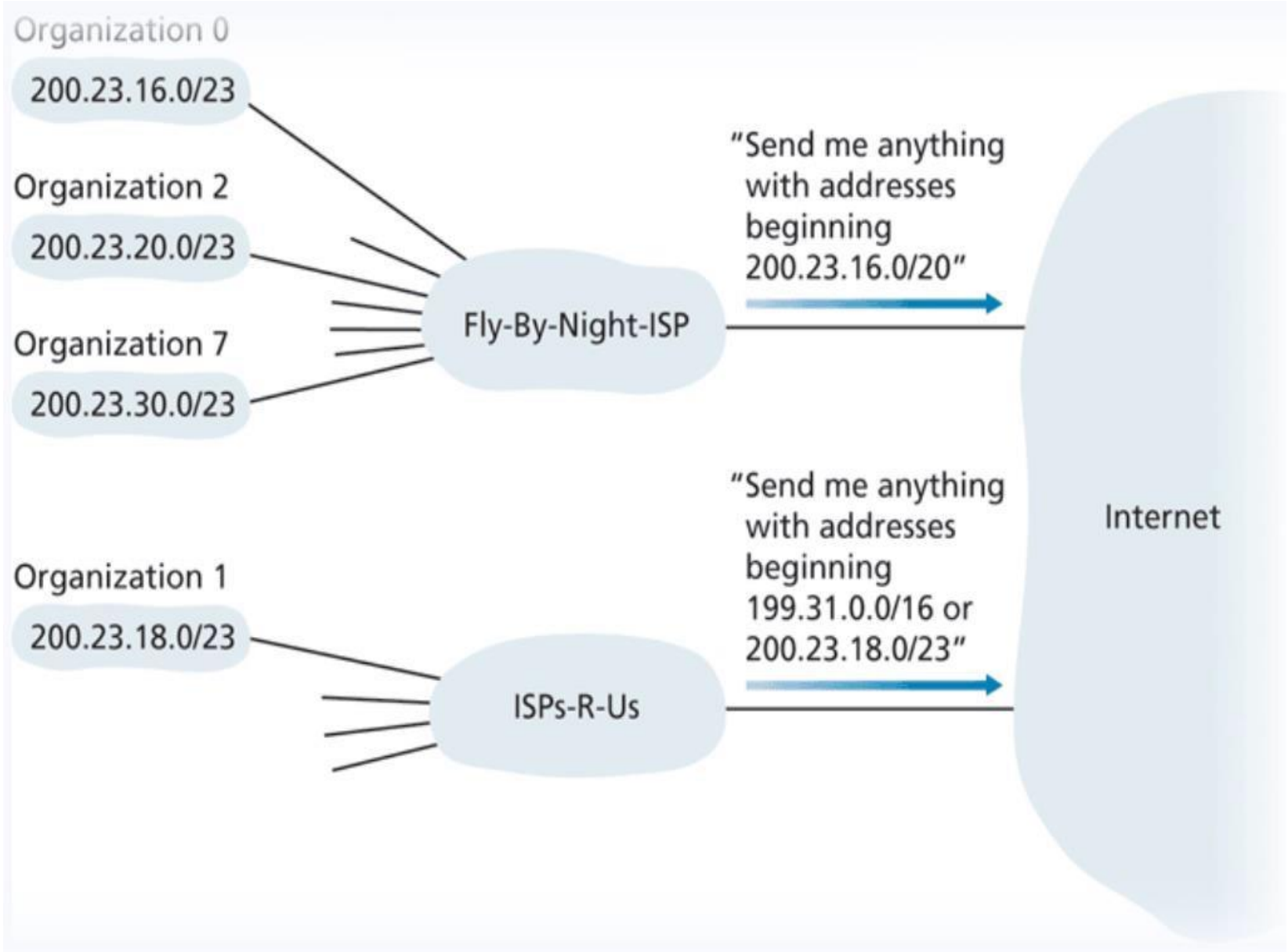
Host bits!

200.23.16.0**/20 =** `11001000.00010111.0001`**`0000.00000000`**

Mask = `11111111.11111111.1111`**`0000.00000000`**

ISPs can now group and advertise organizations by IP blocks

We once again have another hierarchy ☺

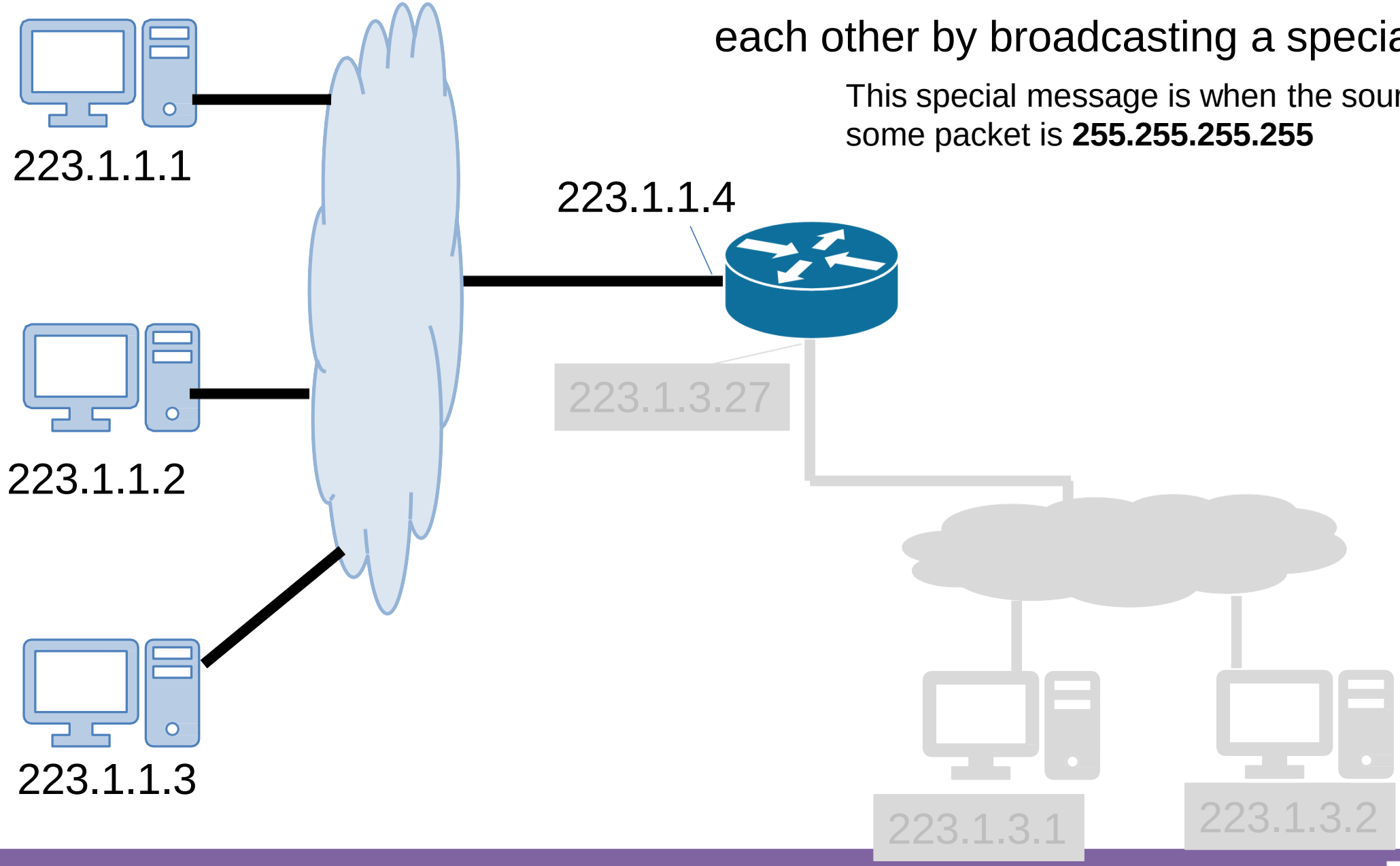


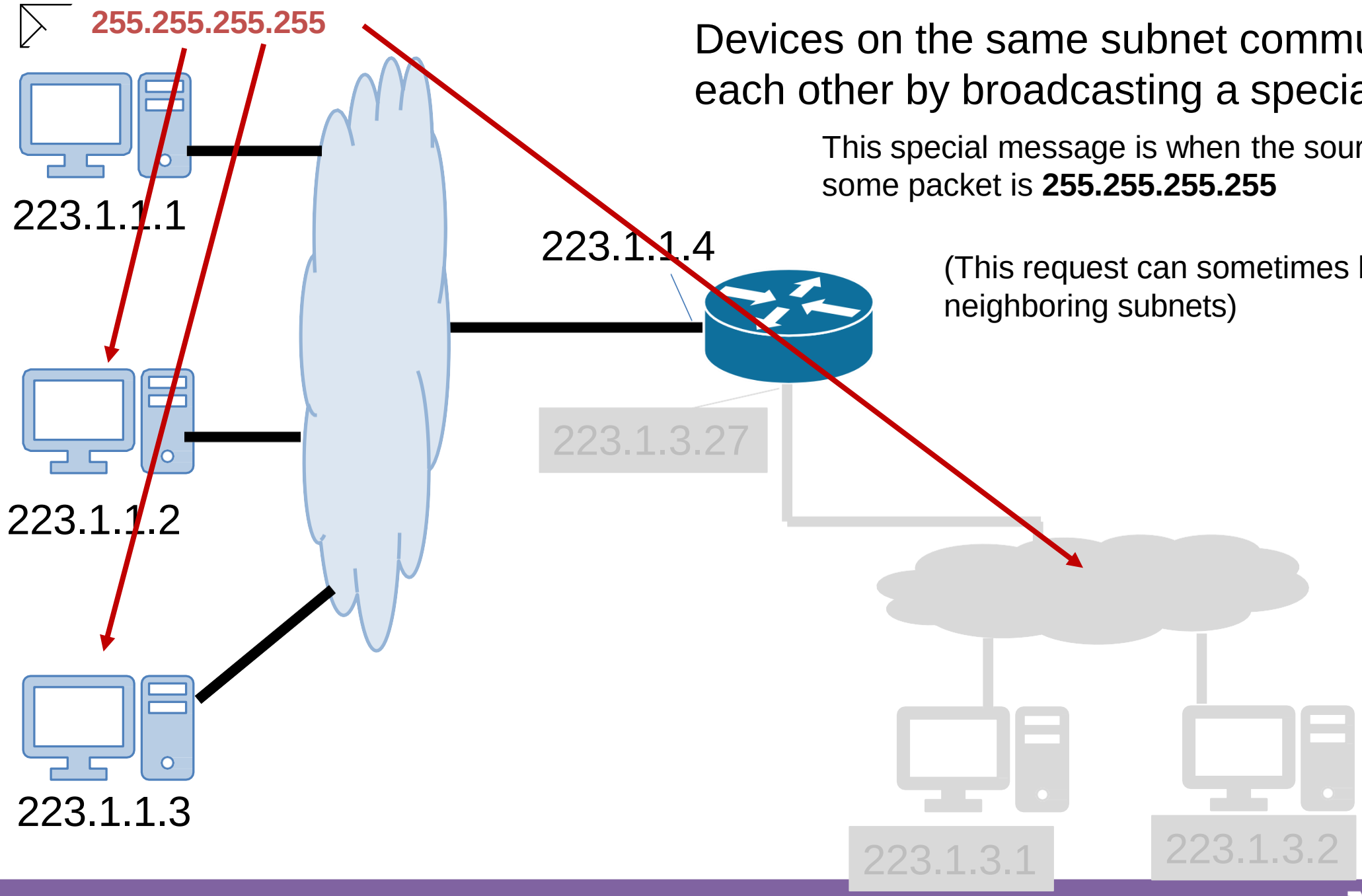
One possible way an ISP could organize IP blocks

ISP's block	200.23.16.0/20	11001000 00010111 00010000 00000000
Organization 0	200.23.16.0/23	**11001000 00010111 0001000**0 00000000
Organization 1	200.23.18.0/23	**11001000 00010111 0001001**0 00000000
Organization 2	200.23.20.0/23	**11001000 00010111 0001010**0 00000000
…		
Organization 7	200.23.30.0.23	**11001000 00010111 0001111**0 00000000

Devices on the same subnet communicate with each other by broadcasting a special message

This special message is when the source IP address of some packet is **255.255.255.255**





Devices on the same subnet communicate with each other by broadcasting a special message

This special message is when the source IP address of some packet is **255.255.255.255**

(This request can sometimes be echoed to neighboring subnets)

A **subnet mask** can be used to reveal which bits are the host bits

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0	10101100.00010000.00000001.00000000
255.255.0.0	11111111.11111111.00000000.00000000

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.00000001.00000000

255.255.0.0 11111111.11111111.00000000.00000000

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.**00000001.00000000**

255.255.0.0 11111111.11111111.00000000.00000000

The last two octets do not match 1s, so they are the **host bits**

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.**00000001.00000000**

255.255.0.0 11111111.11111111.00000000.00000000

The last two octets do not match 1s, so they are the **host bits**

This IP address would fall under the range of:

172.16.0.0 **/16**

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0	10101100.00010000.00000001.00000000
255.255.224.0	11111111.11111111.11100000.00000000

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.000**00001.00000000**

255.255.224.0 11111111.11111111.11100000.00000000

The last 13 bits are the host bits, the first 19 bits are the network bits

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.000**00001.00000000**

255.255.224.0 11111111.11111111.11100000.00000000

The last 13 bits are the host bits, the first 19 bits are the network bits

This IP address would fall under the range of

172.16.0.0/19

CIDR

A **subnet mask** can be used to reveal which bits are the host bits

172.16.1.0 10101100.00010000.000**00001.00000000**

255.255.224.0 11111111.11111111.11100000.00000000

The last 13 bits are the host bits, the first 19 bits are the network bits

This IP address would fall under the range of

172.16.0.0/19 CIDR

A **subnet mask** can be used to reveal which bits are the host bits

CIDR 172.0.0.0/9

Subnet Mask?

A **subnet mask** can be used to reveal which bits are the host bits

CIDR 172.0.0.0/9

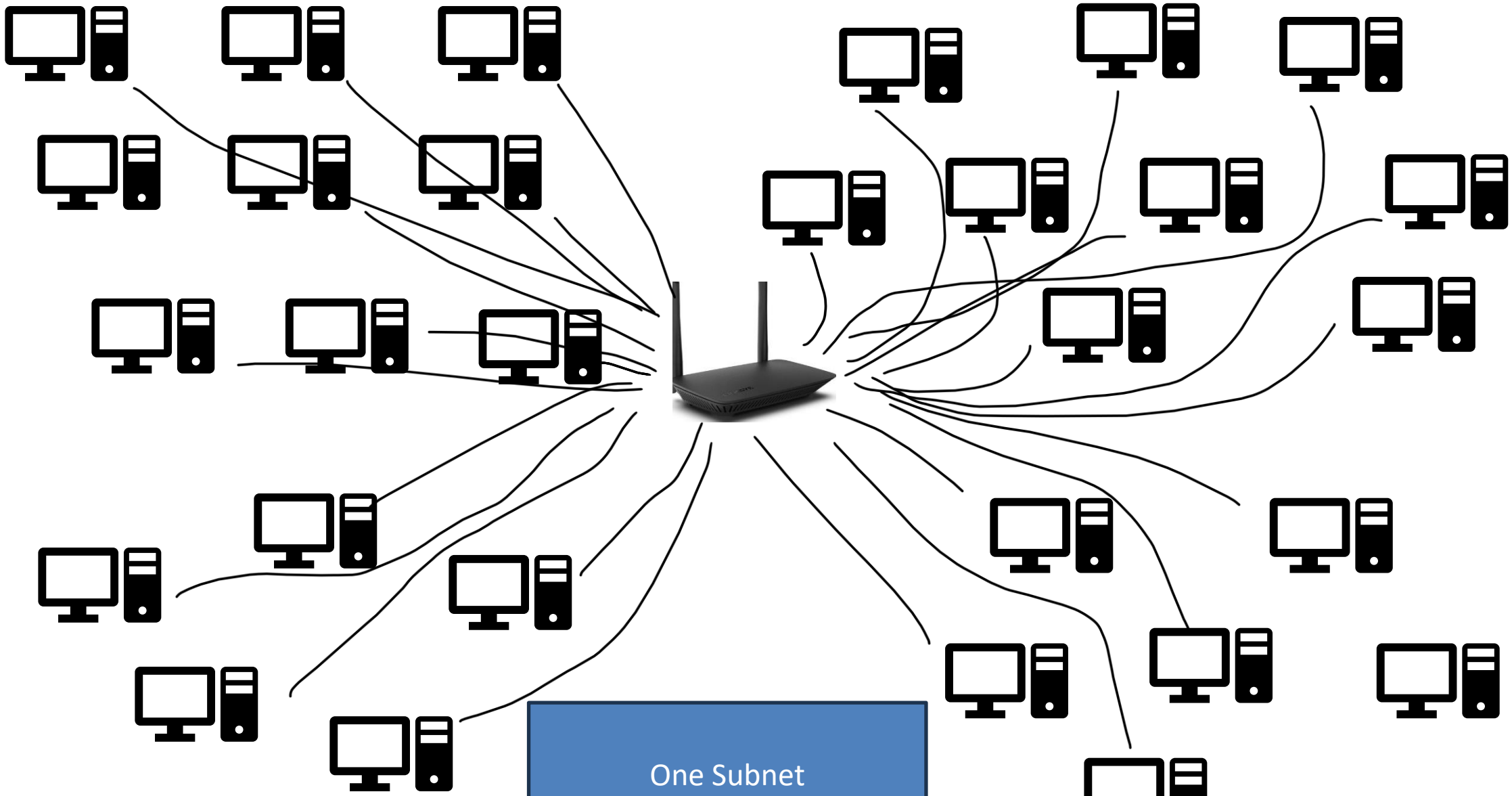
Subnet Mask? 11111111.10000000.00000000.00000000

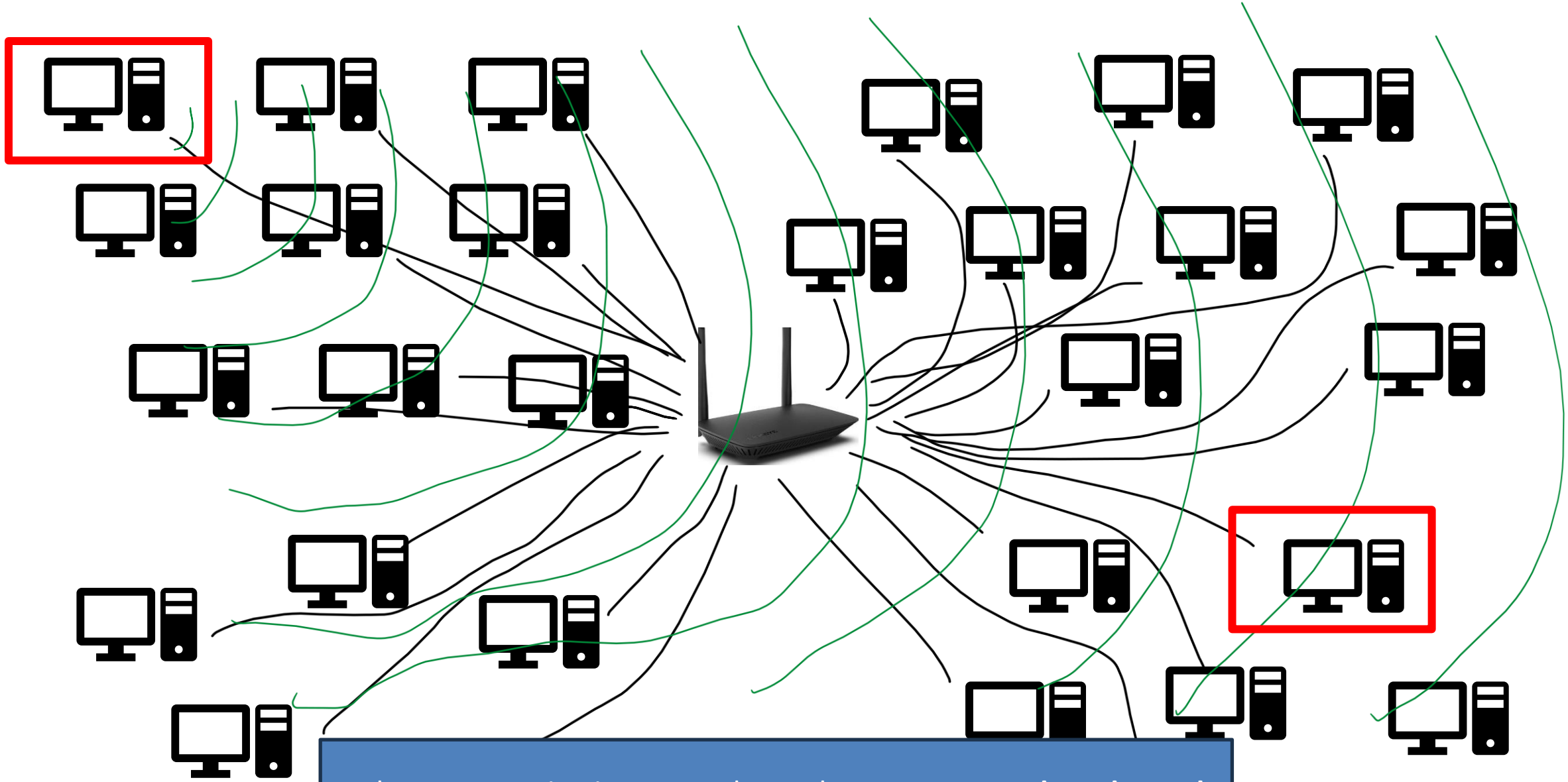
A **subnet mask** can be used to reveal which bits are the host bits

CIDR 172.0.0.0/9

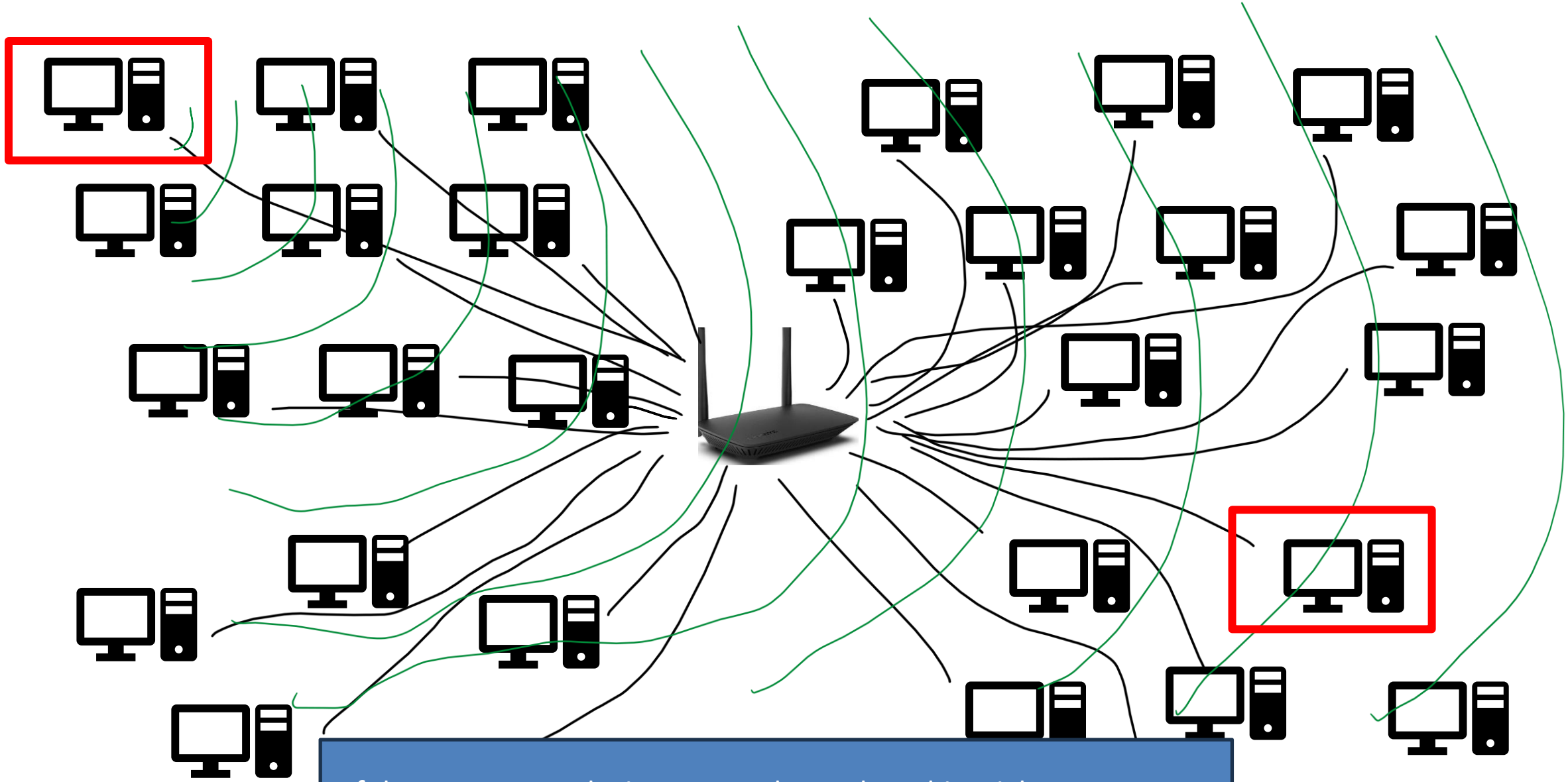
Subnet Mask? 11111111.10000000.00000000.00000000

255.128.0.0

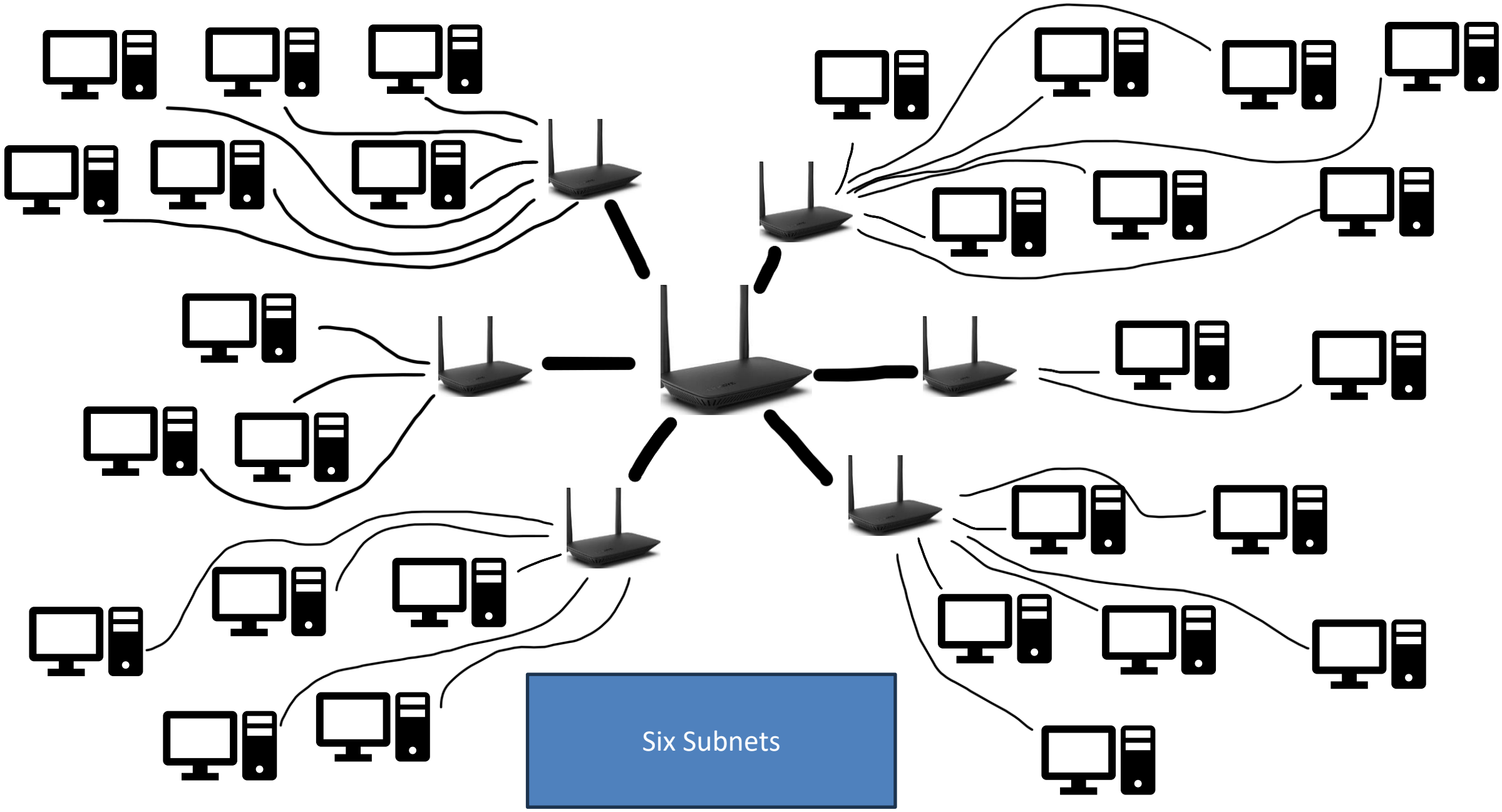

One Subnet

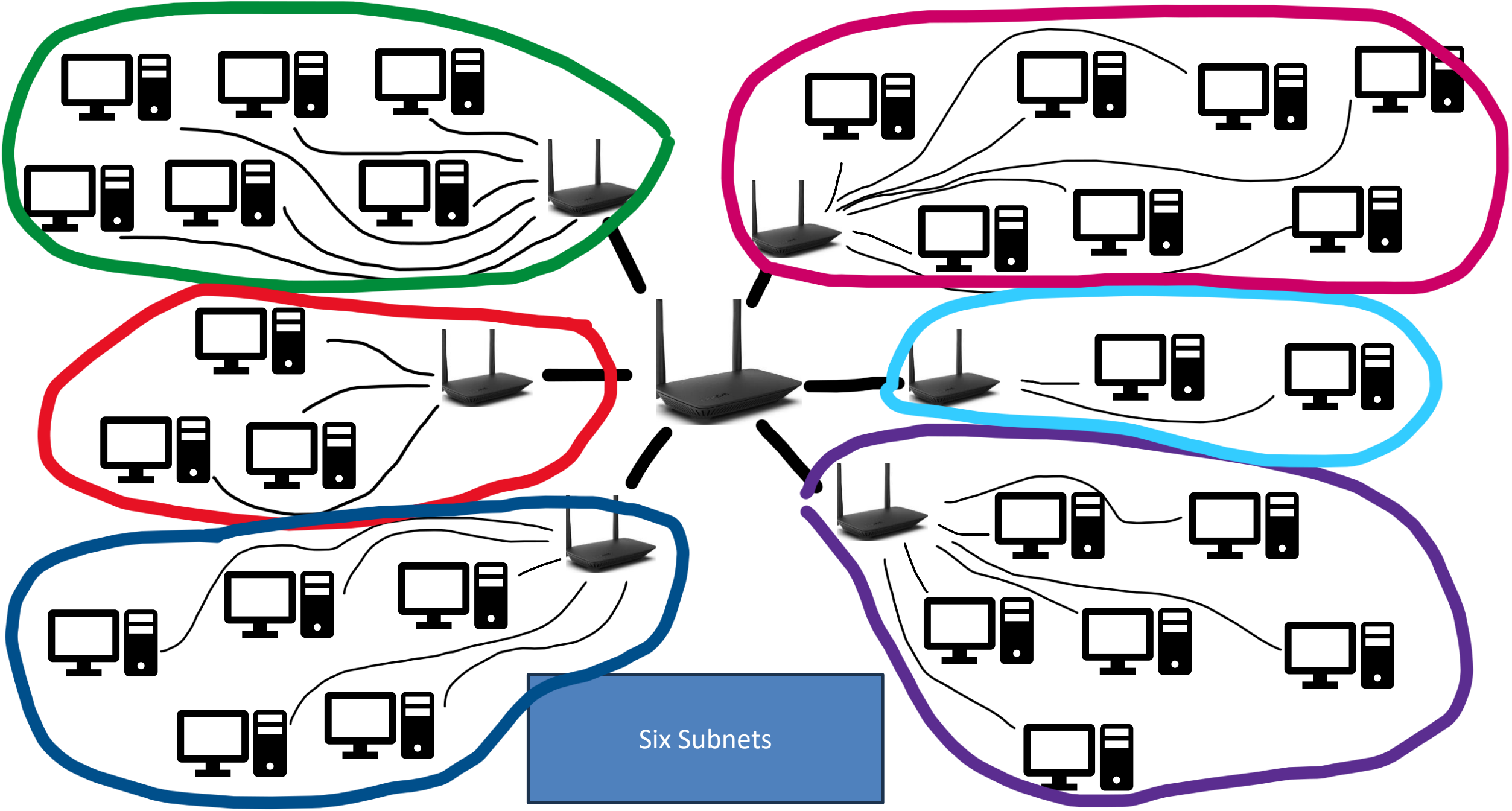


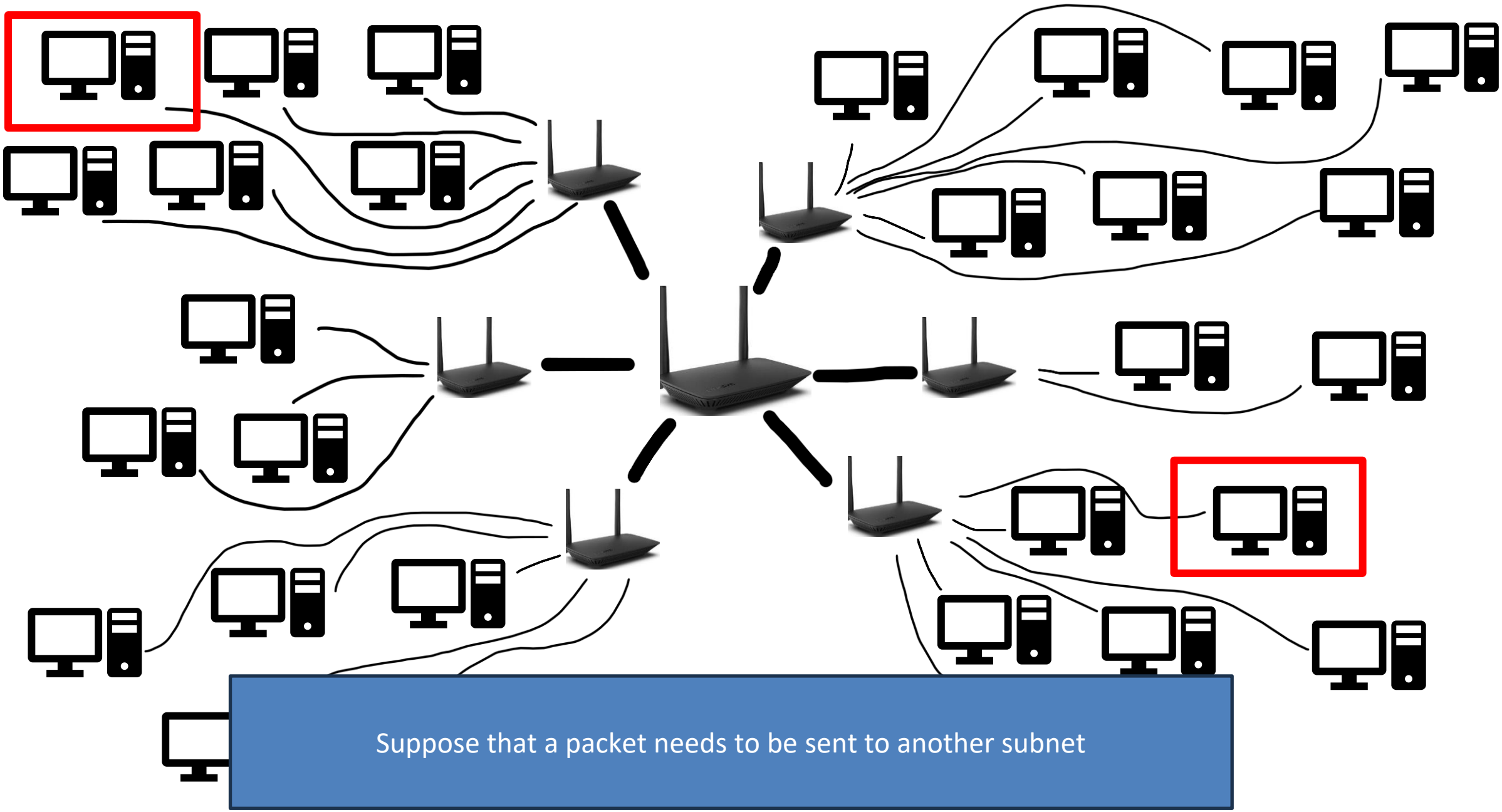
When communicating on a subnet, the message gets **broadcasted** to every device on the network

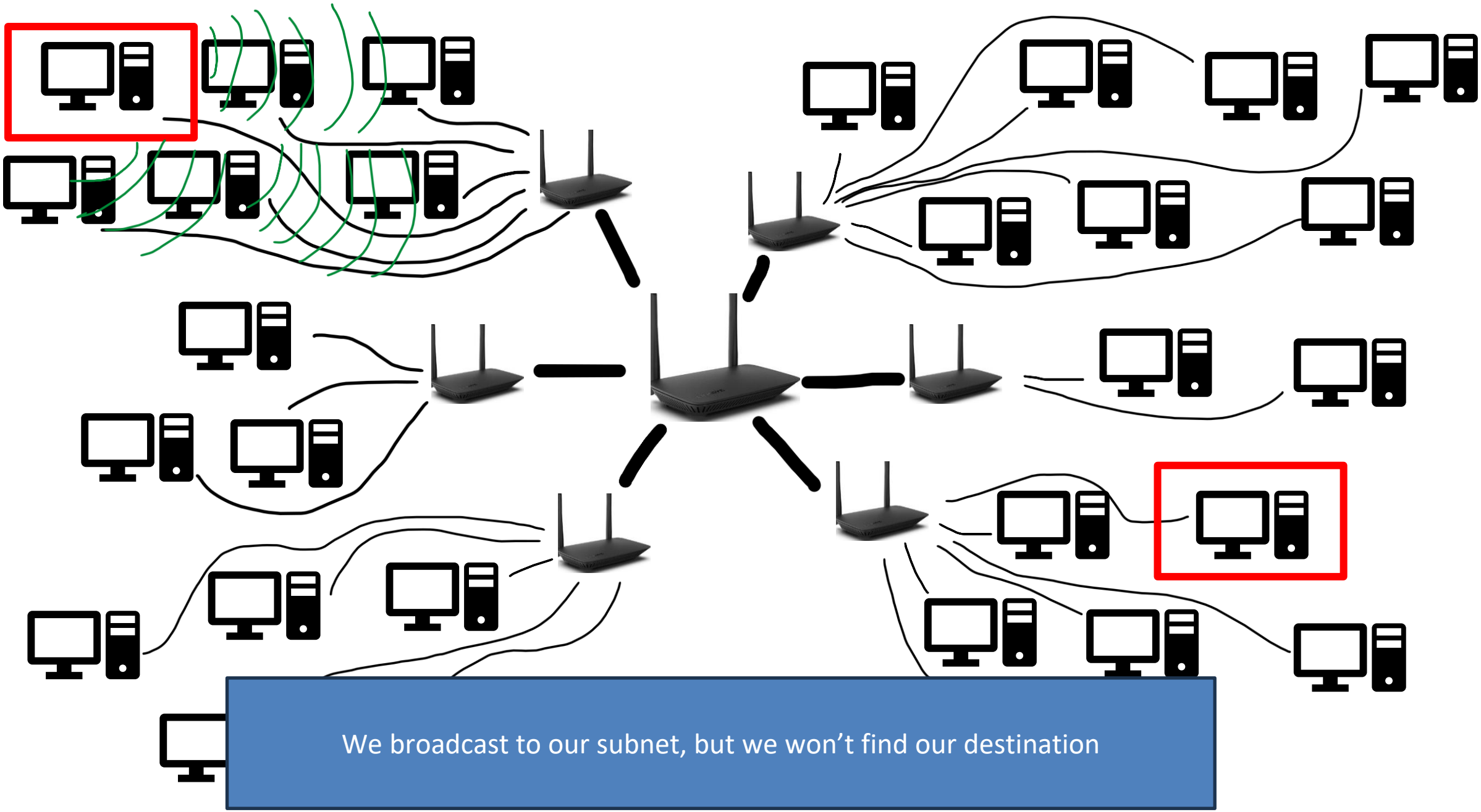


If there are many devices on a subnet, then this might cause some problems (congestion, lag, delay)

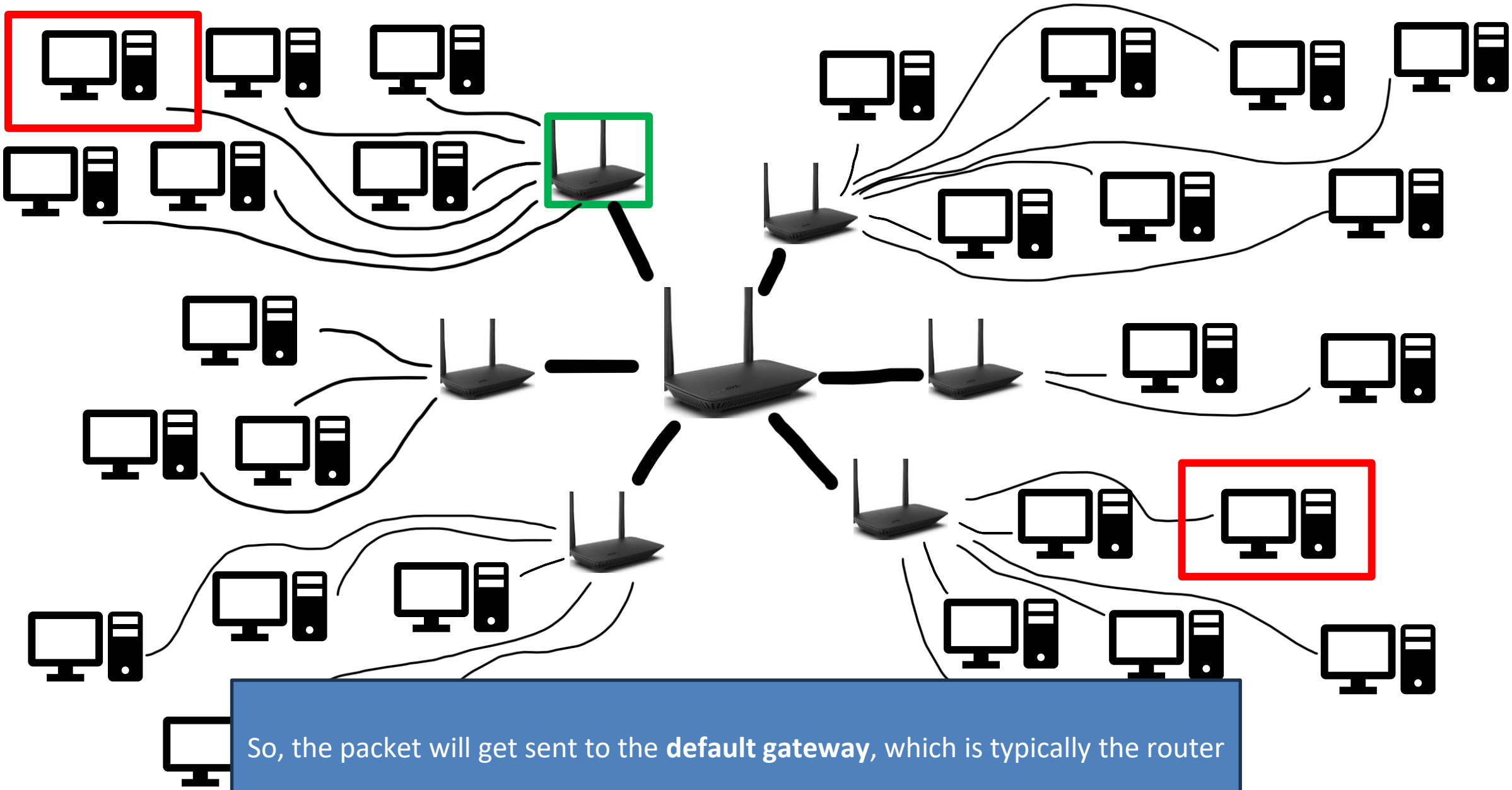

Six Subnets


Six Subnets

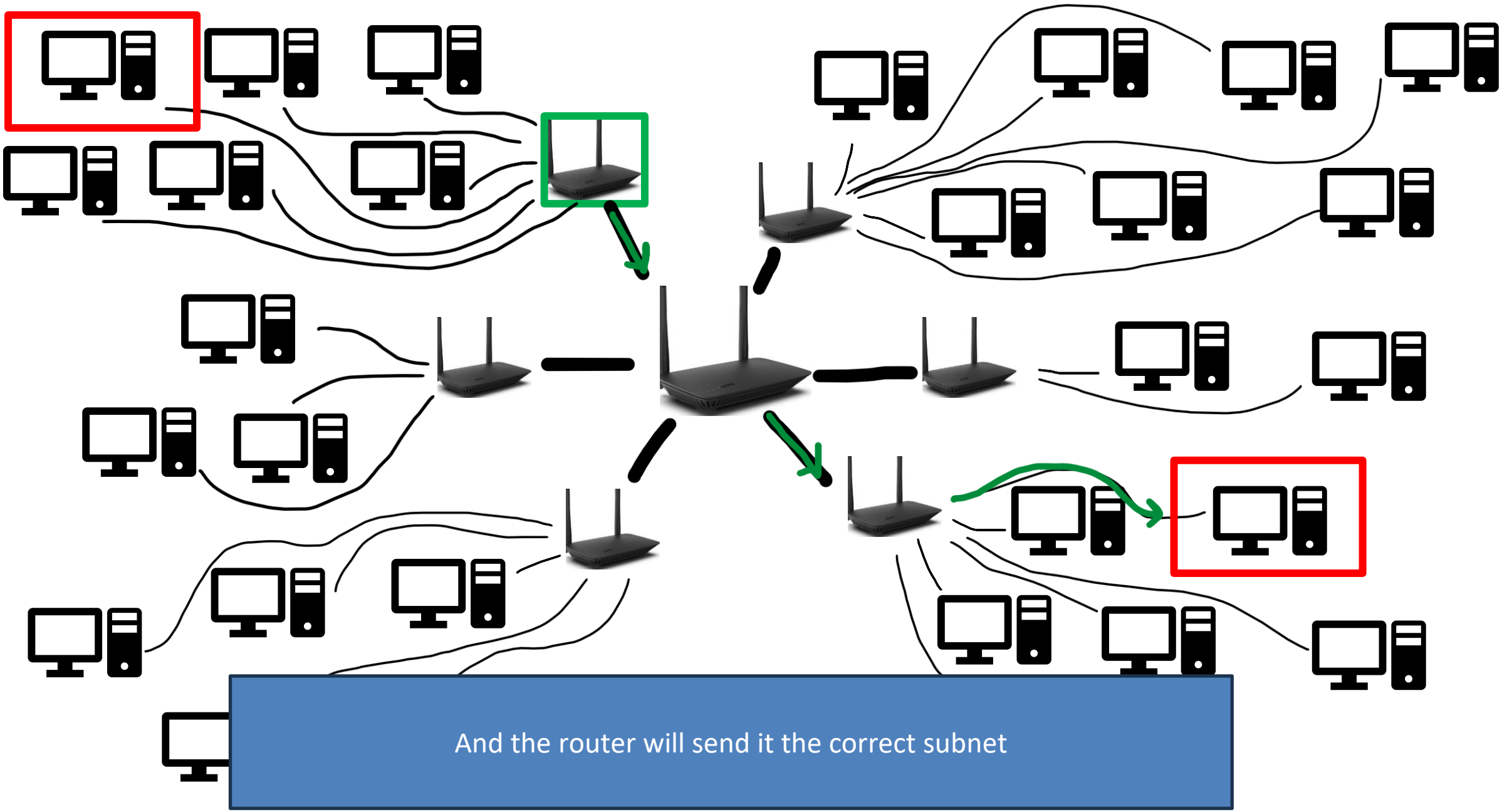

Suppose that a packet needs to be sent to another subnet



We broadcast to our subnet, but we won't find our destination



So, the packet will get sent to the **default gateway**, which is typically the router


And the router will send it the correct subnet

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

What if we wanted to divide this network into multiple subnets?

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

What if we wanted to divide this network into multiple subnets?

We sacrifice one of our host bits, and give it to the network bits

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

What if we wanted to divide this network into multiple subnets?

We sacrifice one of our host bits, and give it to the network bits

Subnet 1: 11111111.11111111.11111111.00000000

Subnet 2: 11111111.11111111.11111111.10000000

2 Networks with 126 hosts

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

What if we wanted to divide this network into multiple subnets?

We sacrifice one of our host bits, and give it to the network bits

Subnet 1: 11111111.11111111.11111111.00000000

Subnet 2: 11111111.11111111.11111111.10000000

Subnet 3: 11111111.11111111.11111111.01000000

Subnet 4: 11111111.11111111.11111111.11000000

4 Networks with 62 hosts

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

*What if we wanted to divide this network into **four** subnets?*

We sacrifice one of our host bits, and give it to the network bits

Subnet 1: 11111111.11111111.11111111.00000000

Subnet 2: 11111111.11111111.11111111.10000000

Subnet 3: 11111111.11111111.11111111.01000000

Subnet 4: 11111111.11111111.11111111.11000000

4 Networks with 62 hosts

Subnets

Suppose we have a **class c** network

X.X.X.0/24 **Subnet Mask** 11111111.11111111.11111111.00000000

1 network with 253* hosts

*What if we wanted to divide this network into **four** subnets?*

We sacrifice one of our host bits, and give it to the network bits

New Subnet Mask 11111111.11111111.11111111.11000000

255.255.255.192

Issue: IPv4 can only support ~4 billion devices

If every device on the internet gets an IP address, we are going to run out of IP addresses to use

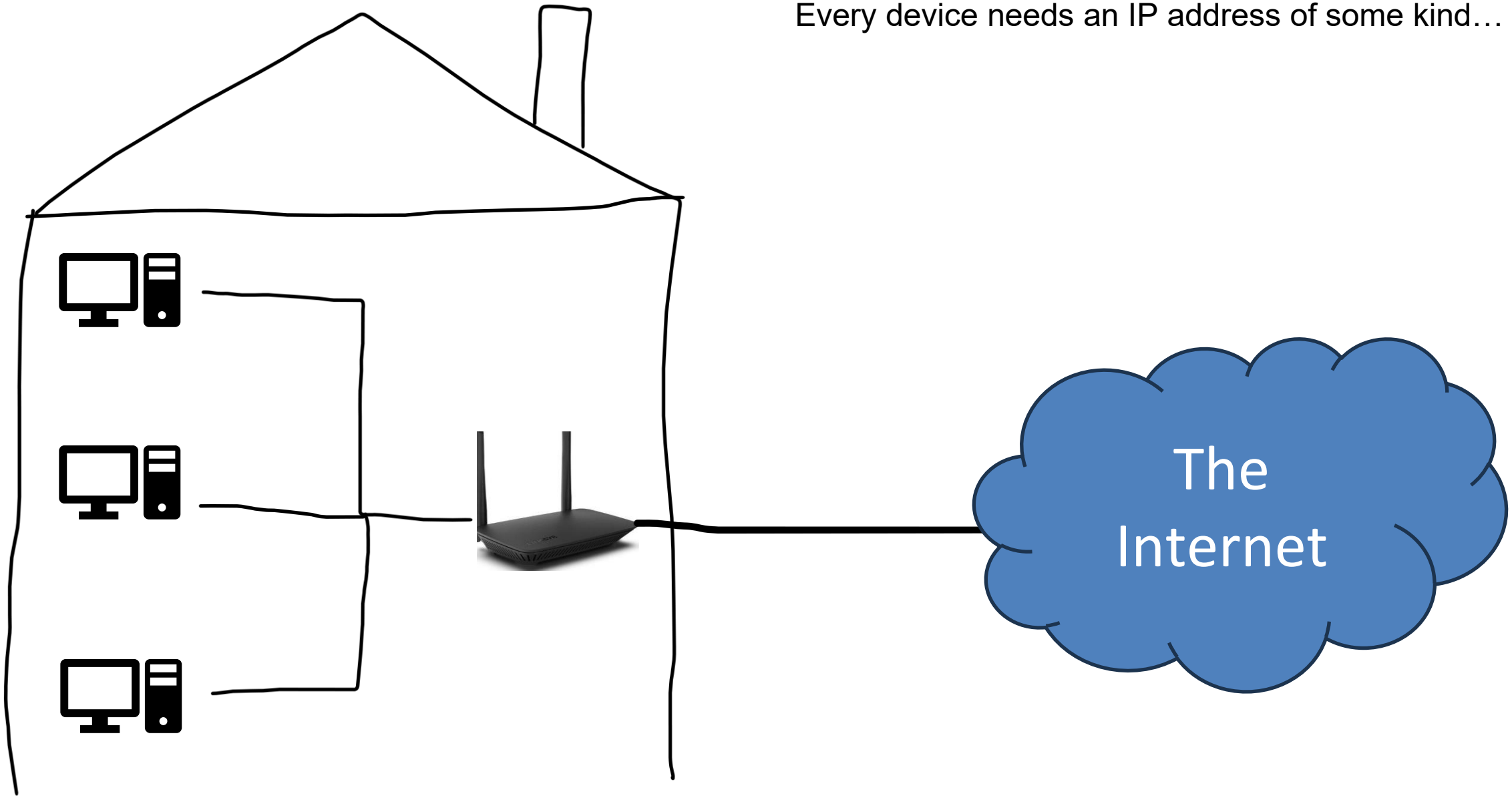
Issue: IPv4 can only support ~4 billion devices

If every device on the internet gets an IP address, we are going to run out of IP addresses to use

Solution: IPv6 (This will take awhile to implement and switch to)

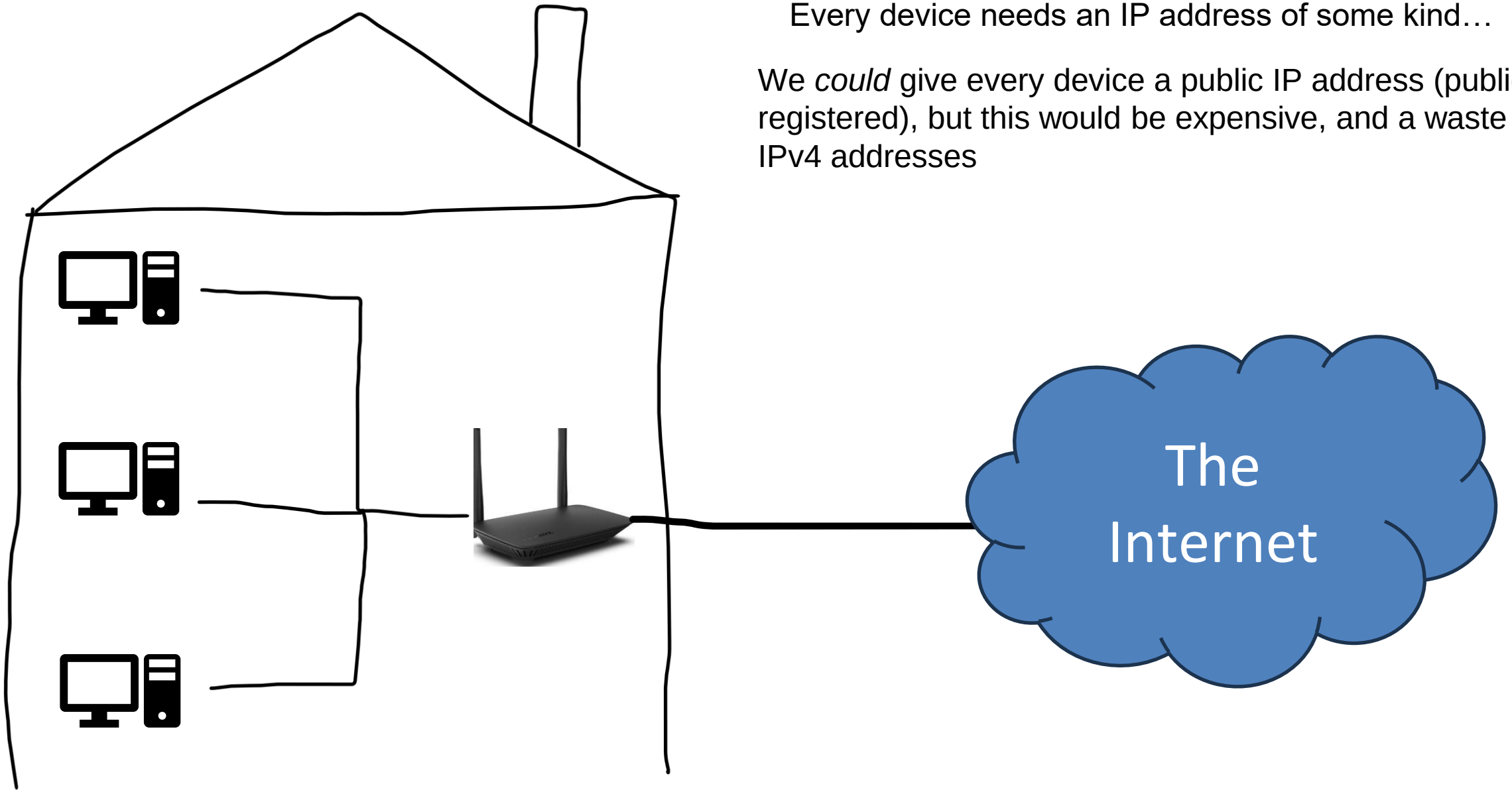
Temporary Solution: Public IP addresses, Private IP addresses, NAT

Every device needs an IP address of some kind…



Every device needs an IP address of some kind…

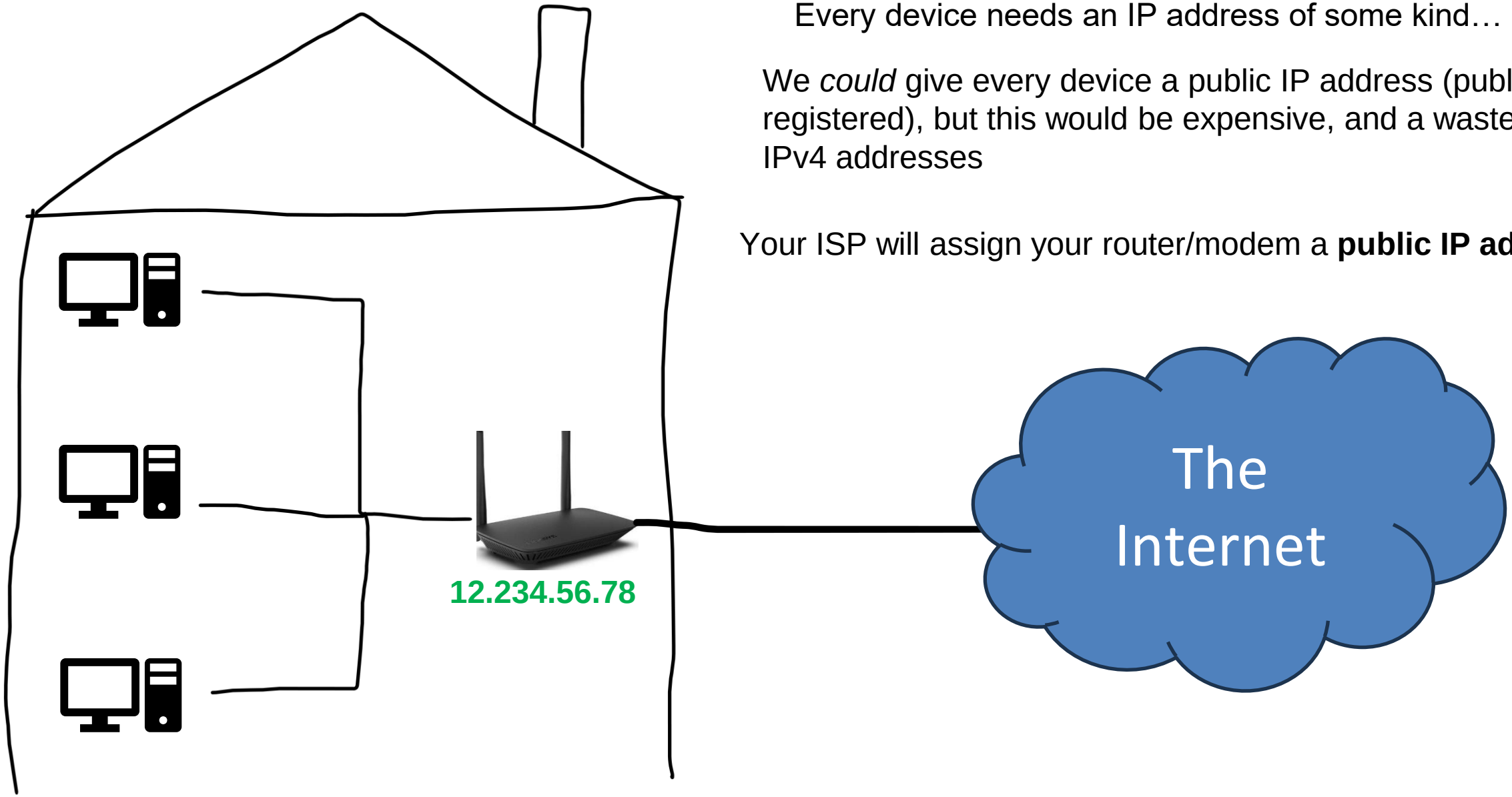
We *could* give every device a public IP address (public registered), but this would be expensive, and a waste of IPv4 addresses

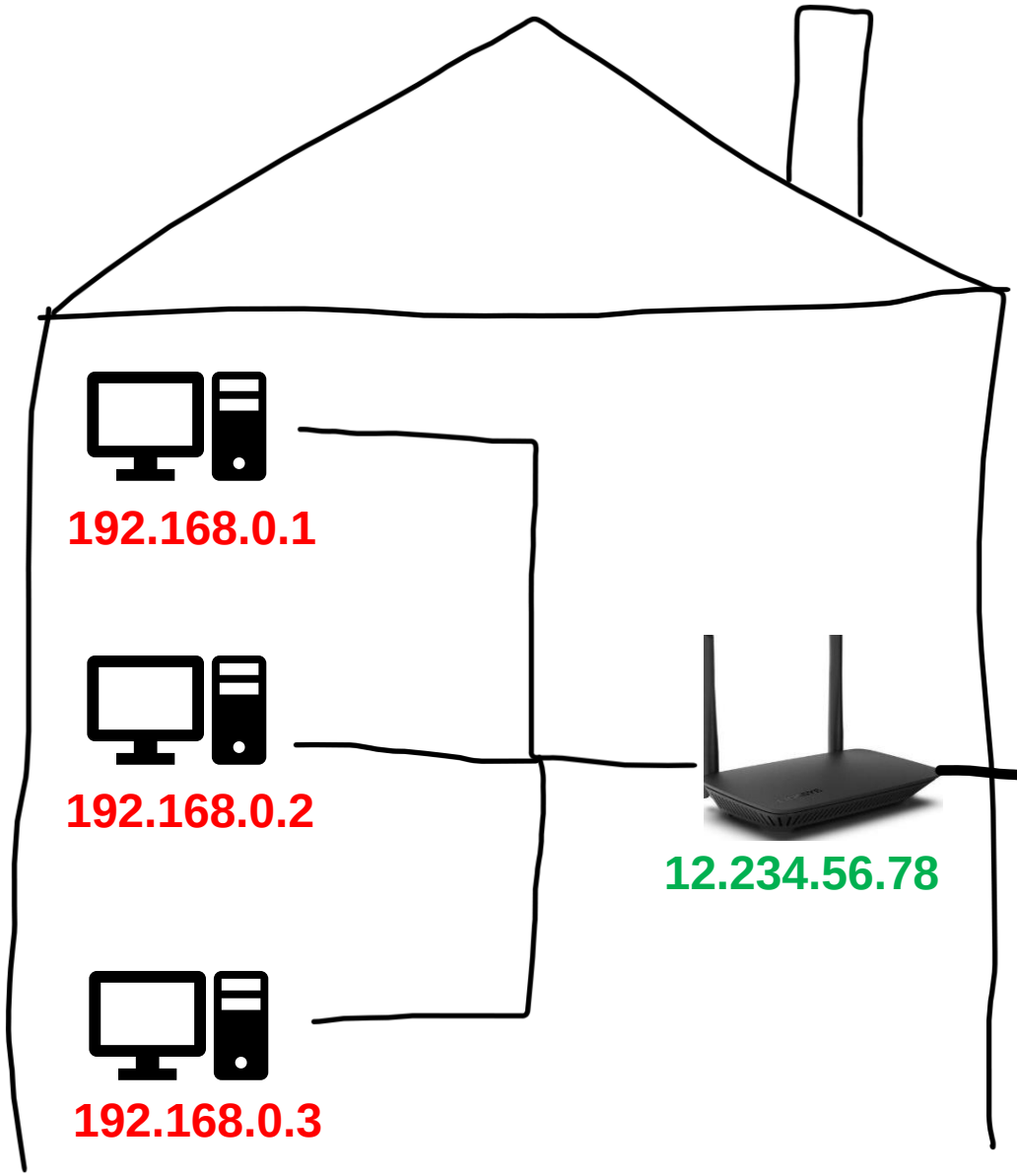


Every device needs an IP address of some kind…

We *could* give every device a public IP address (public registered), but this would be expensive, and a waste of IPv4 addresses

Your ISP will assign your router/modem a **public IP address**





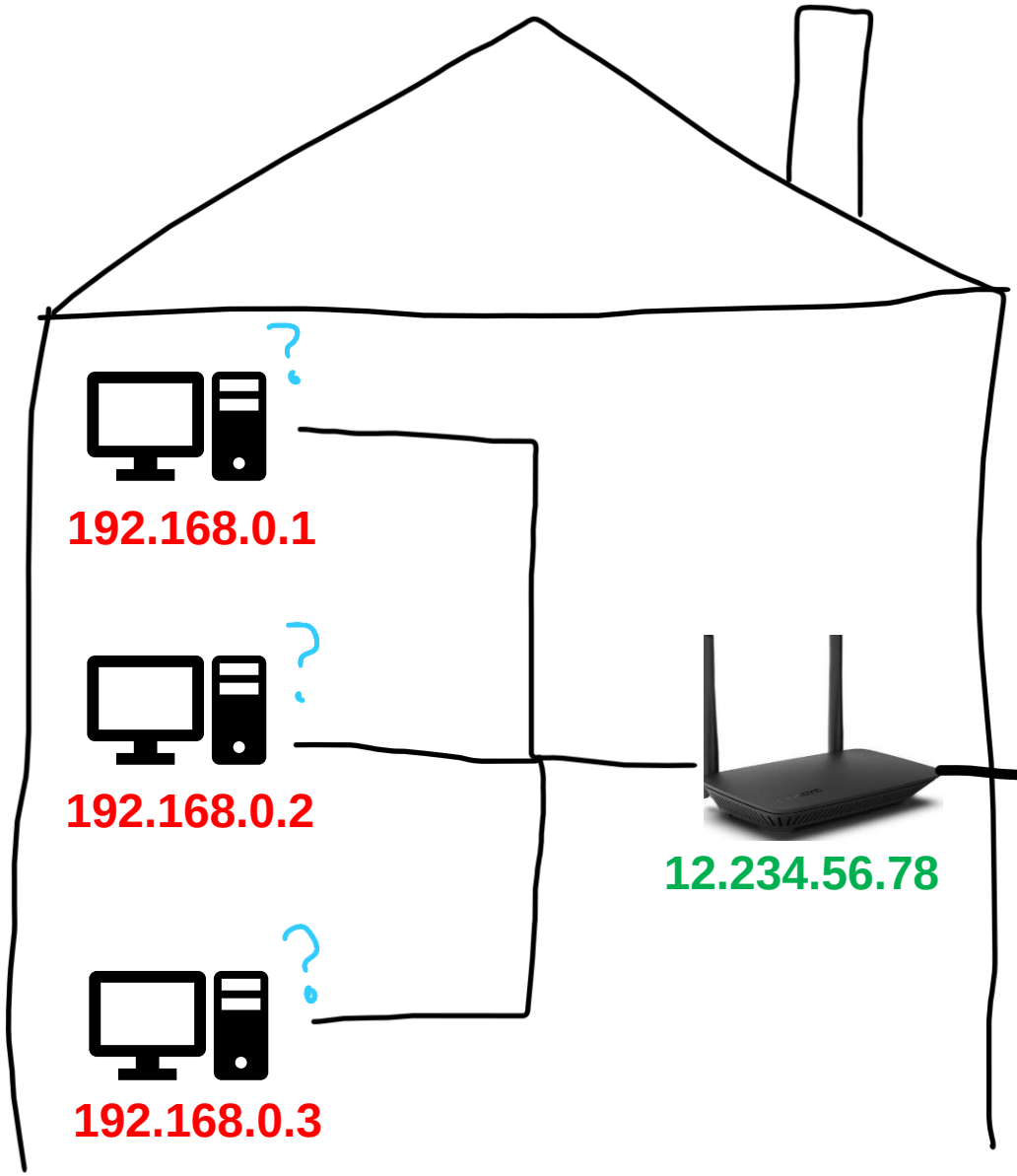
Your router will assign **private IP addresses** to each device via **DHCP**

Private IP addresses are only used internally, they are not seen by the “outside world”

Private IP addresses typically have a common prefix

The Internet

10.x.x.x
172.x.x.x
192.168.x.x

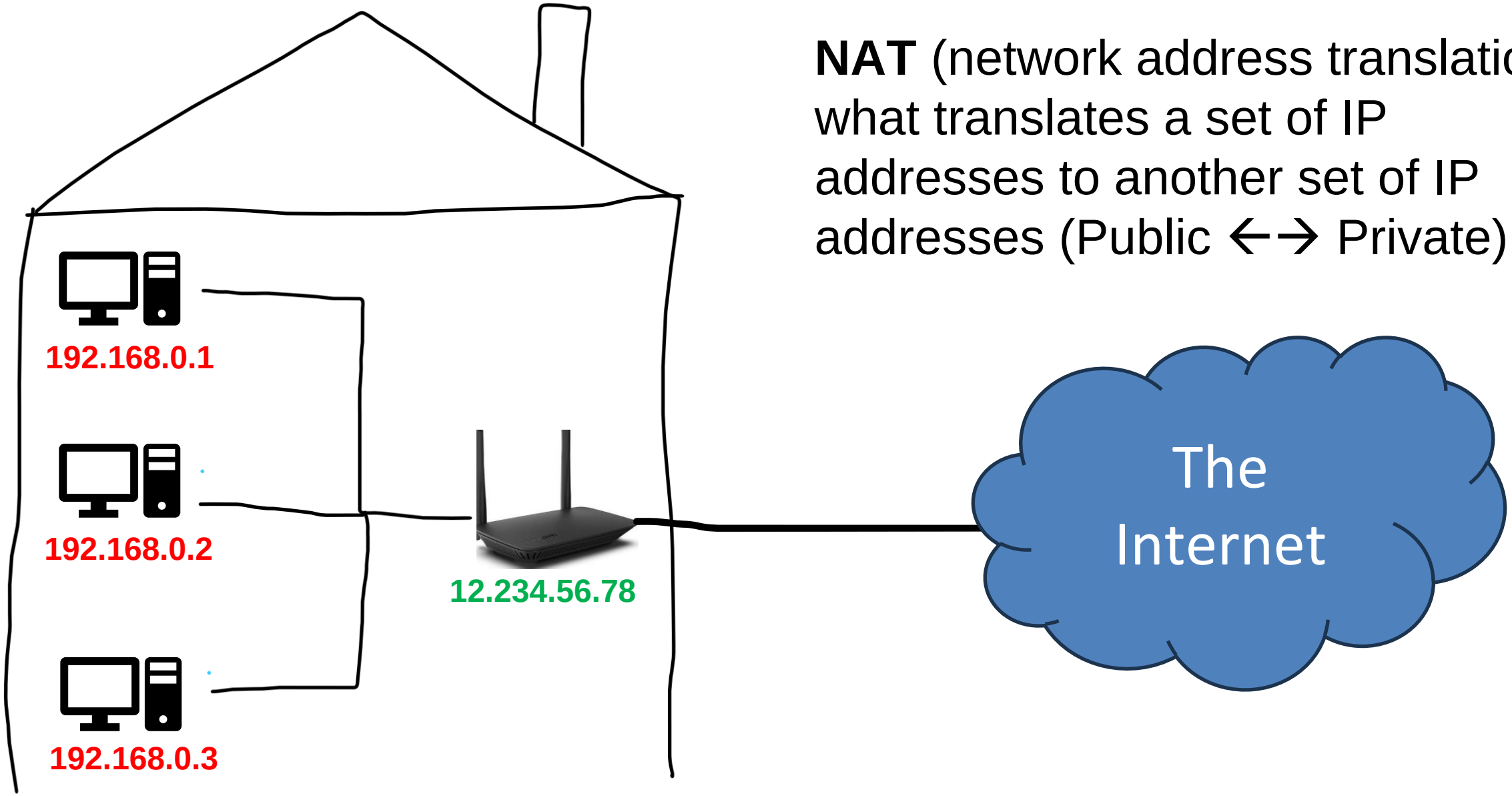


Issue: When receiving a packet with our public IP address, we need to convert to our private IP address to ensure it gets delivered to the correct host

(and vice versa)

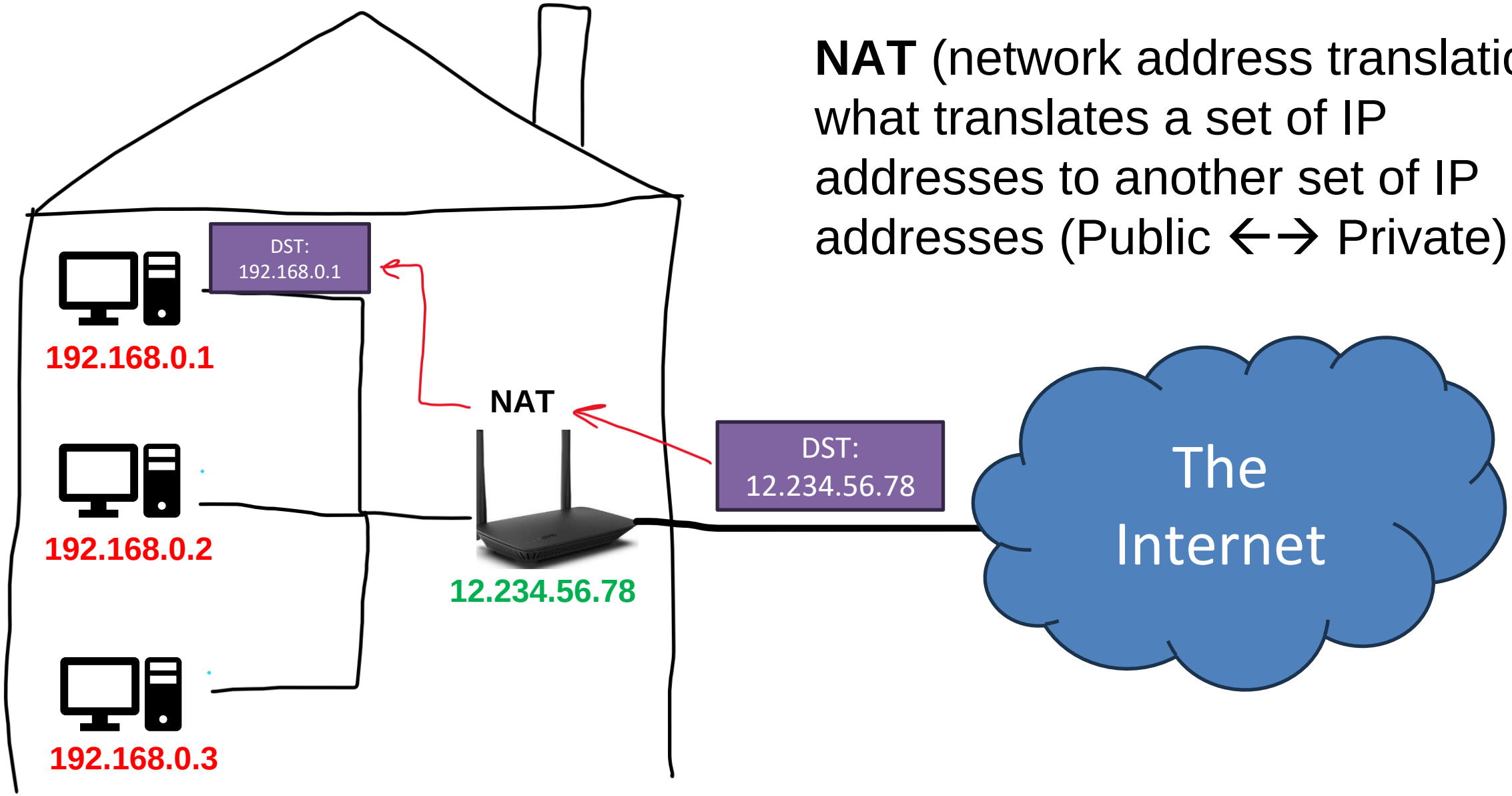
DST:
12.234.56.78

The Internet

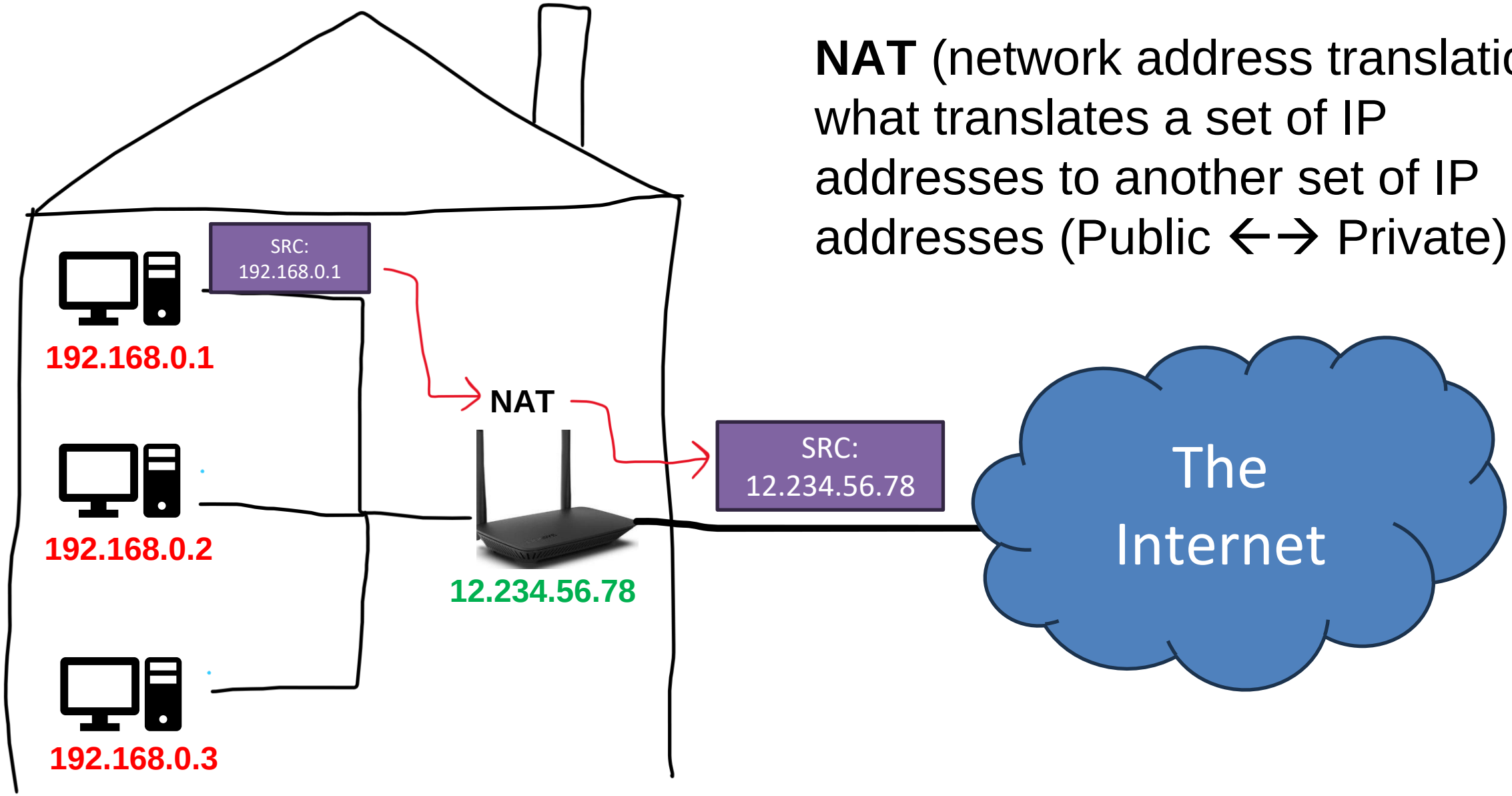


NAT (network address translation) is what translates a set of IP addresses to another set of IP addresses (Public ←→ Private)

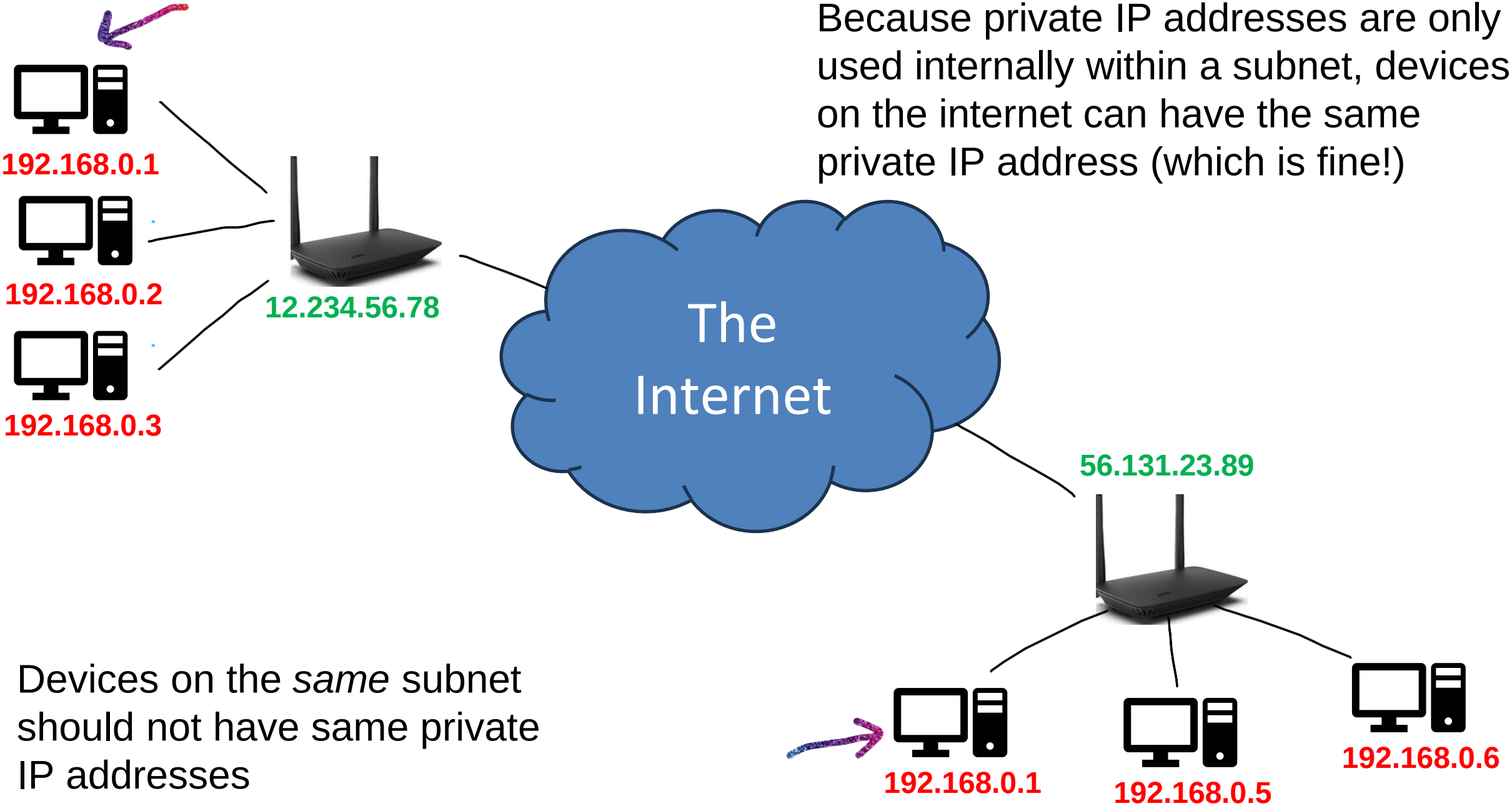
NAT (network address translation) is what translates a set of IP addresses to another set of IP addresses (Public ←→ Private)



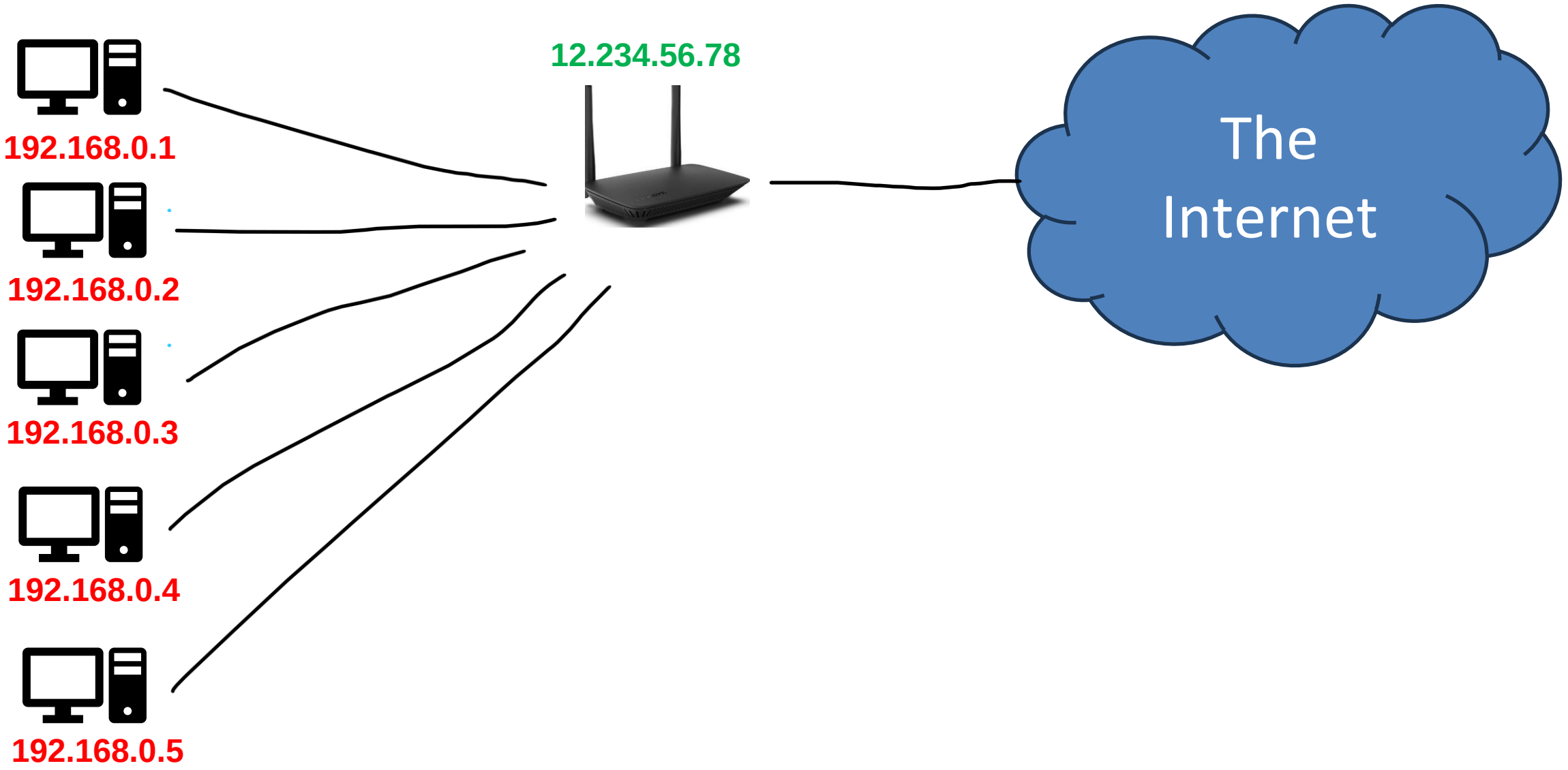
NAT (network address translation) is what translates a set of IP addresses to another set of IP addresses (Public ←→ Private)

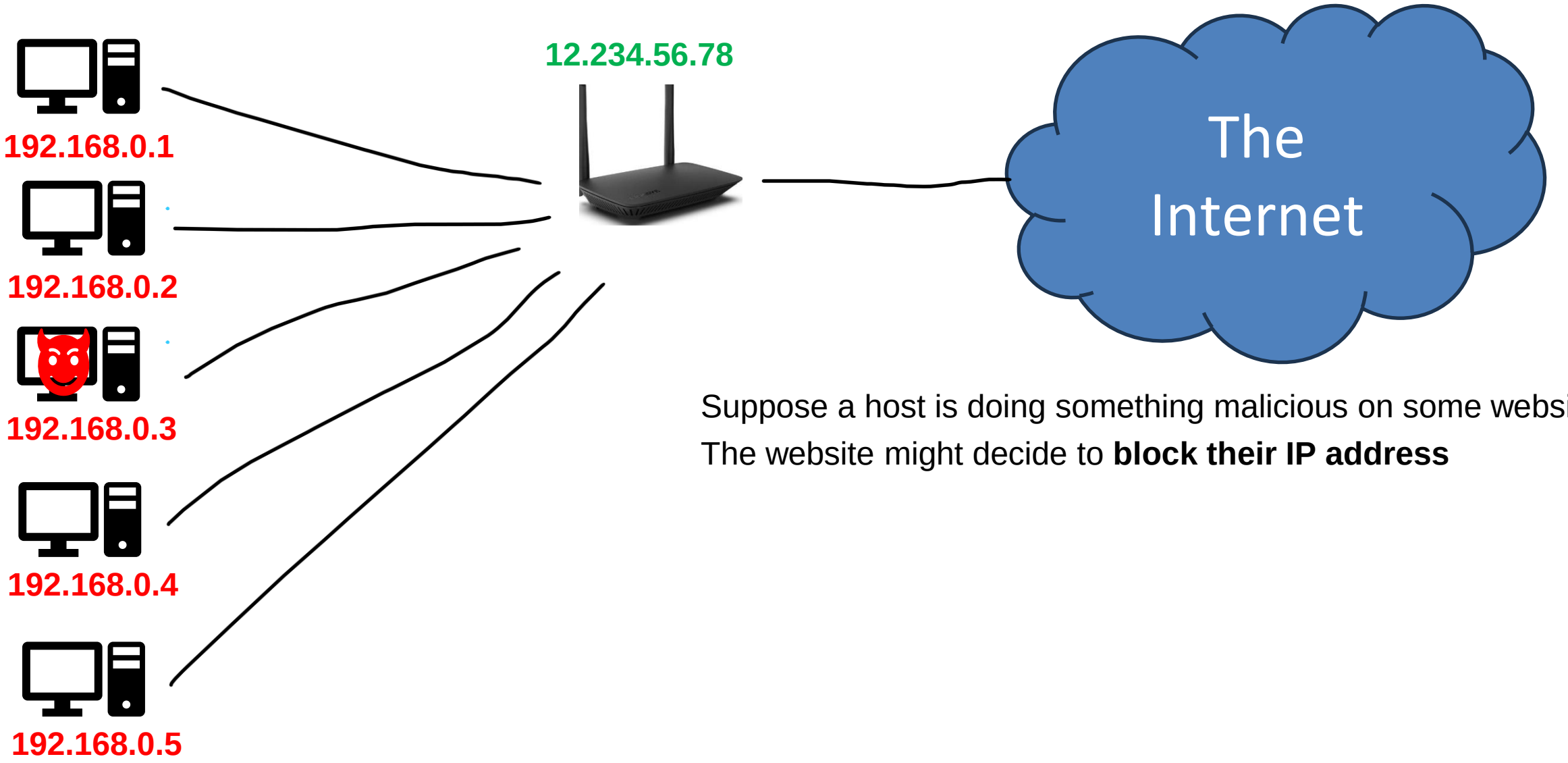


Because private IP addresses are only used internally within a subnet, devices on the internet can have the same private IP address (which is fine!)



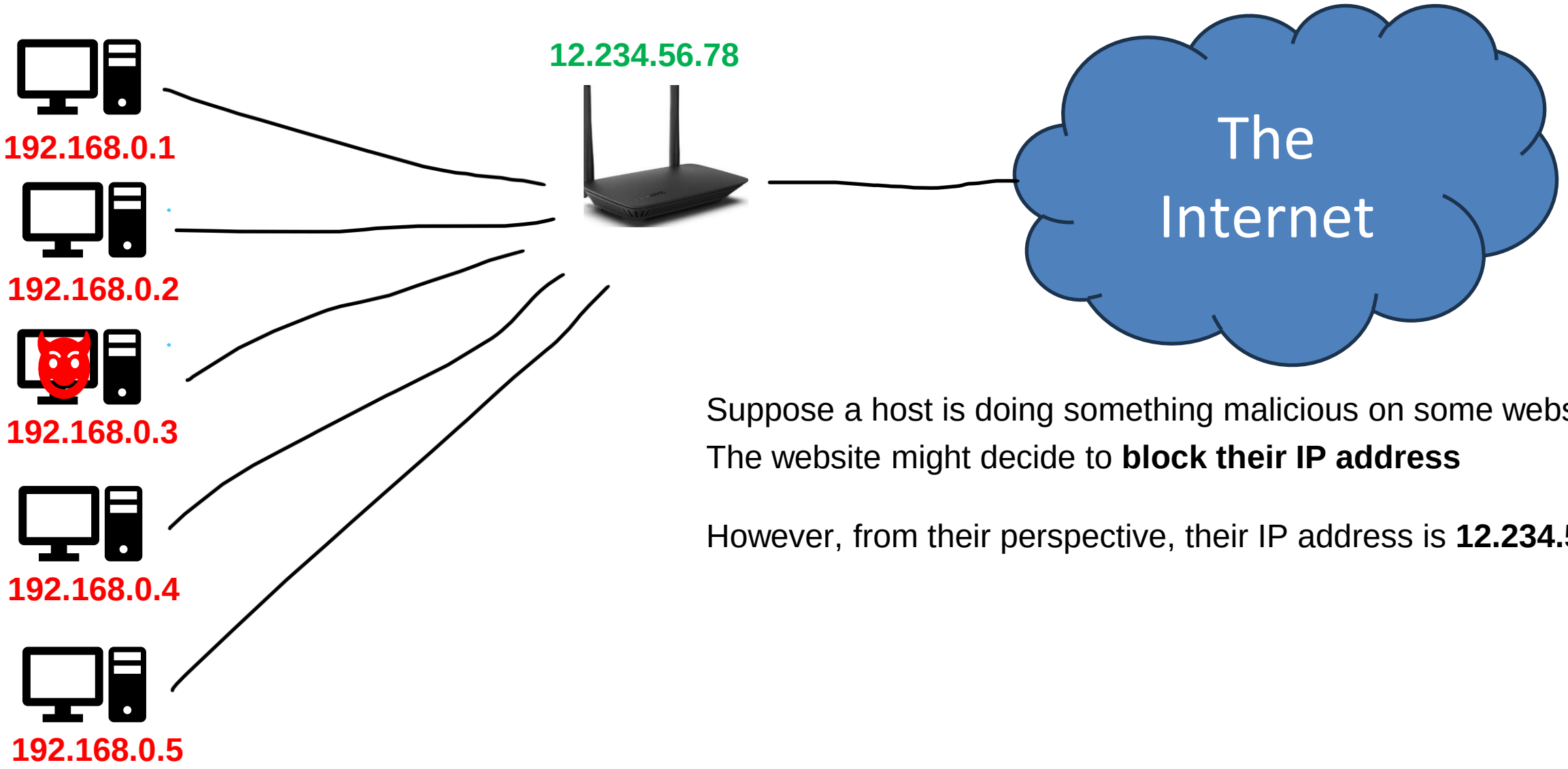
Devices on the *same* subnet should not have same private IP addresses


12.234.56.78
The Internet
192.168.0.1
192.168.0.2
192.168.0.3
192.168.0.4
192.168.0.5



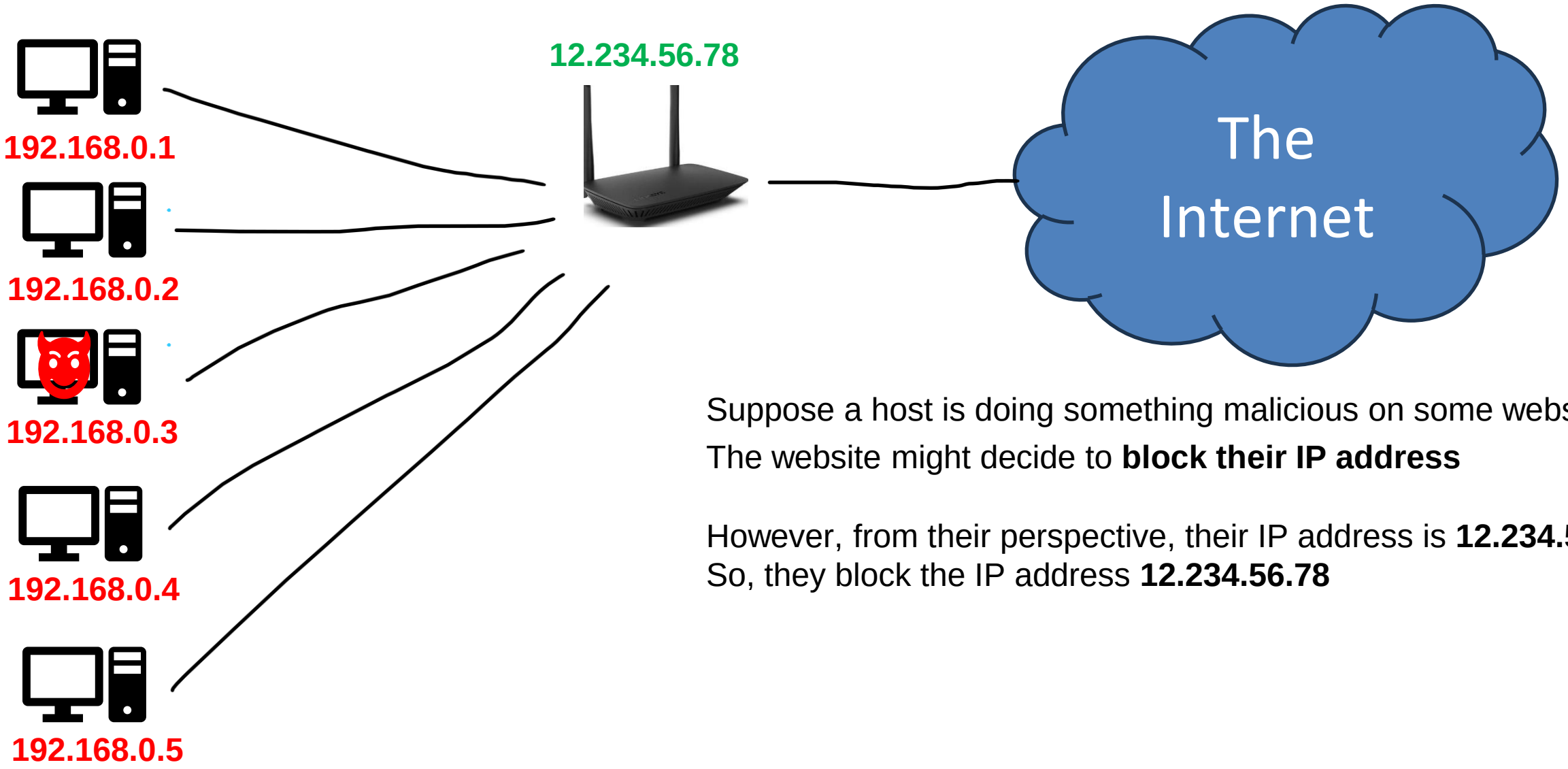
Suppose a host is doing something malicious on some website

The website might decide to **block their IP address**



Suppose a host is doing something malicious on some website
The website might decide to **block their IP address**

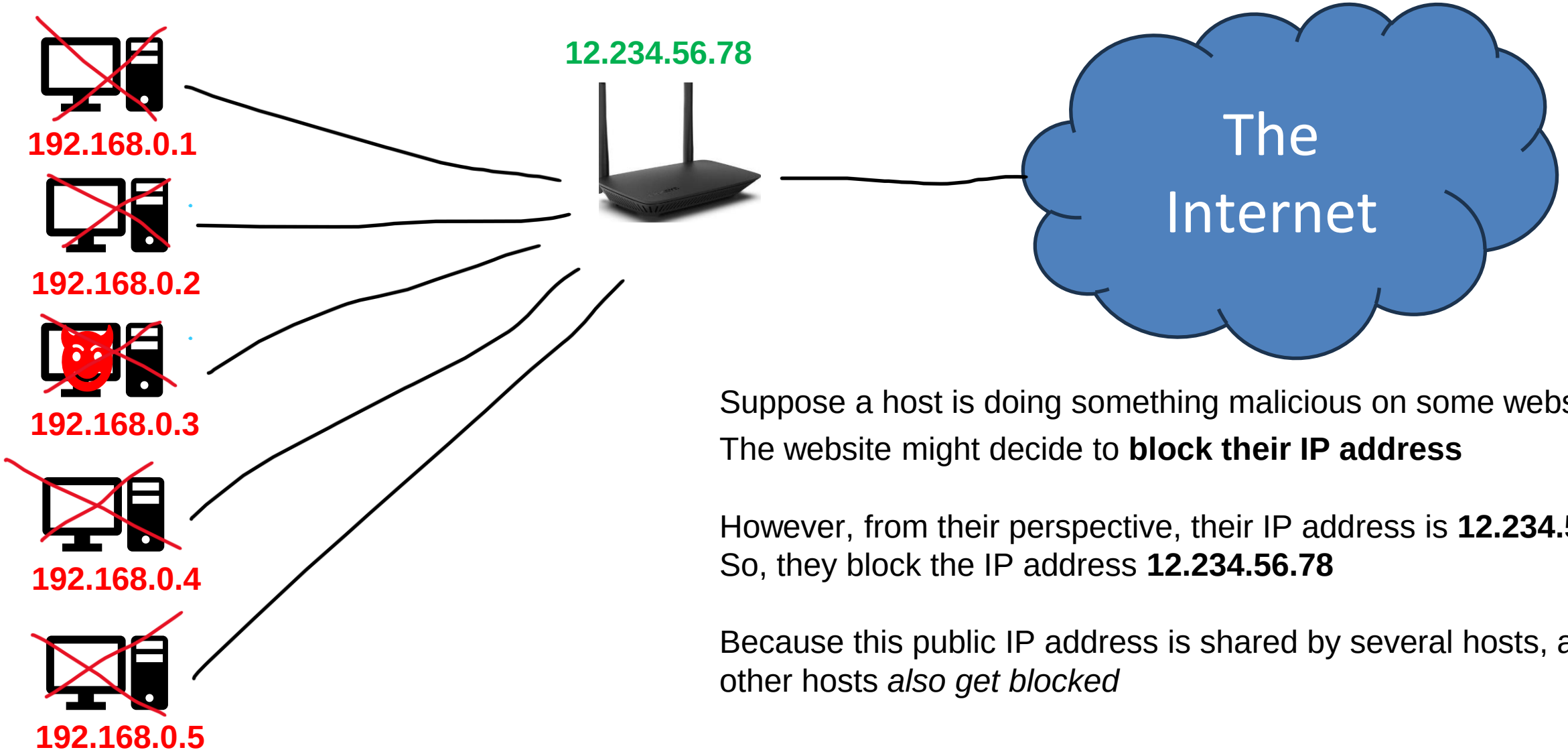
However, from their perspective, their IP address is **12.234.56.78**



Suppose a host is doing something malicious on some website

The website might decide to **block their IP address**

However, from their perspective, their IP address is **12.234.56.78**
So, they block the IP address **12.234.56.78**



Suppose a host is doing something malicious on some website
The website might decide to **block their IP address**

However, from their perspective, their IP address is **12.234.56.78**
So, they block the IP address **12.234.56.78**

Because this public IP address is shared by several hosts, all the other hosts *also get blocked*

How do IPs get obtained/assigned?

The Internet Corporation for Assigned Names and Numbers (ICANN) is responsible for managing and allocating IP address space for ISPs and organizations

(they also manage the DNS root servers!)

When an organization gets a range of IP addresses to use, how to we give assign them to devices?

Do we do it manually?

we could….

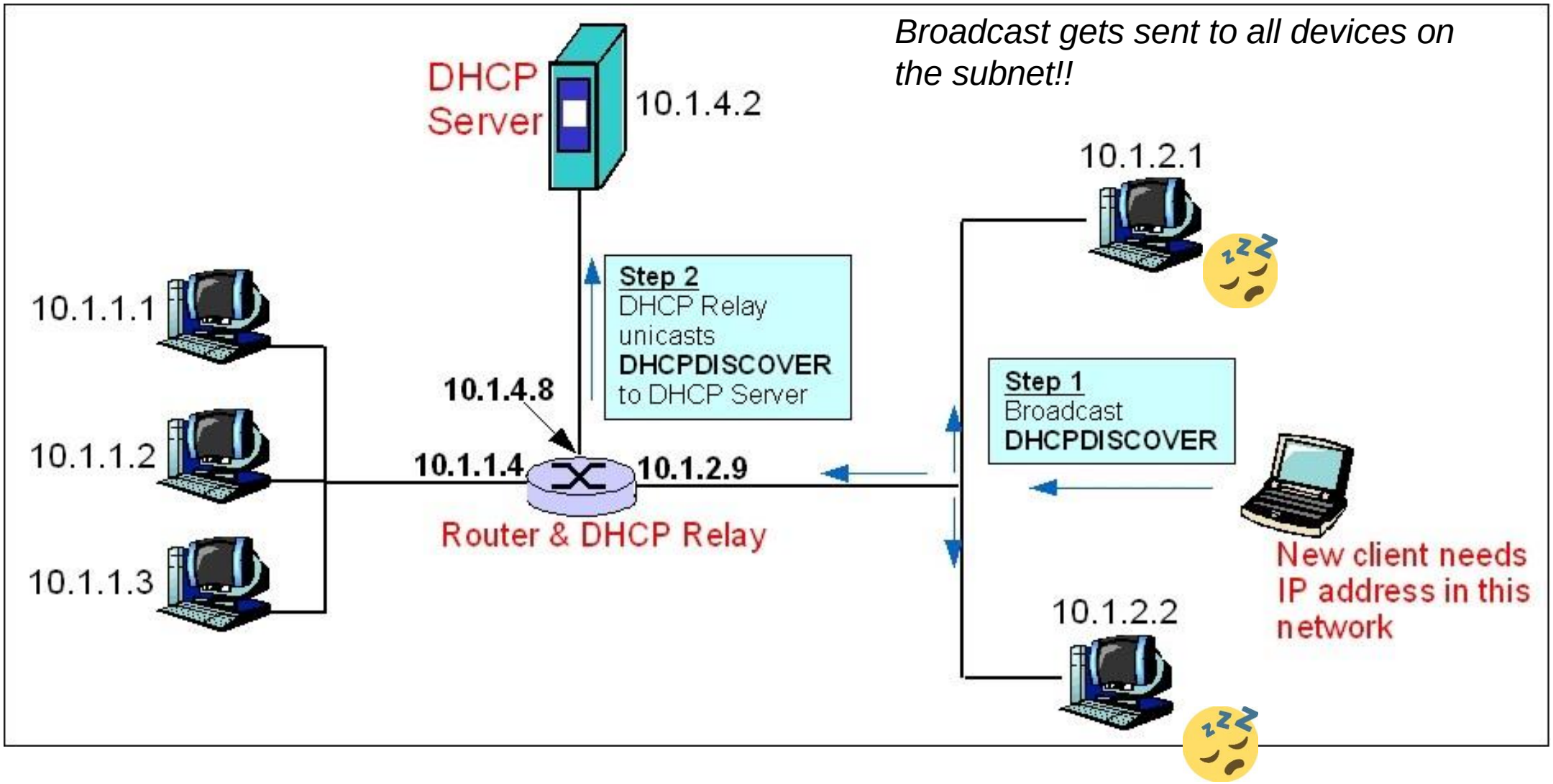
DHCP

Dynamic Host Configuration Protocol (DHCP) is a **plug-and-play**, client-server protocol that allows a host to obtain an IP address automatically

When a host is automatically assigned an IP address, it might keep that one forever, or the IP addresses can be temporary

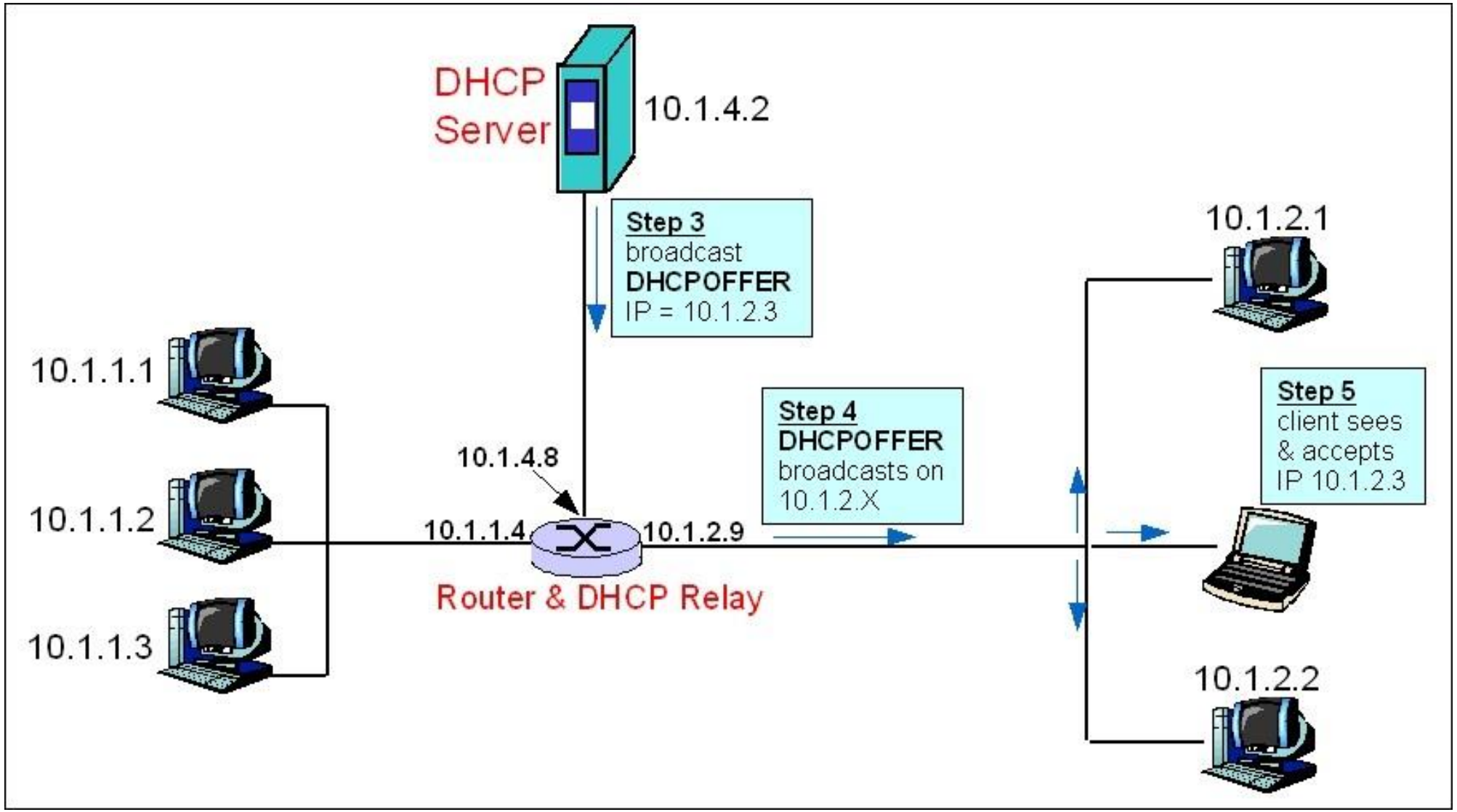
(more common)

This process is similar to a TCP handshake!

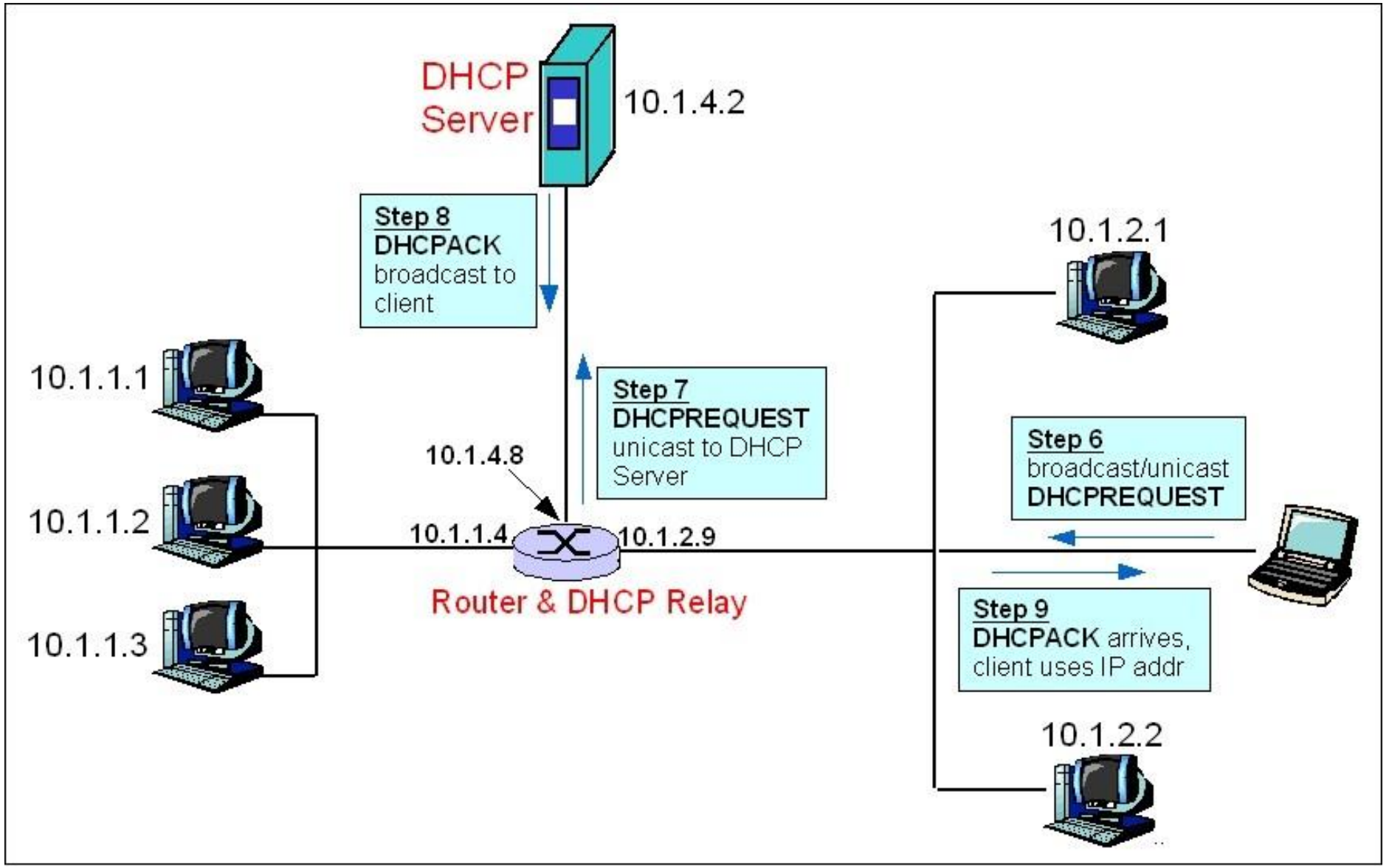


But devices that are **not** the dhcp server will ignore it

This process is similar to a TCP handshake!



This process is similar to a TCP handshake!



IP/DHCP in Wireshark