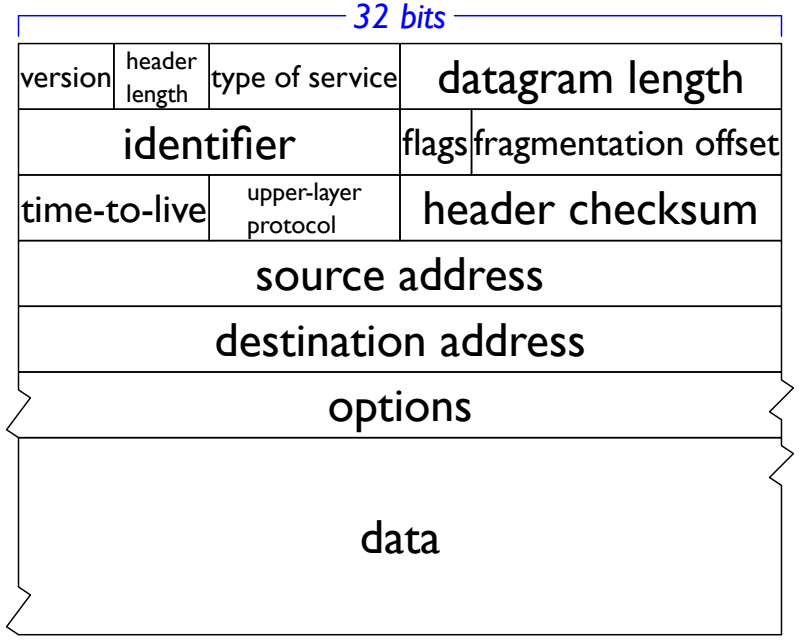
Network Layer

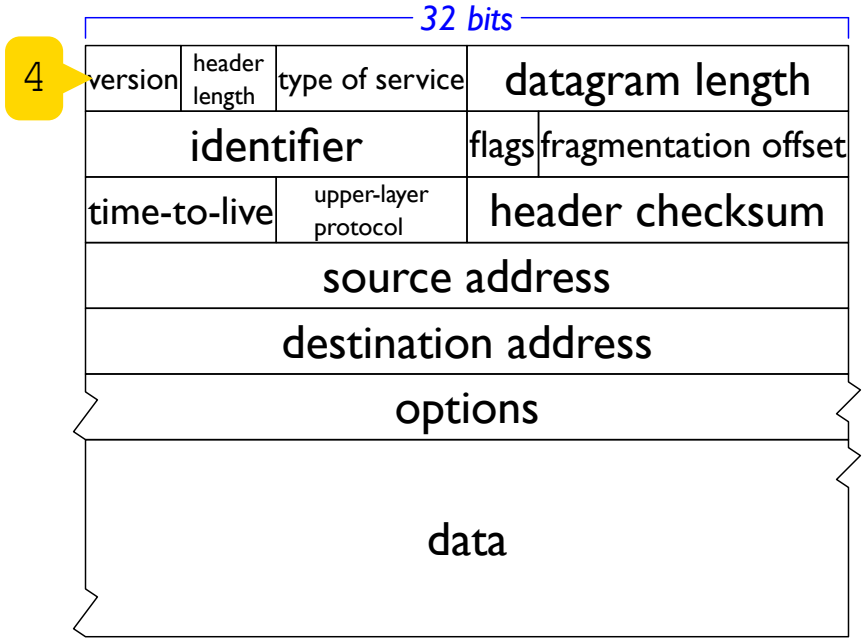
Data plane: what packets look like

Control plane: how packets are routed

IPv4 Packet Format

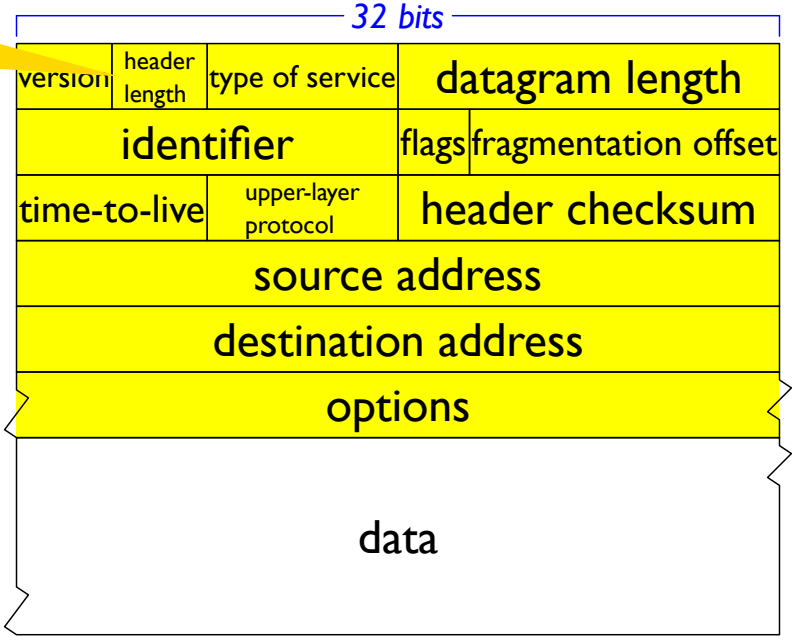


IPv4 Packet Format



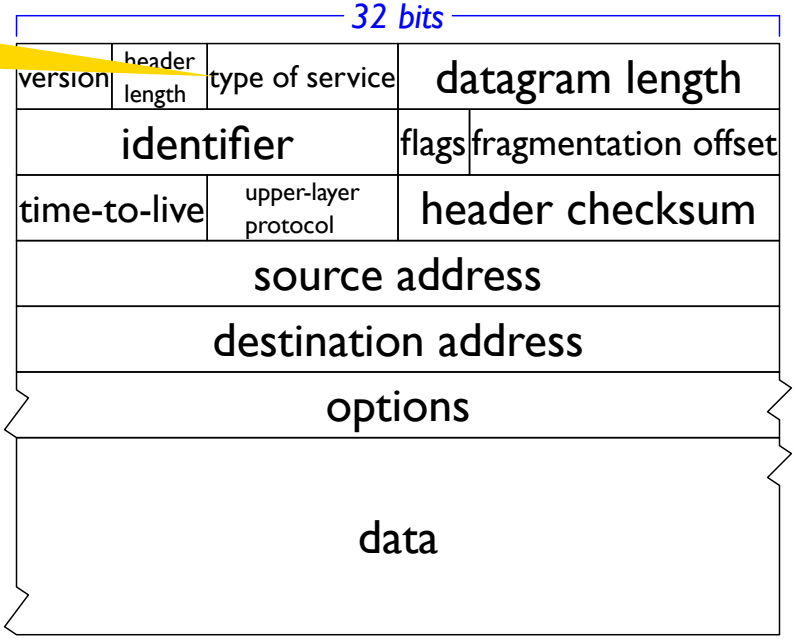
IPv4 Packet Format

In 32-bit words, so 5 means 20 bytes

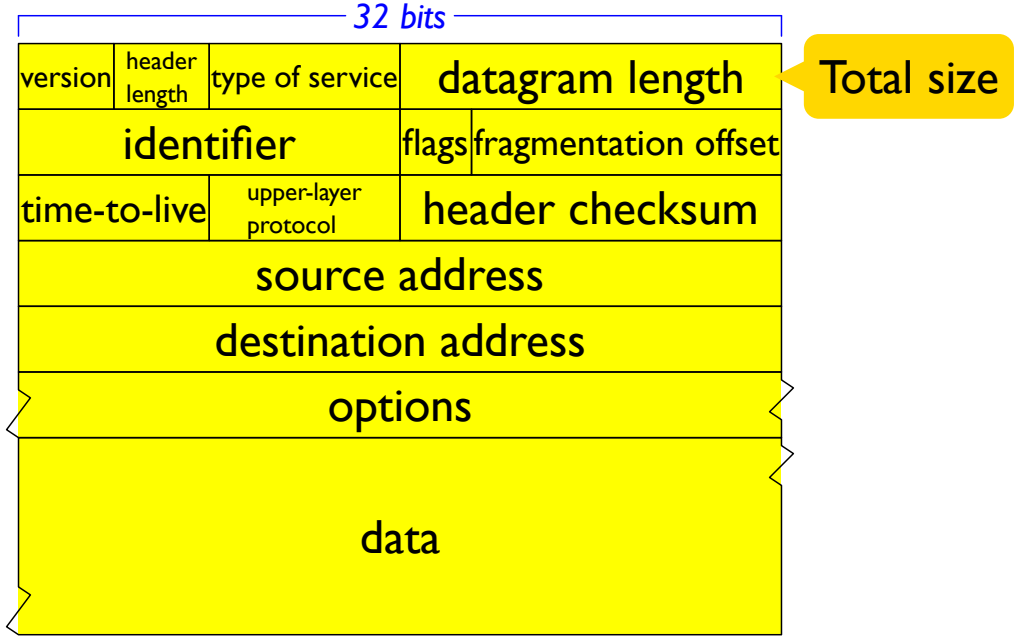


IPv4 Packet Format

Can be used to prioritize some traffic

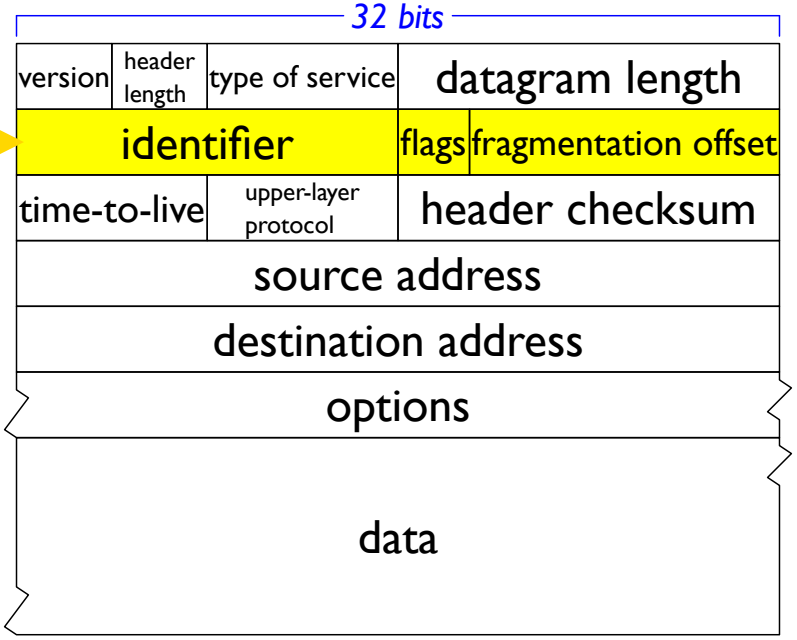


IPv4 Packet Format

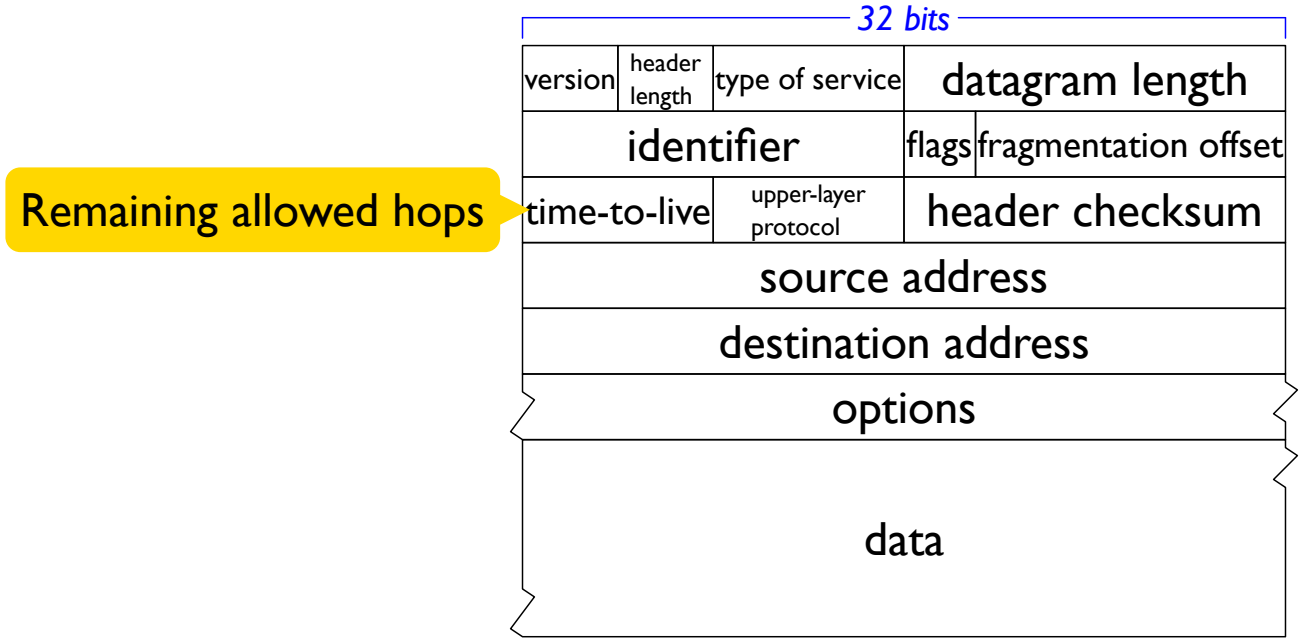


IPv4 Packet Format

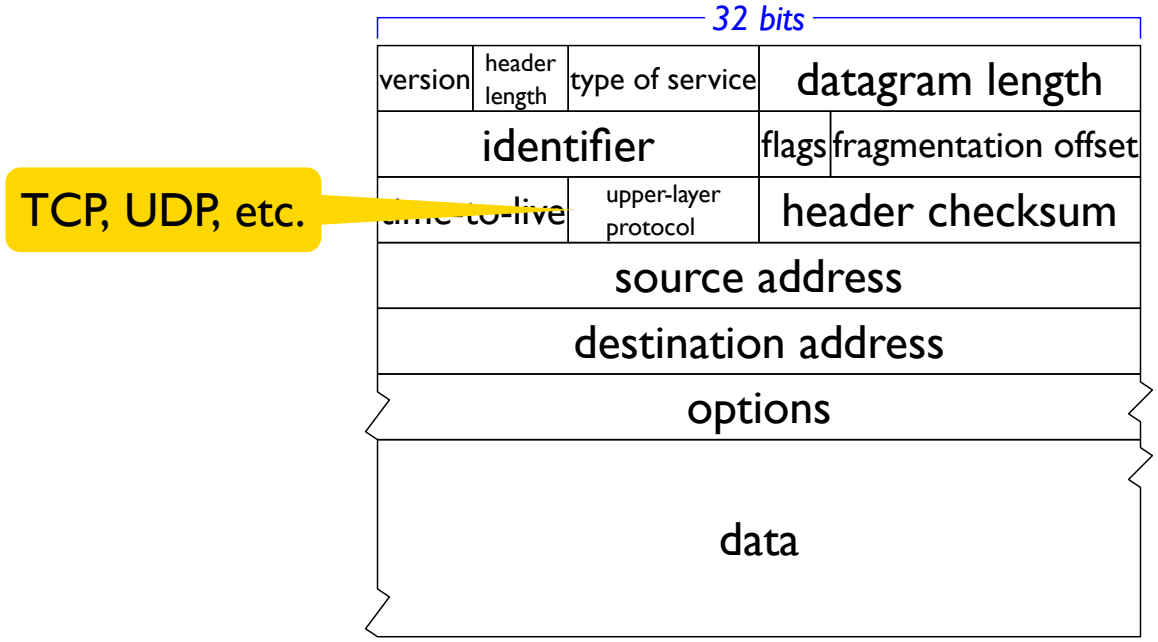
Allows splitting a packet, in case it's too large for some link layer



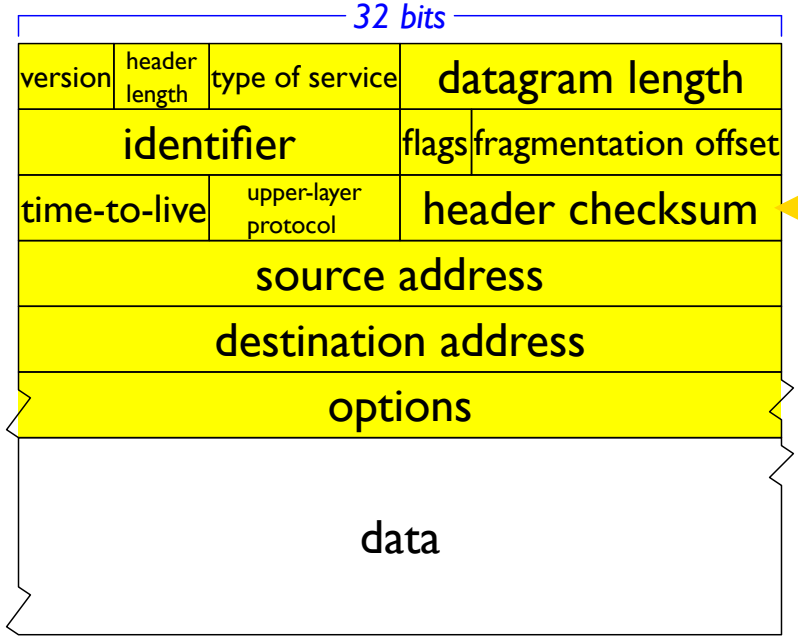
IPv4 Packet Format



IPv4 Packet Format

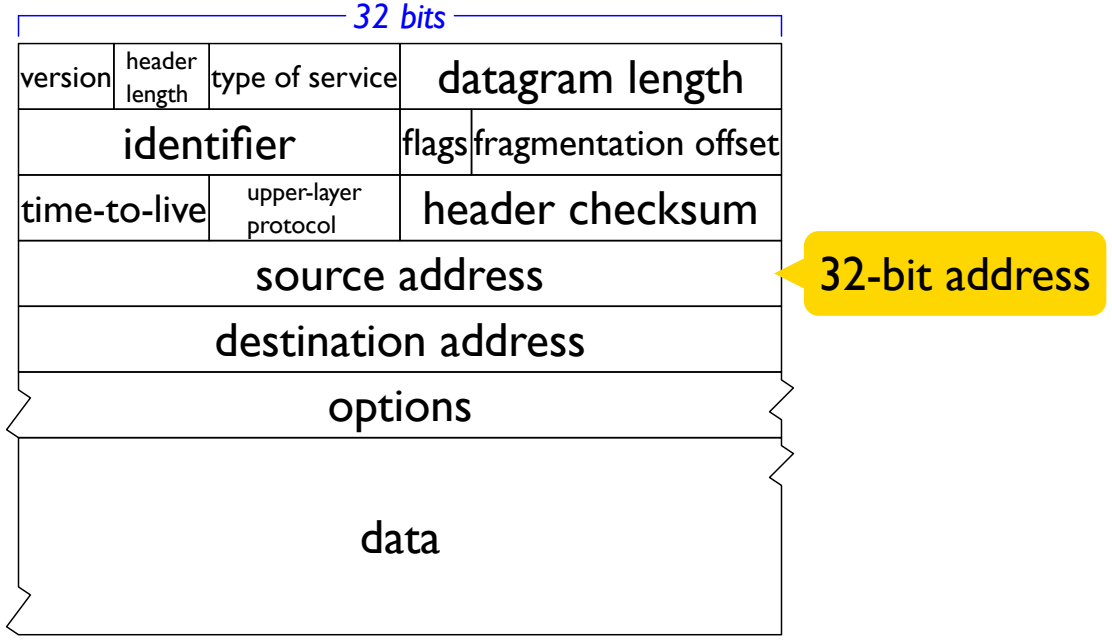


IPv4 Packet Format

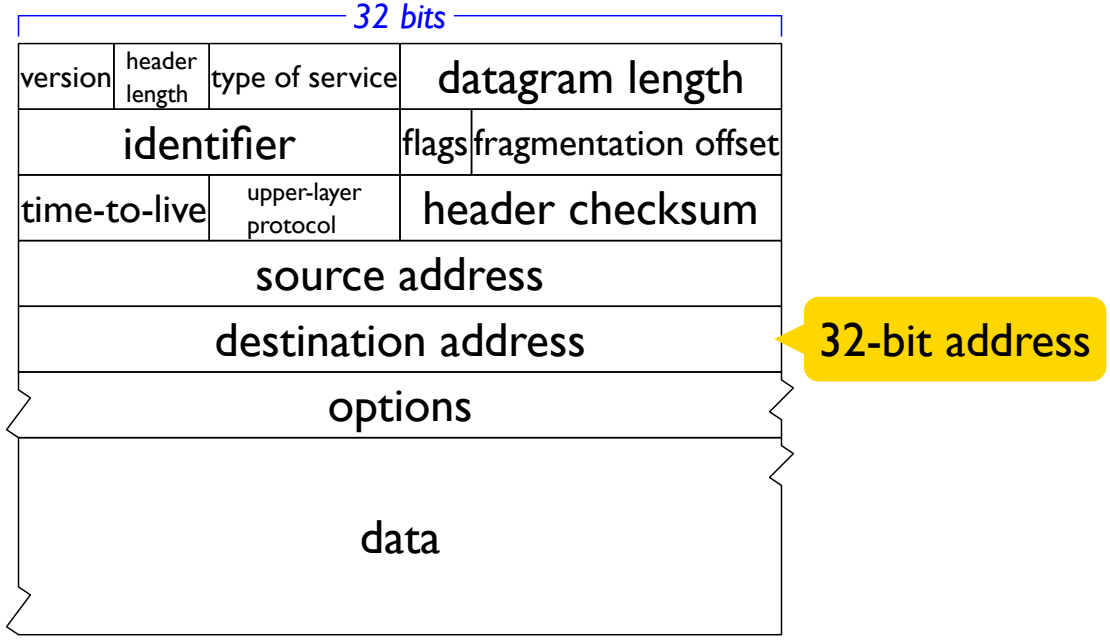


Like TCP, but only header

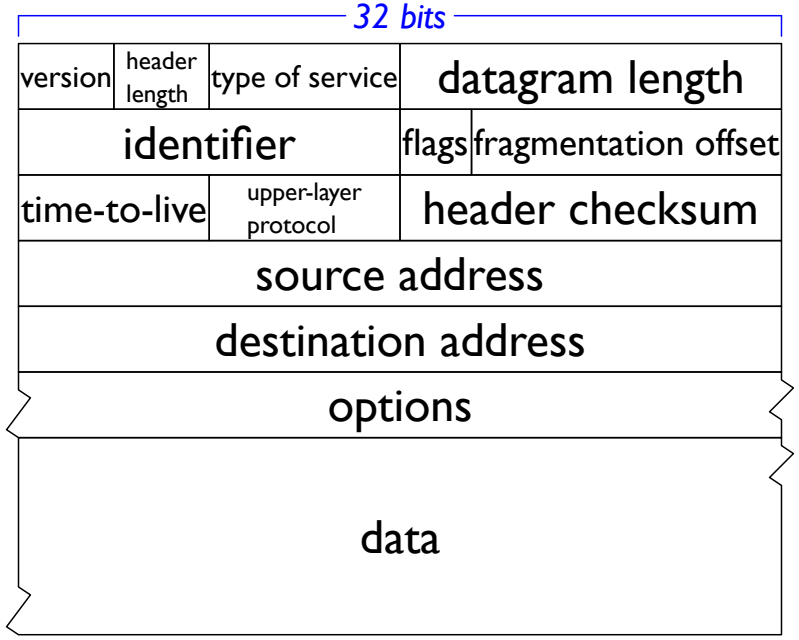
IPv4 Packet Format



IPv4 Packet Format

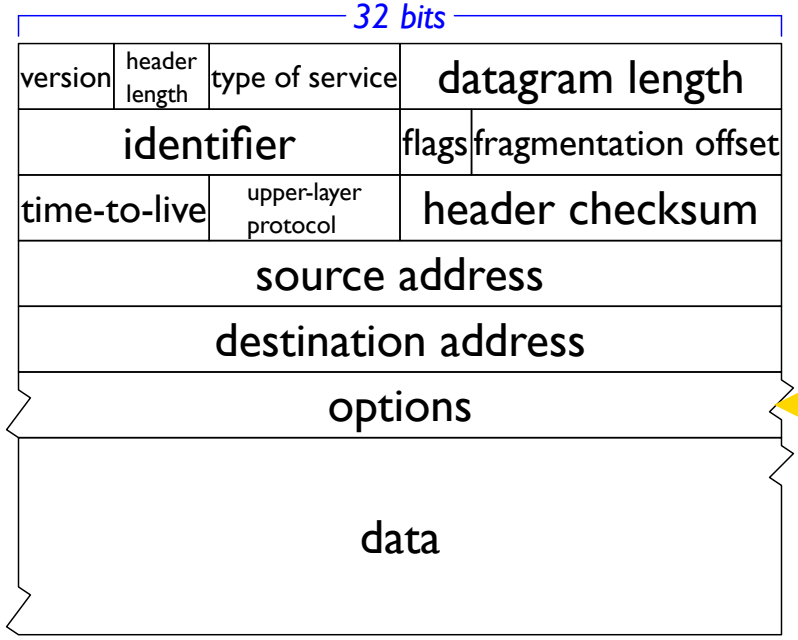


IPv4 Packet Format



Note: no port numbers

IPv4 Packet Format



Optional options field (complicates processing)

Dealing with IPv4's Limited Address Space

There are enough IPv4 addresses that each U.S. home, say, can have one

There are not enough for each *device* in each U.S. home

Solutions:

- use a larger address space: **IPv6**
- make many IPv4 hosts look like one: **NAT**

Network Address Translation (NAT)

Network Address Translation (NAT) works around address-size limitations

IPv4 level: endpoint is a 32-bit address

TCP/UDP level: endpoint is a ⟨32-bit address, 16-bit port number⟩

Network Address Translation (NAT)

Network Address Translation (NAT) works around address-size limitations

IPv4 level: endpoint is a 32-bit address

TCP/UDP level: endpoint is a ⟨32-bit address, 16-bit port number⟩

Not enough of these

Network Address Translation (NAT)

Network Address Translation (NAT) works around address-size limitations

IPv4 level: endpoint is a 32-bit address

TCP/UDP level: endpoint is a ⟨32-bit address, 16-bit port number⟩

A **NAT router** translates TCP and UDP packets to convert

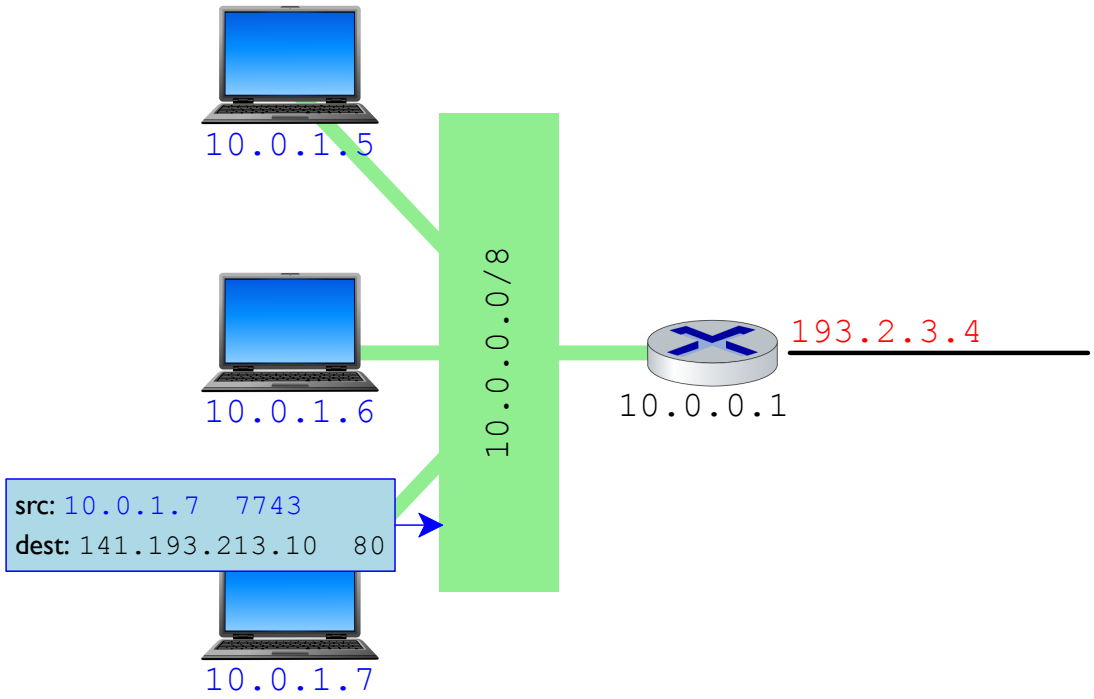
$$\langle addr_{\text{inside}}, port_{\text{inside}} \rangle$$

to

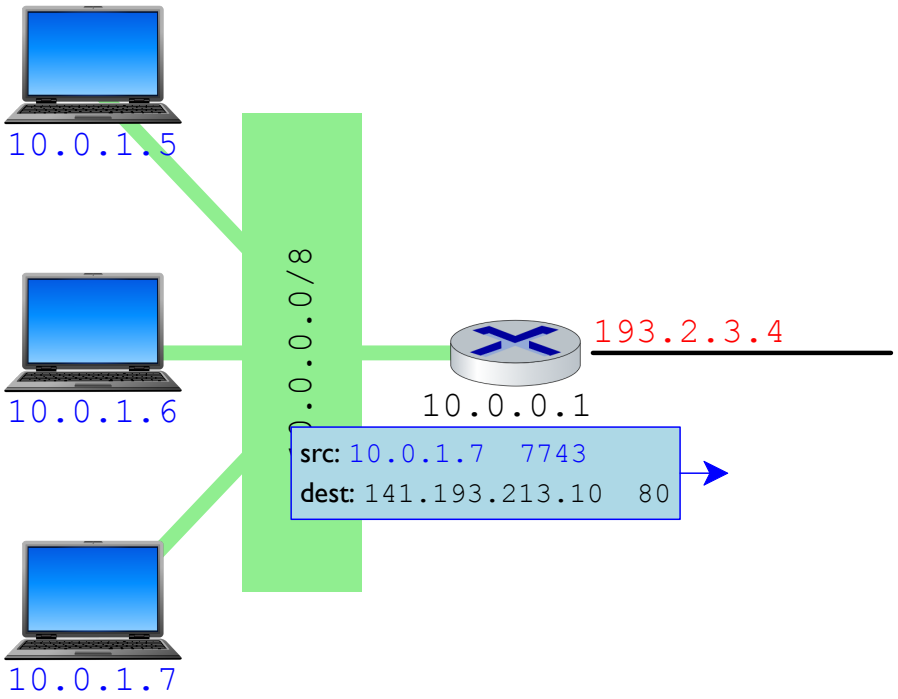
$$\langle addr_{\text{outside}}, port_{\text{outside}} \rangle$$

where $addr_{\text{outside}}$ is NAT router's own address

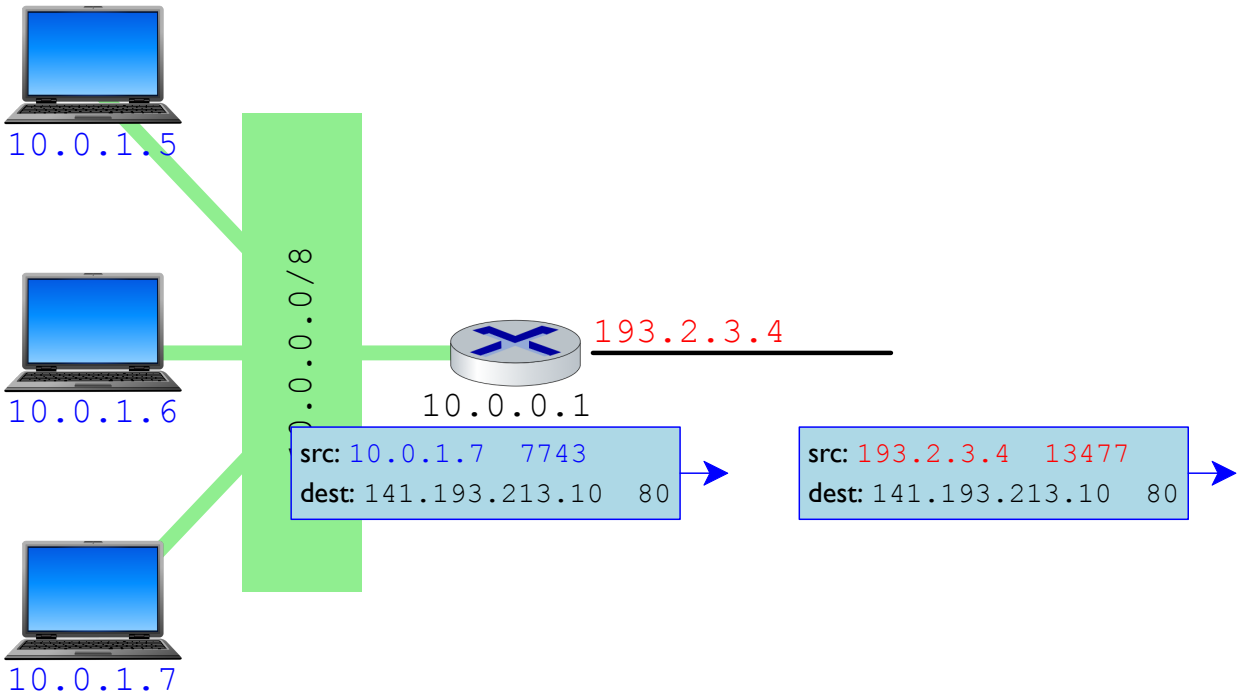
Network Address Translation (NAT)



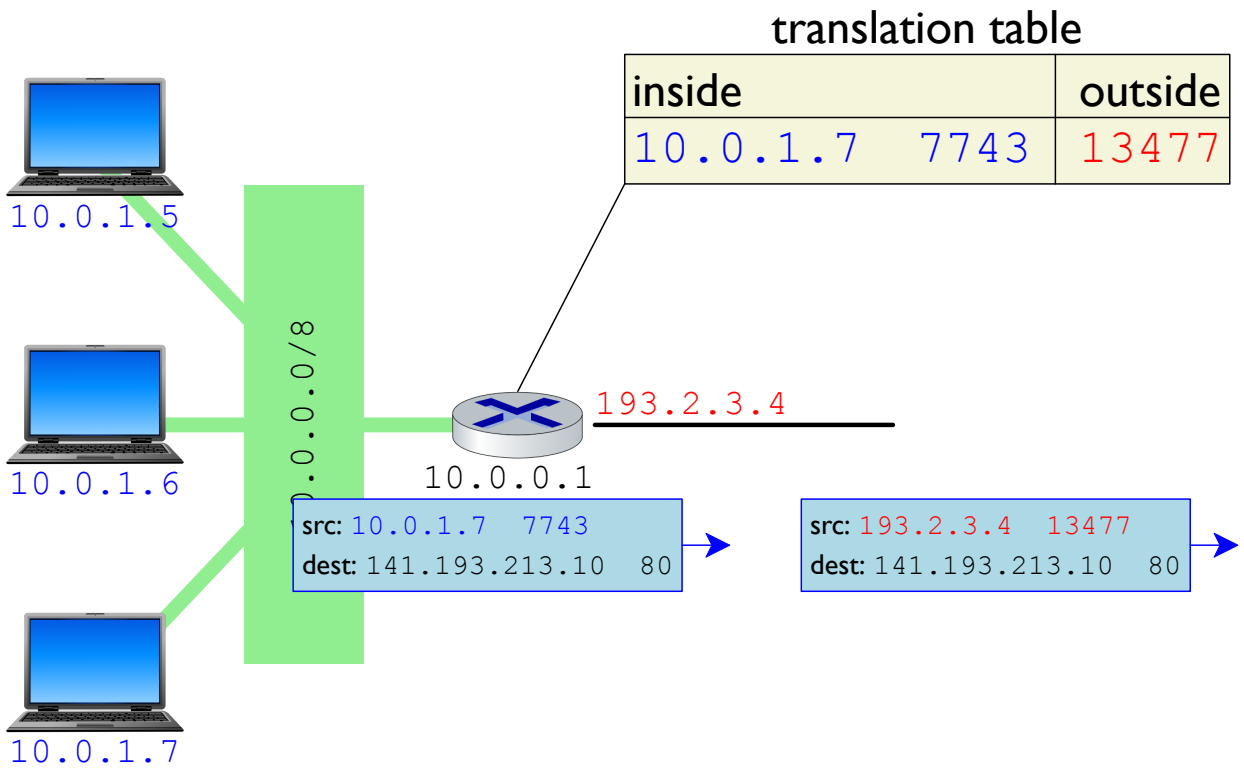
Network Address Translation (NAT)



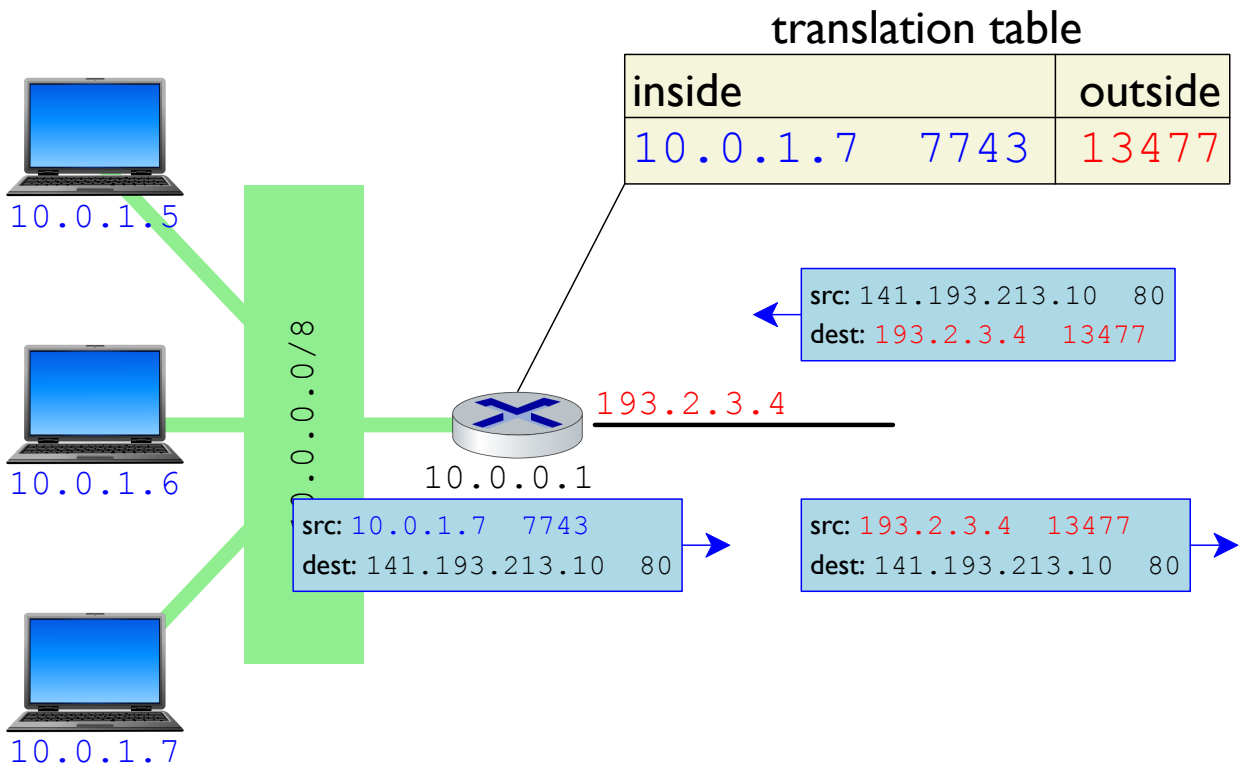
Network Address Translation (NAT)



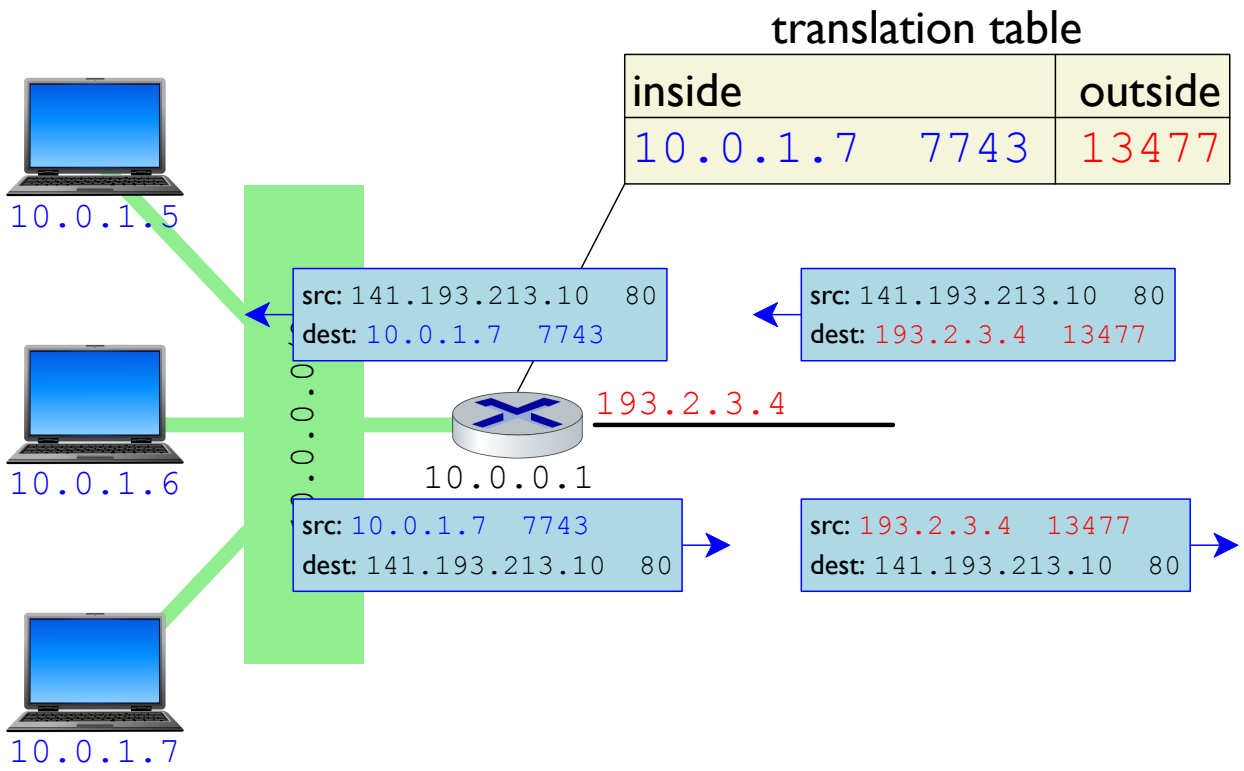
Network Address Translation (NAT)



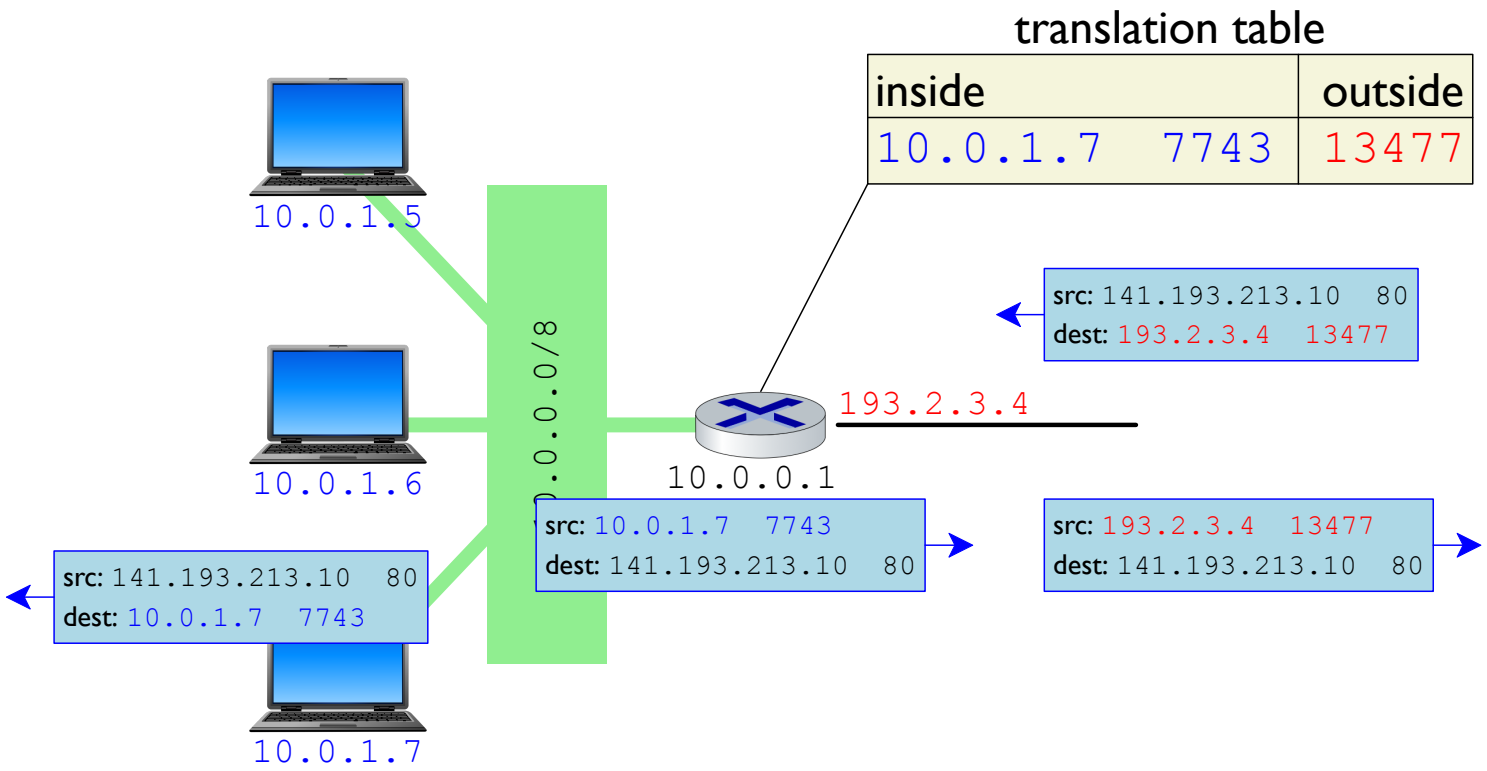
Network Address Translation (NAT)



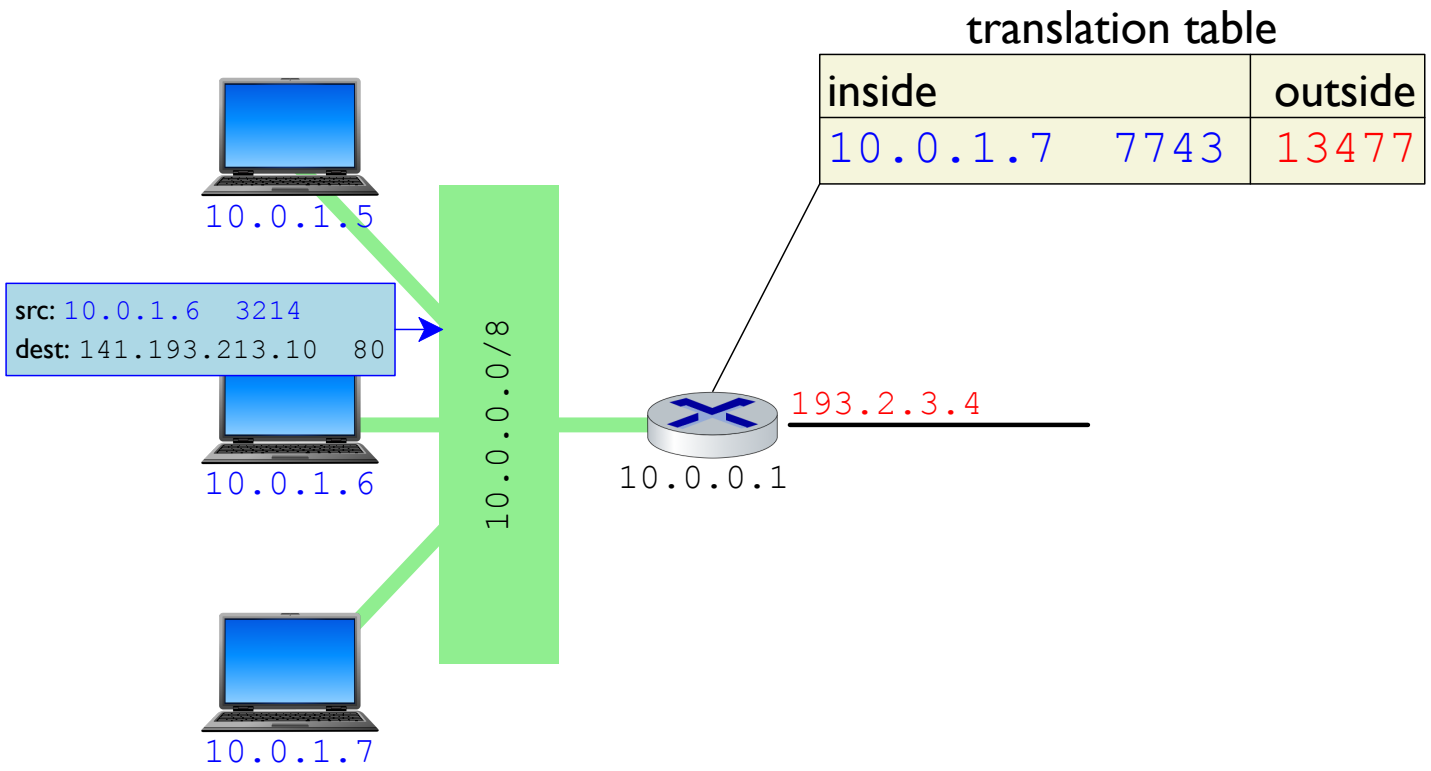
Network Address Translation (NAT)



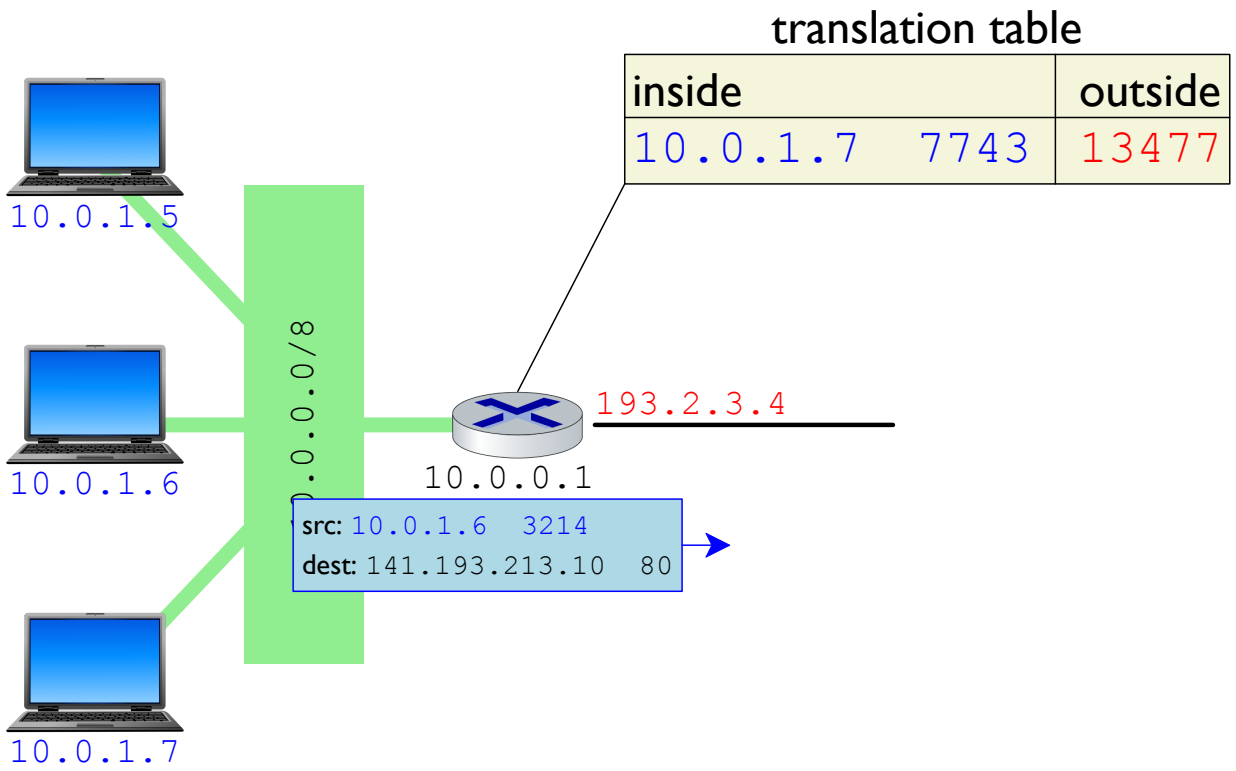
Network Address Translation (NAT)



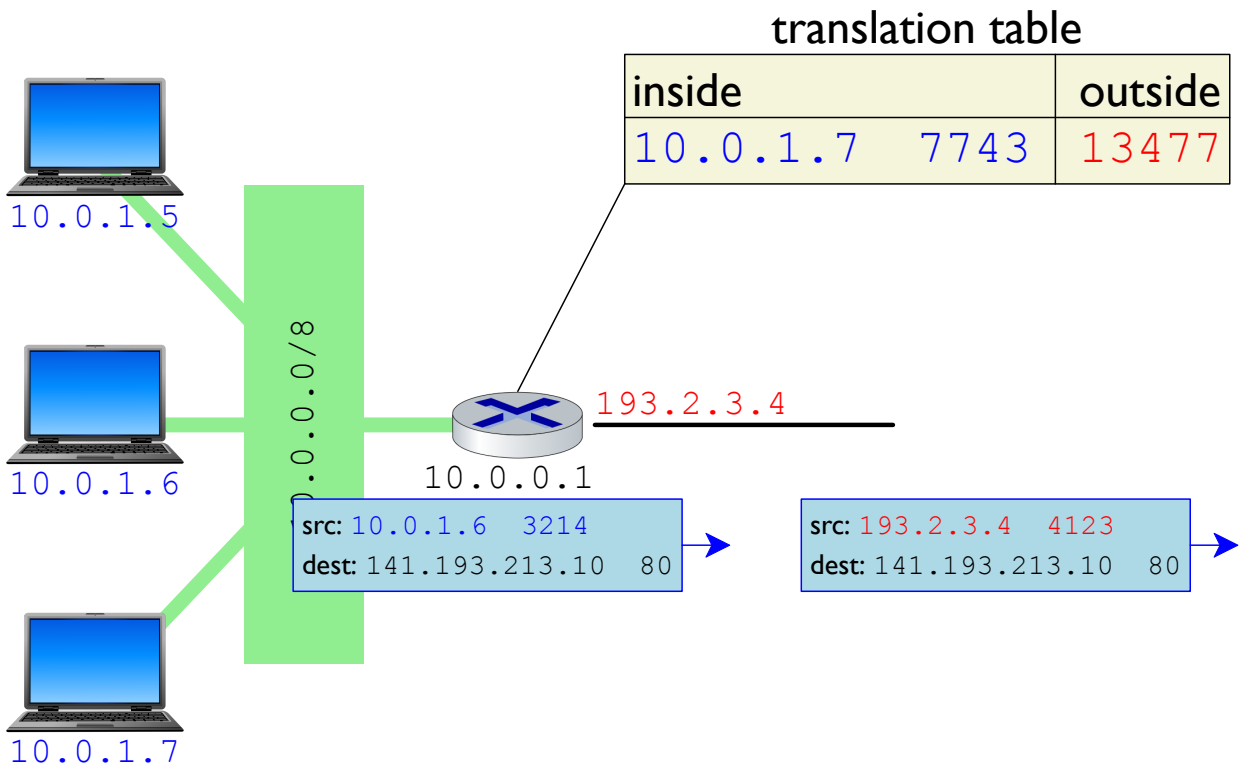
Network Address Translation (NAT)



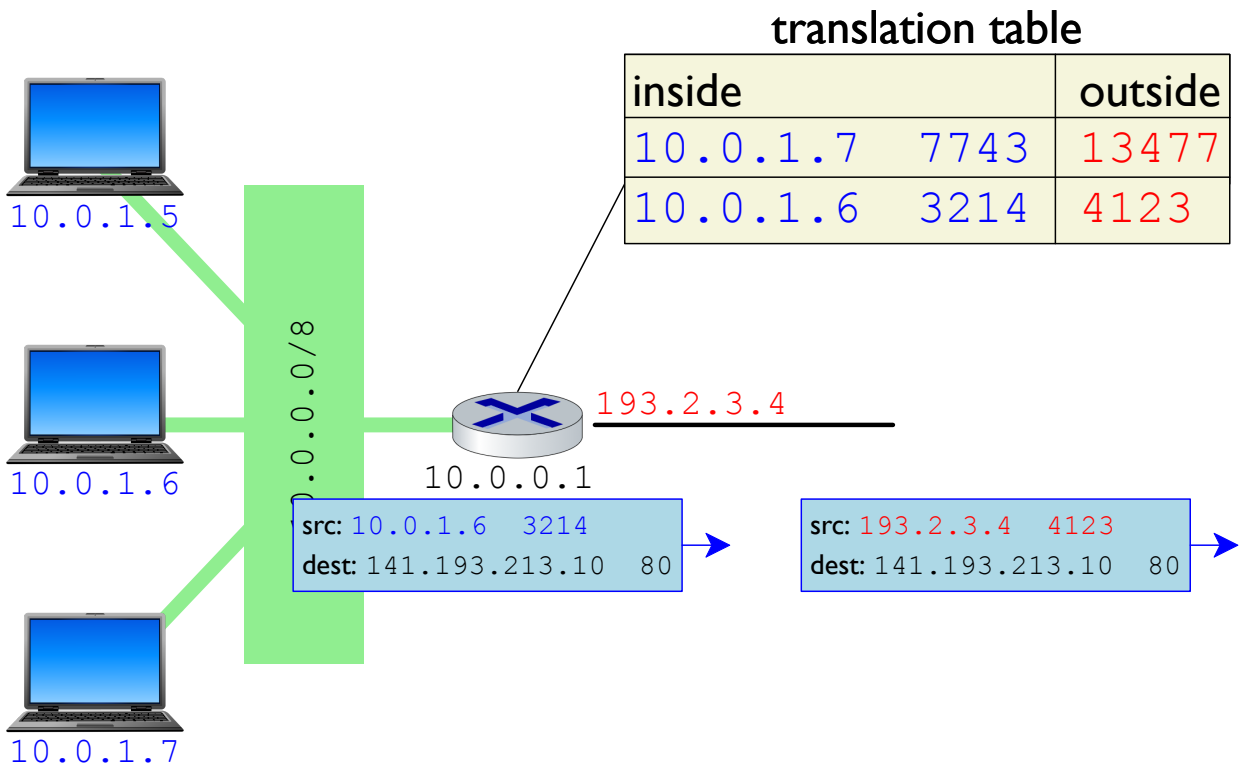
Network Address Translation (NAT)



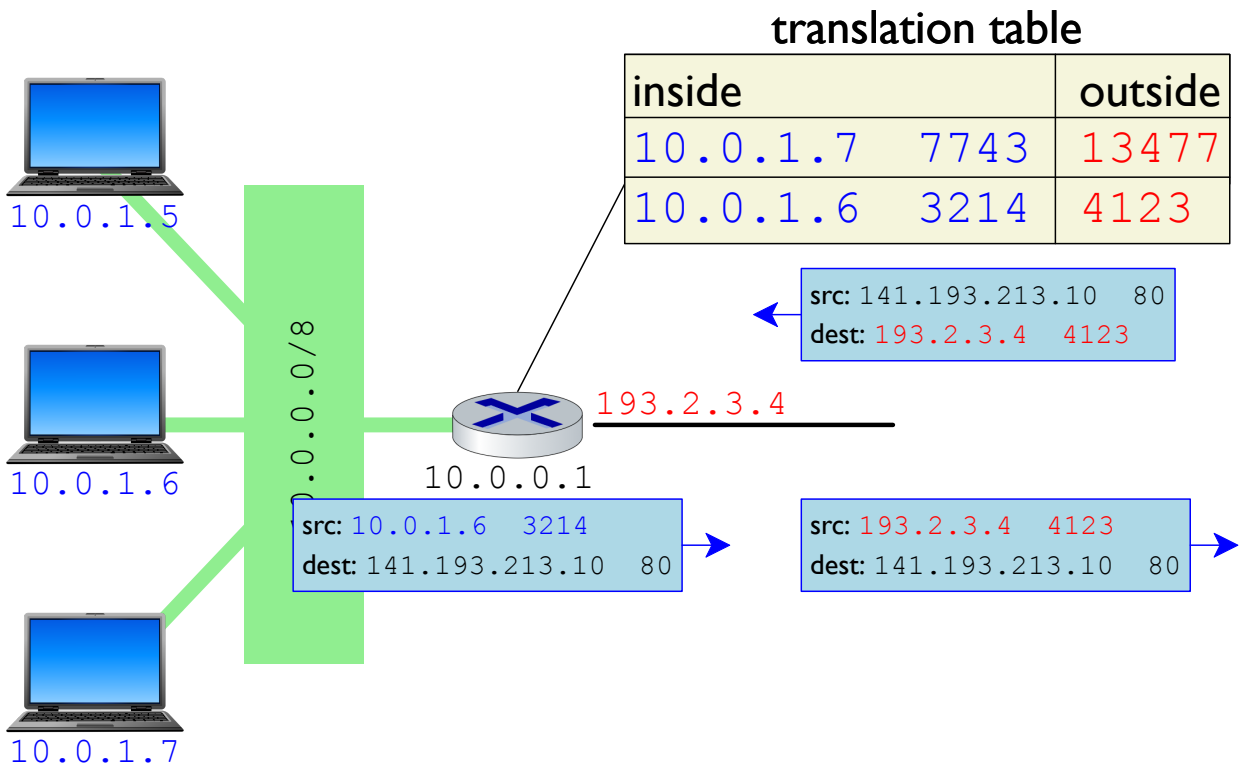
Network Address Translation (NAT)



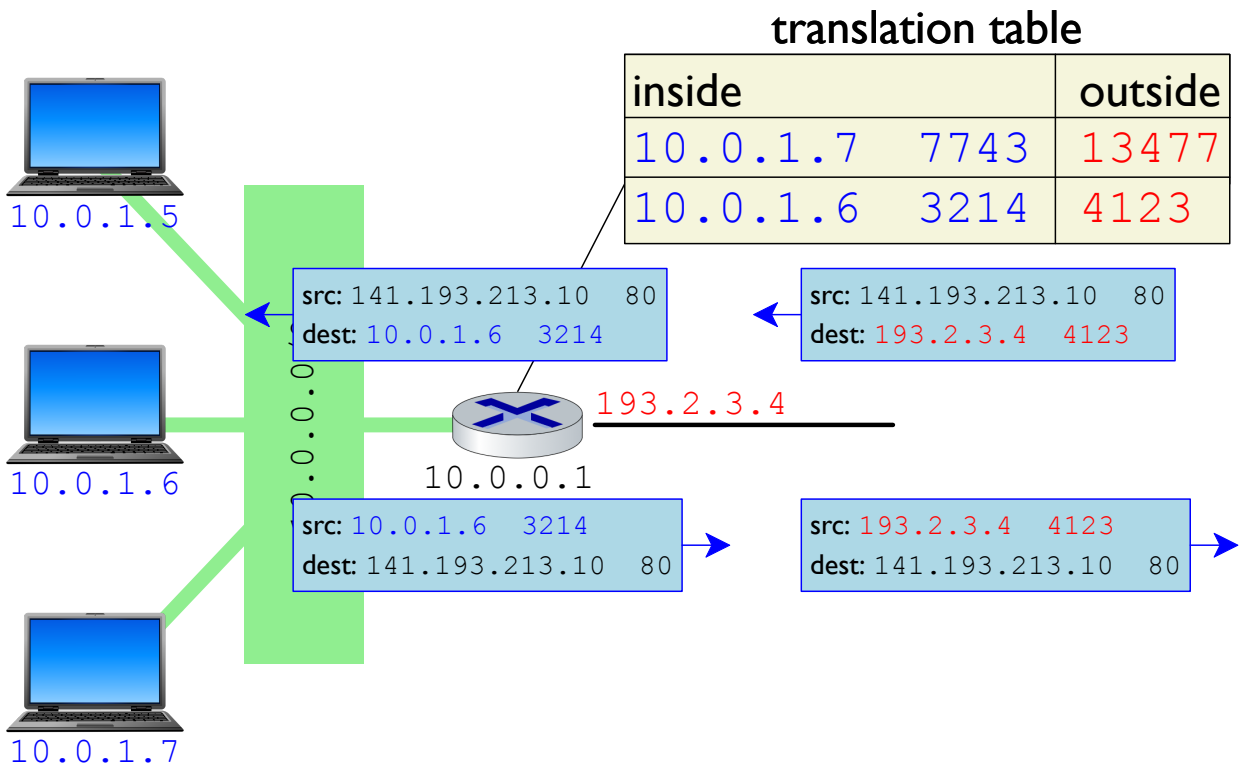
Network Address Translation (NAT)



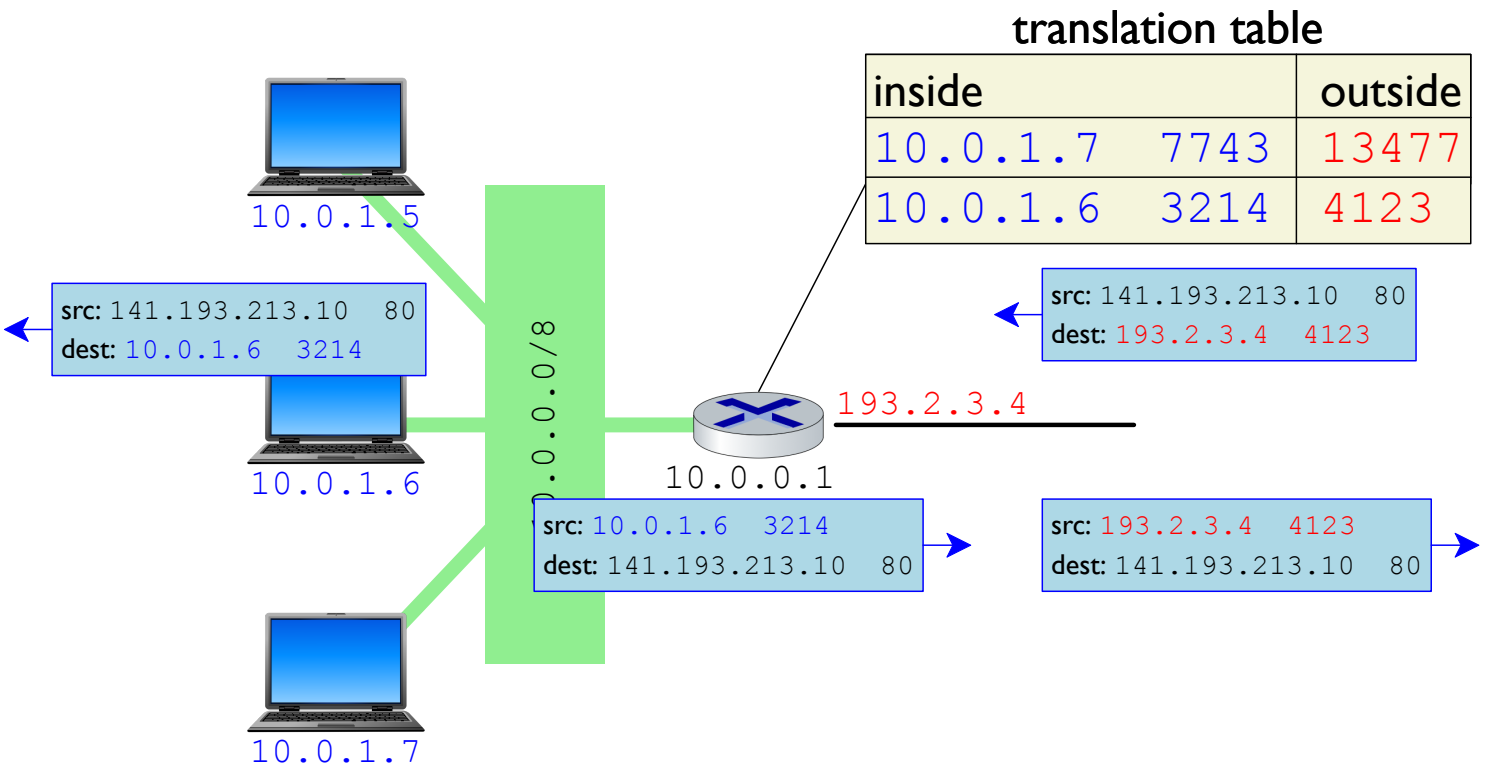
Network Address Translation (NAT)



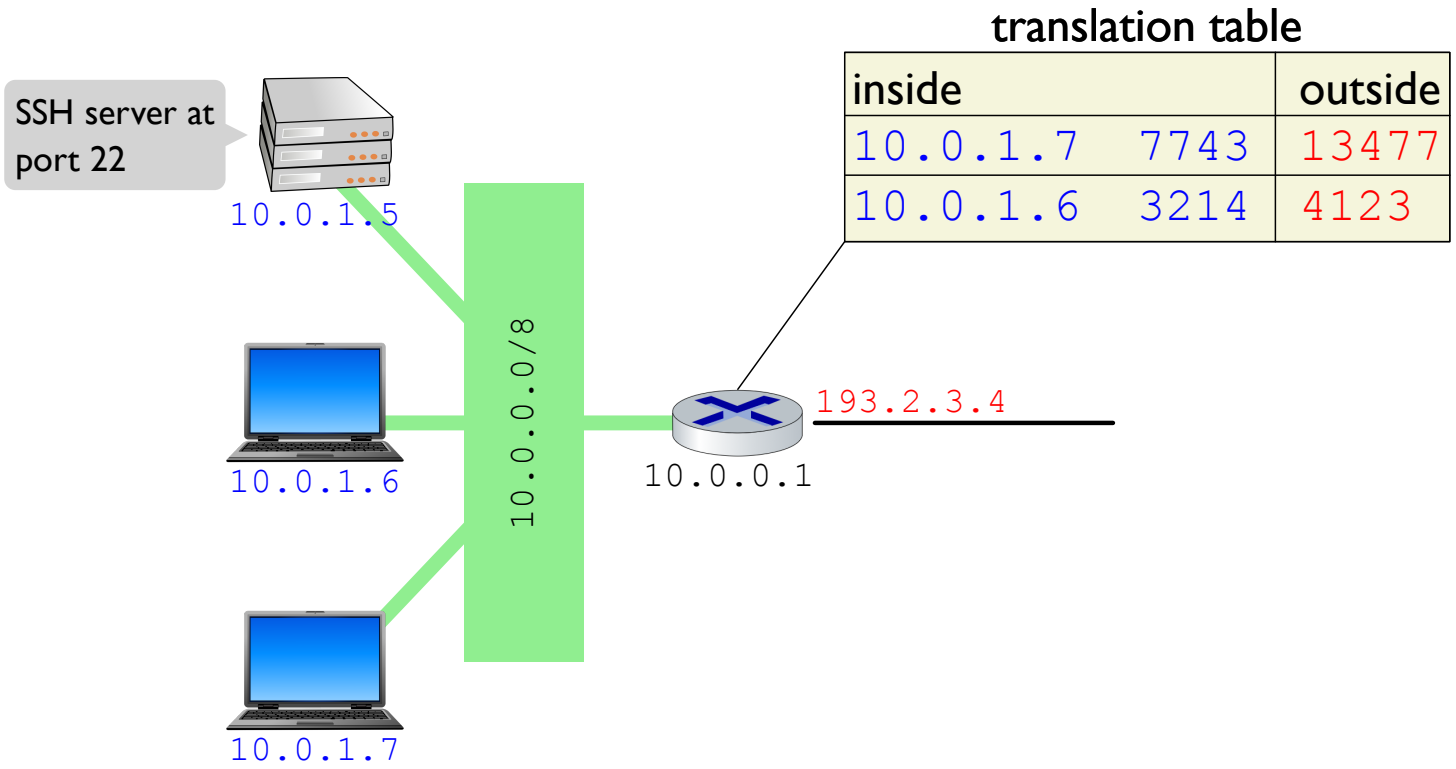
Network Address Translation (NAT)



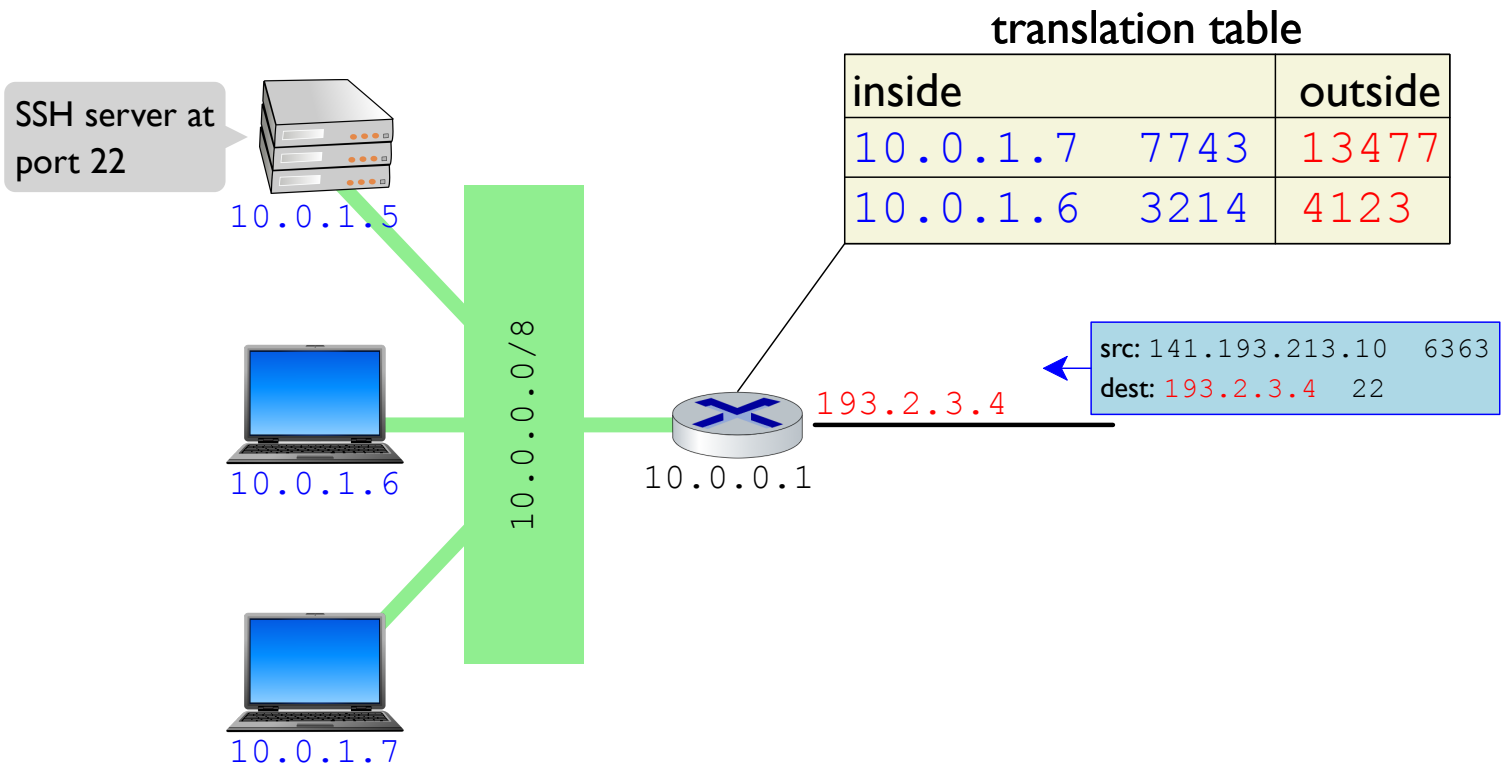
Network Address Translation (NAT)



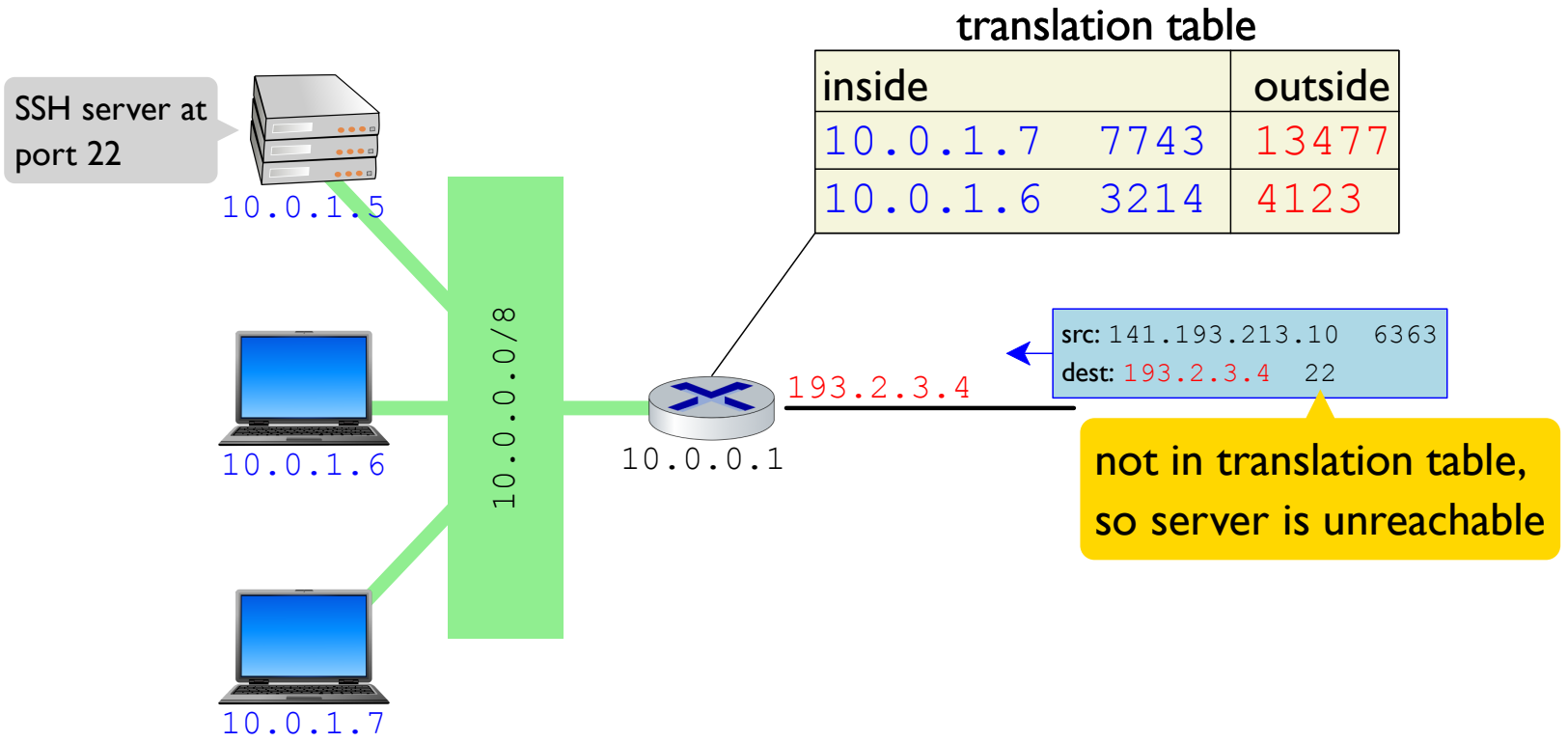
Network Address Translation (NAT)



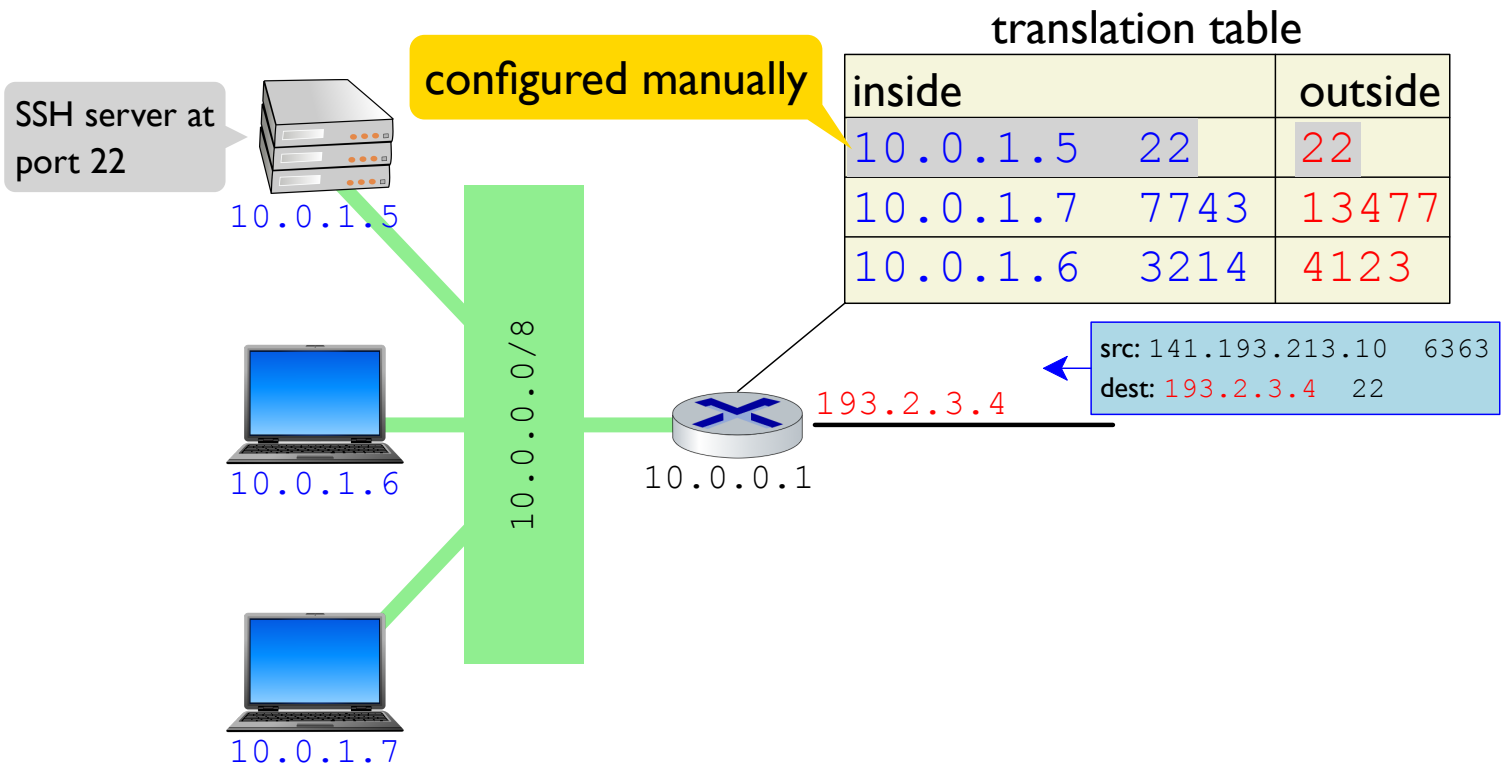
Network Address Translation (NAT)



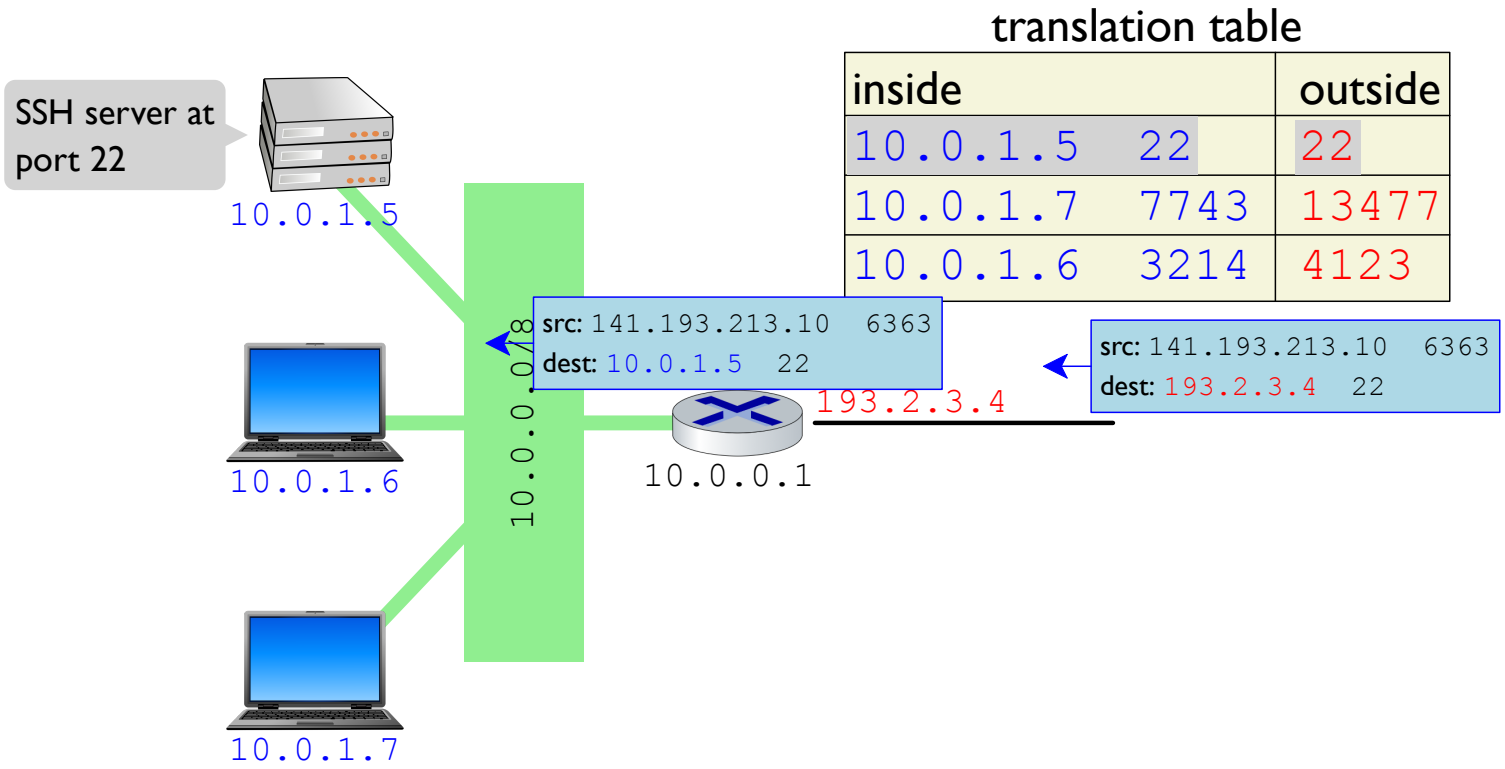
Network Address Translation (NAT)



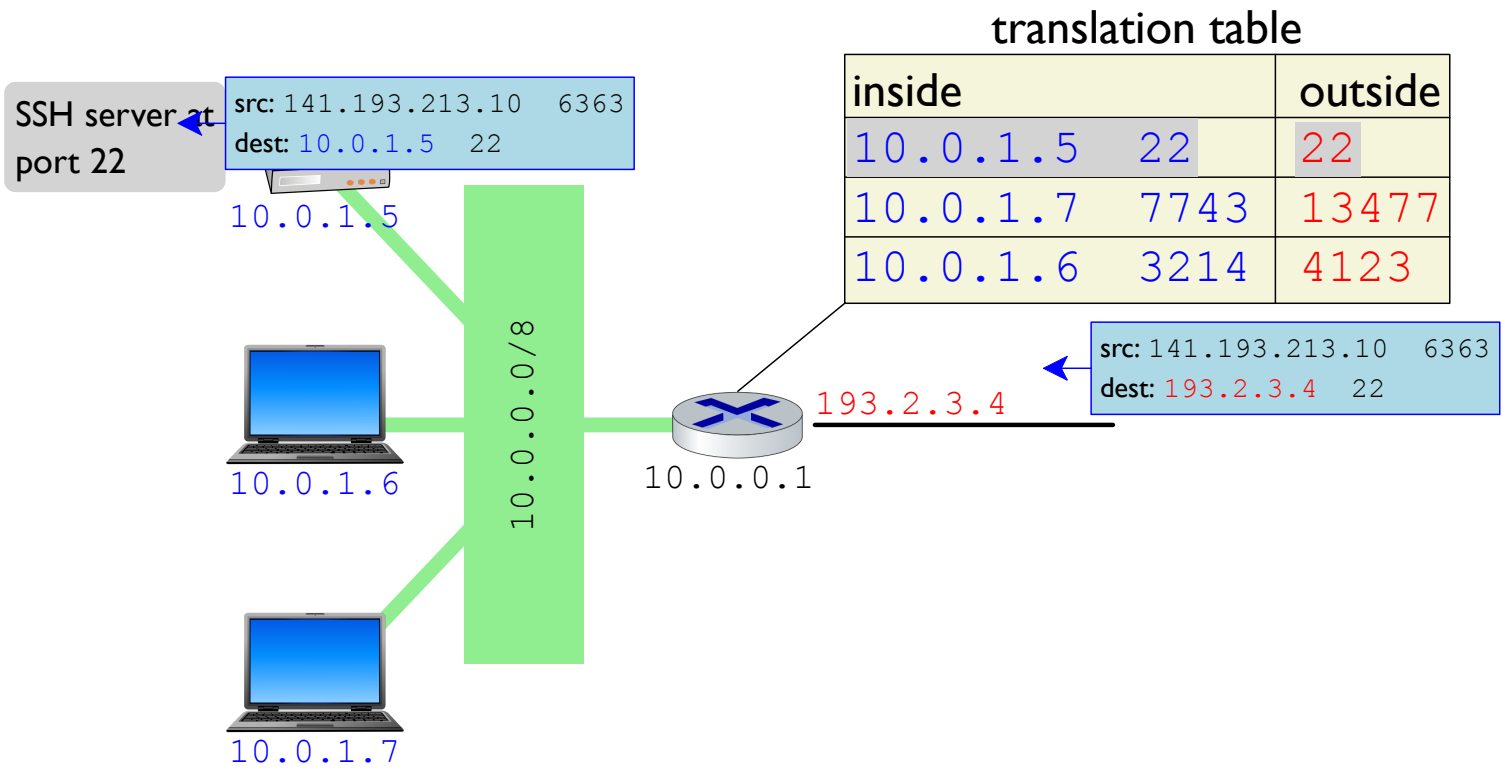
Network Address Translation (NAT)



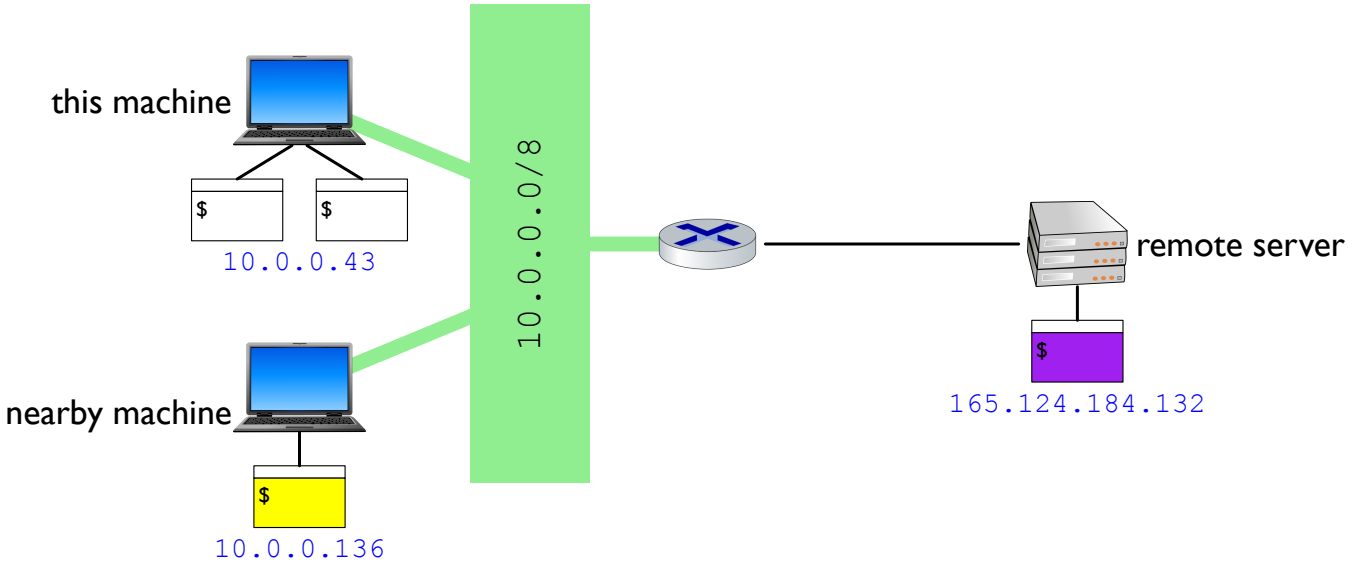
Network Address Translation (NAT)



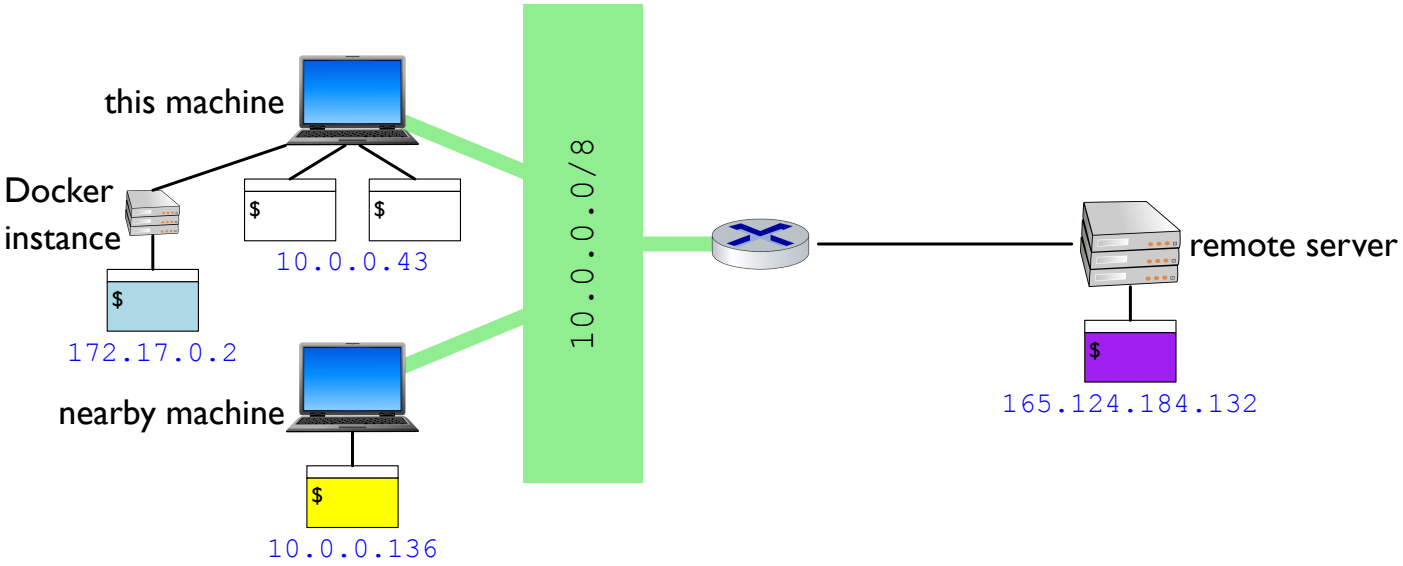
Network Address Translation (NAT)



Demo


this machine
$
$
10.0.0.43
nearby machine
$
10.0.0.136
10.0.0.0/8
remote server
$
165.124.184.132

Demo


this machine
Docker
instance
$
$
$
10.0.0.43
172.17.0.2
nearby machine
$
10.0.0.136
10.0.0.0/8
remote server
$
165.124.184.132

NAT Issues

A **NAT router** translates TCP and UDP packets to convert

$$\langle addr_{\text{inside}}, port_{\text{inside}} \rangle$$

to

$$\langle addr_{\text{outside}}, port_{\text{outside}} \rangle$$

where $addr_{\text{outside}}$ is NAT router's own address

NAT Issues

A **NAT router** translates TCP and UDP packets to convert

$$\langle addr_{\text{inside}}, port_{\text{inside}} \rangle$$

to

$$\langle addr_{\text{outside}}, port_{\text{outside}} \rangle$$

where $addr_{\text{outside}}$ is NAT router's own address

Does not work for well-known ports

NAT Issues

A **NAT router** translates TCP and UDP packets to convert

$$\langle addr_{\text{inside}}, port_{\text{inside}} \rangle$$

to

$$\langle addr_{\text{outside}}, port_{\text{outside}} \rangle$$

where $addr_{\text{outside}}$ is NAT router's own address

Assumes communication starts from inside

Mixes transport and network layers

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:0815:0000:0000:0000:2004`

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:0815:0000:0000:0000:2004`

Eight 16-bit numbers in hexadecimal

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:815::2004`

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:815::2004`

Up to one contiguous sequence of :-separated `0000`s can be omitted

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:815::2004`

Leading `0`s can be dropped

IPv6

128-bit addresses instead of 32-bit addresses

`172.217.164.4`

vs.

`2607:f8b0:4025:815::2004`

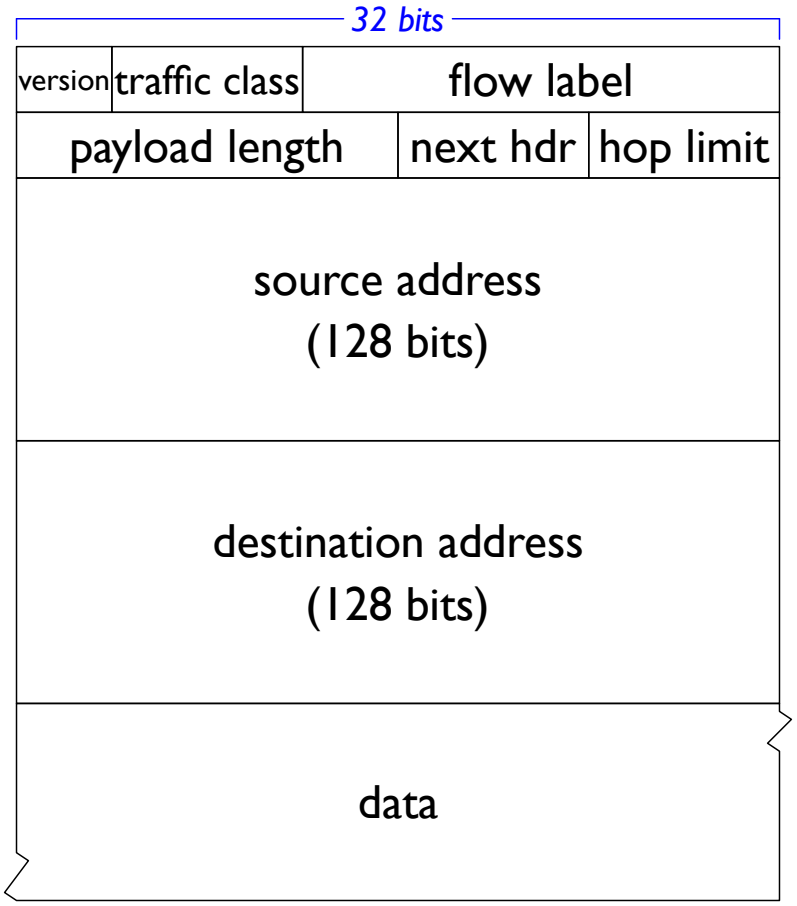
Loopback is `::1` (instead of `127.0.0.1`)

Looking up an IPv6 Address

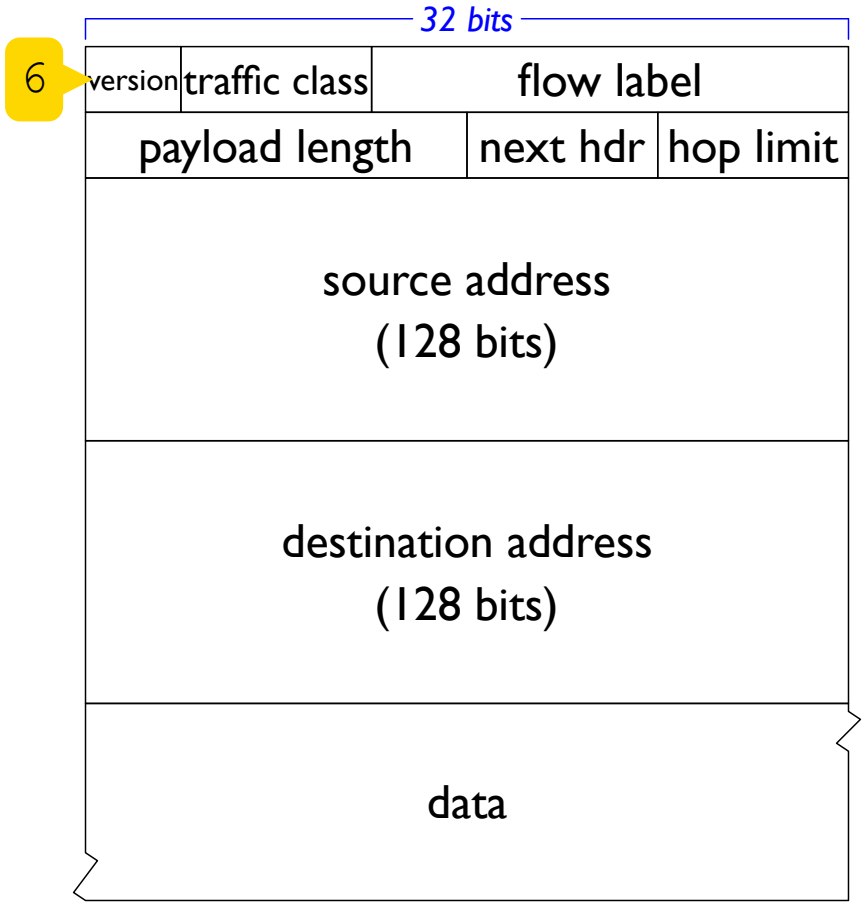
```
$ dig www.google.com A
....
www.google.com.         216     IN      A       172.217.164.4
....

$ dig www.google.com AAAA
....
www.google.com.         250     IN      AAAA    2607:f8b0:4025:803::2004
....
```

IPv6 Packet Format

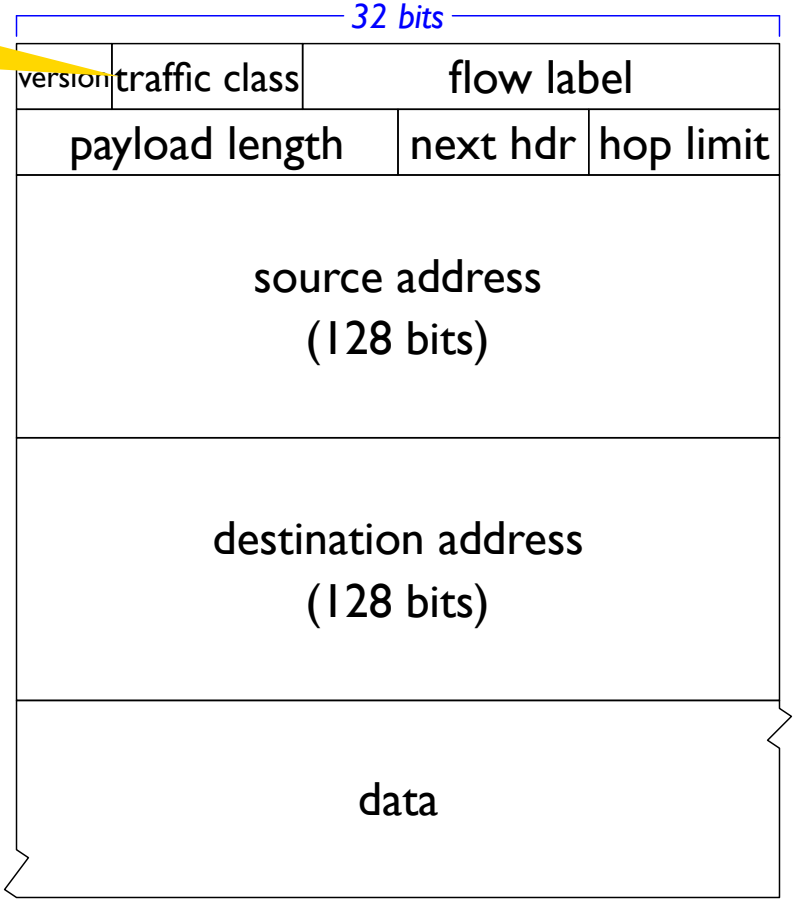


IPv6 Packet Format

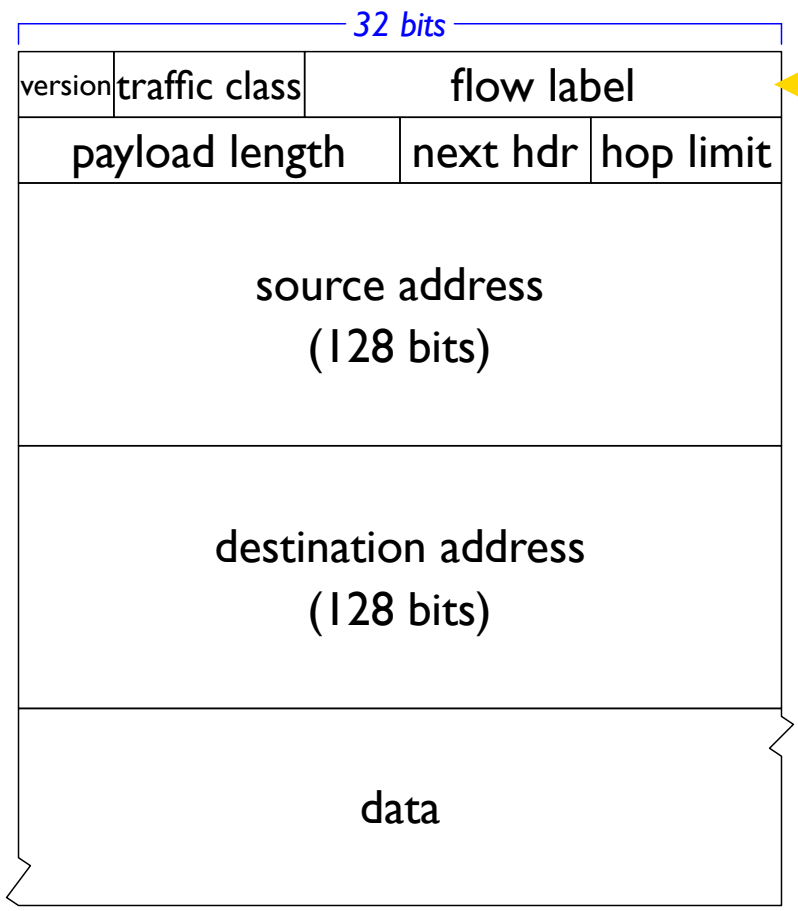


IPv6 Packet Format

Same as "type of service"

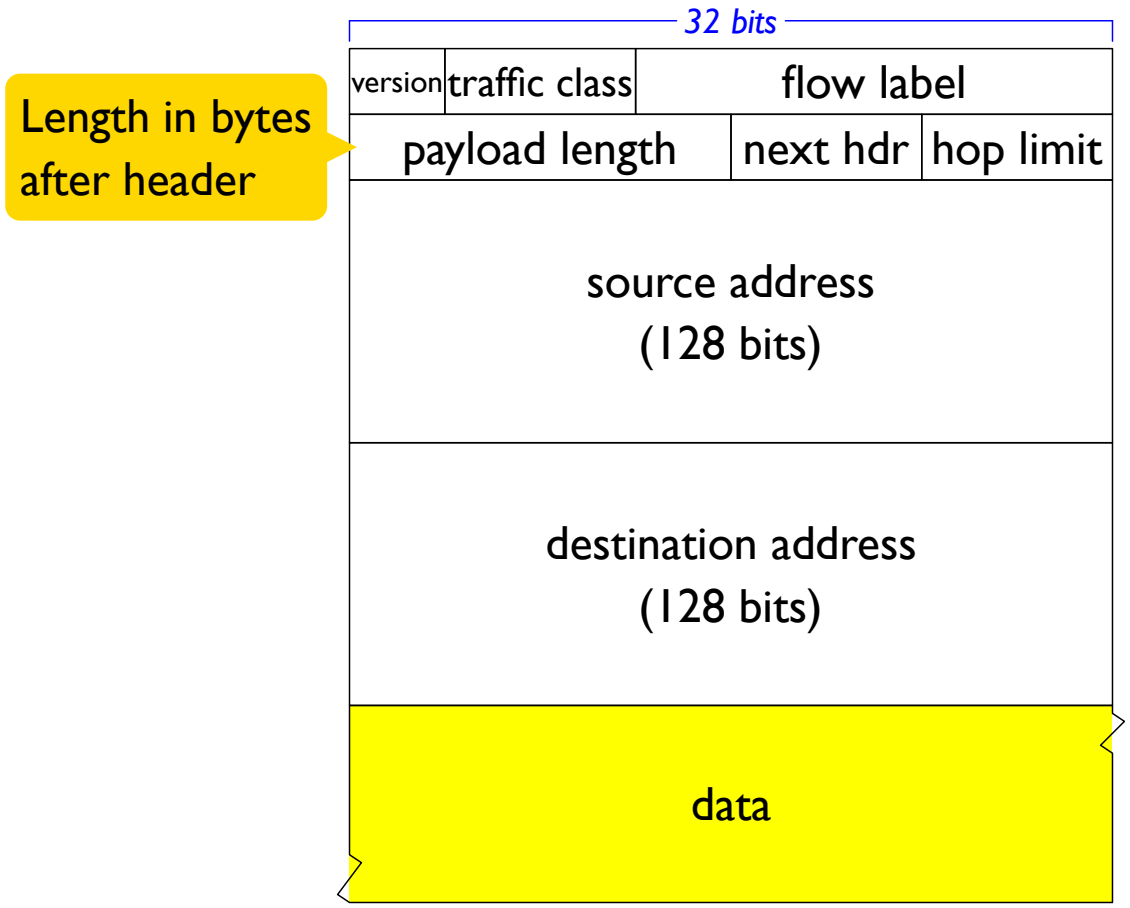


IPv6 Packet Format

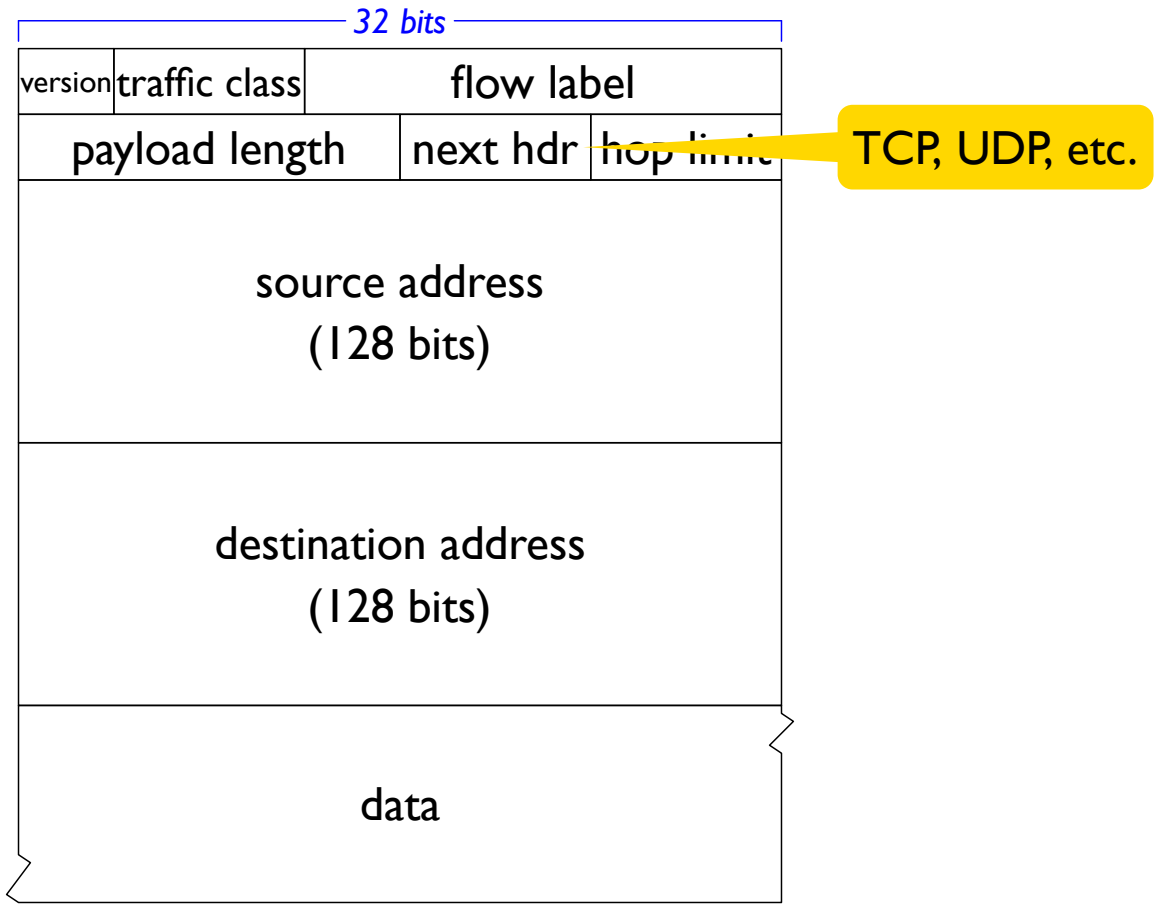


To identify related packets;
like traffic class,
use is up to routers

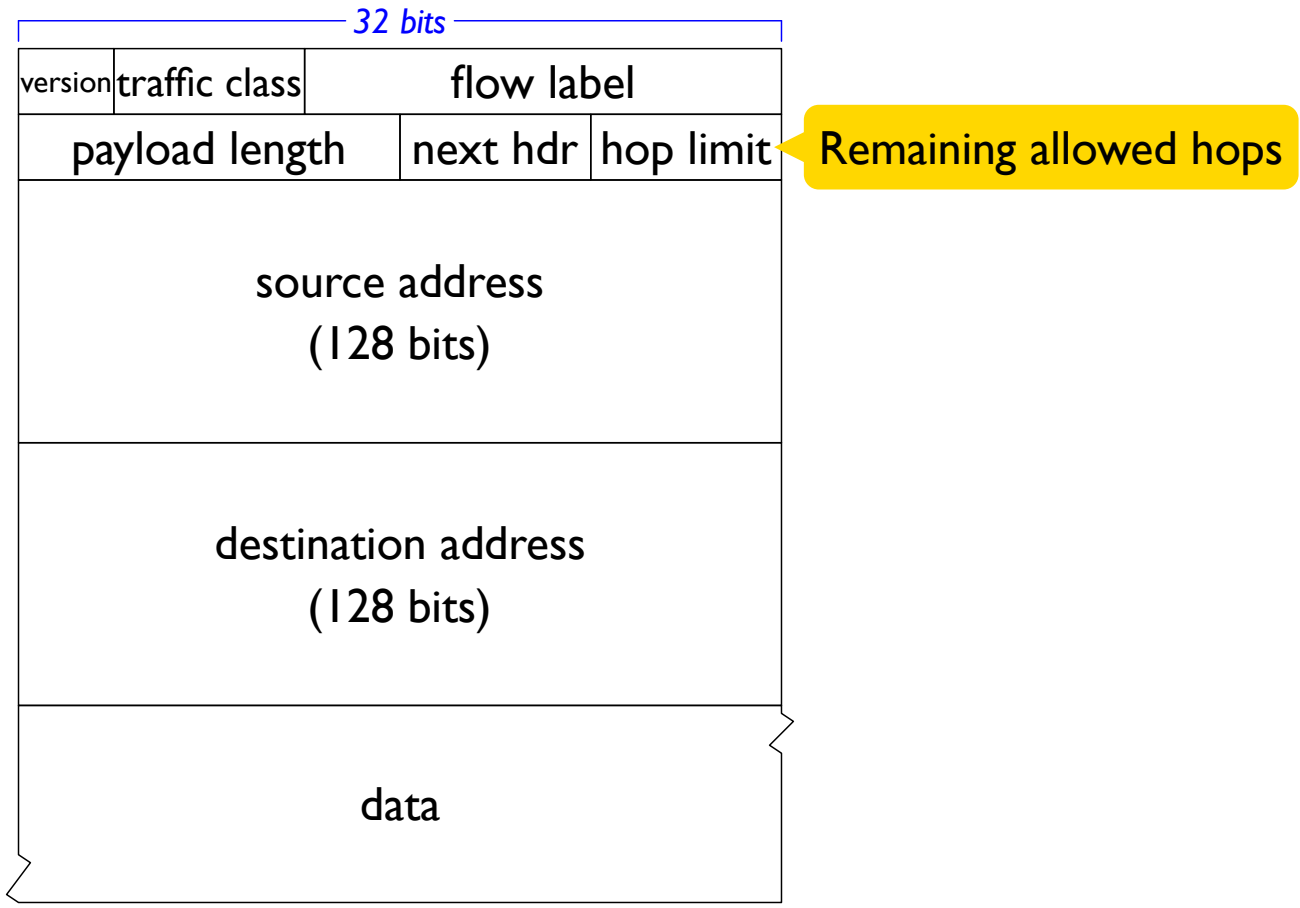
IPv6 Packet Format



IPv6 Packet Format



IPv6 Packet Format



Switching to IPv6

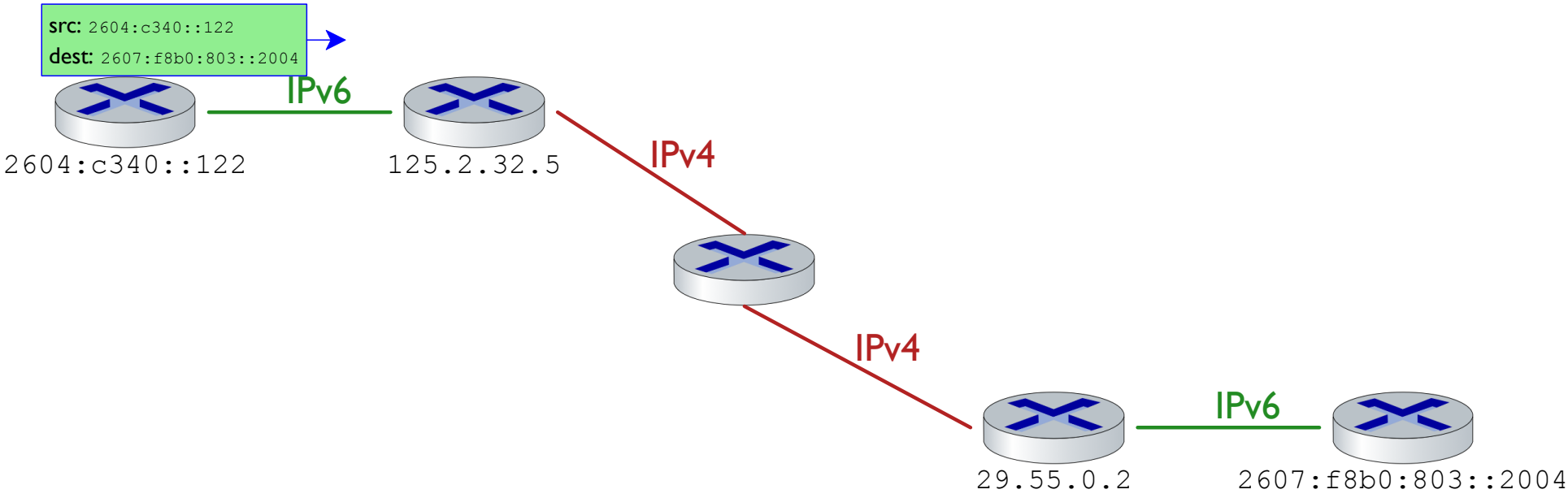
Why don't we all switch to IPv6?

Everything has to change:

- router hardware
- router software
- name-resolver protocols
- operating-system drivers and services
- configuration tools
- applications with thin abstractions

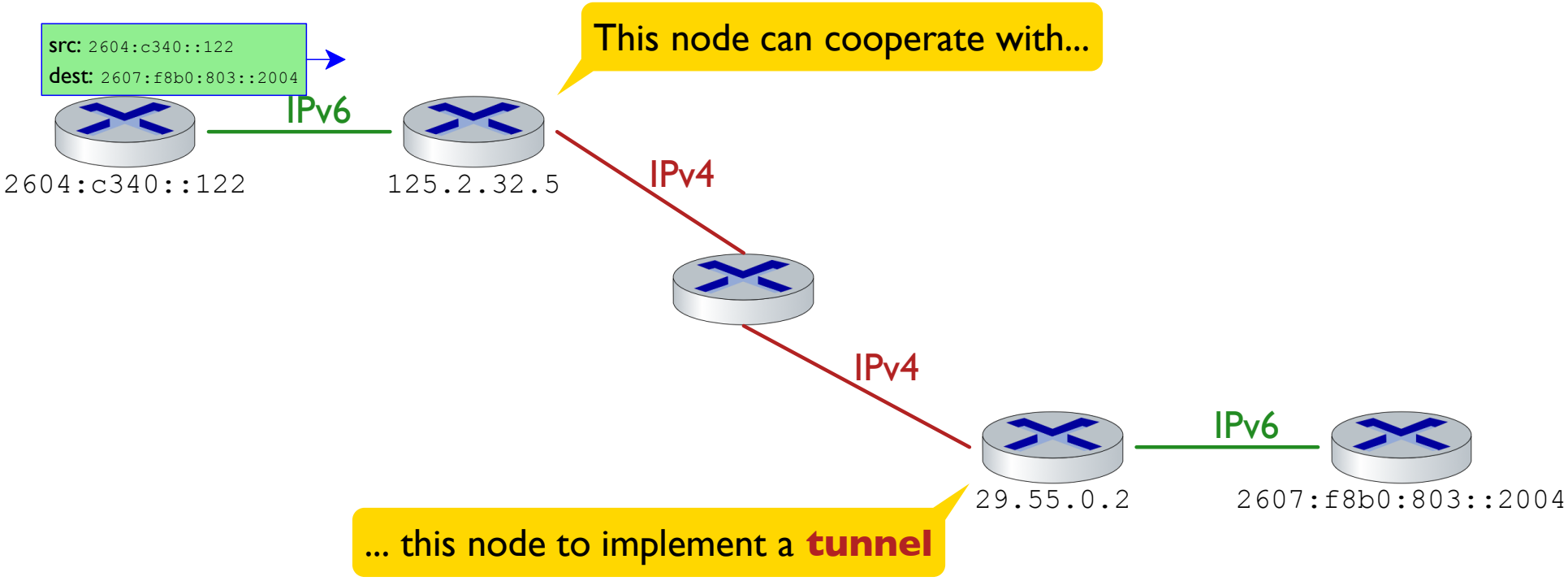
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



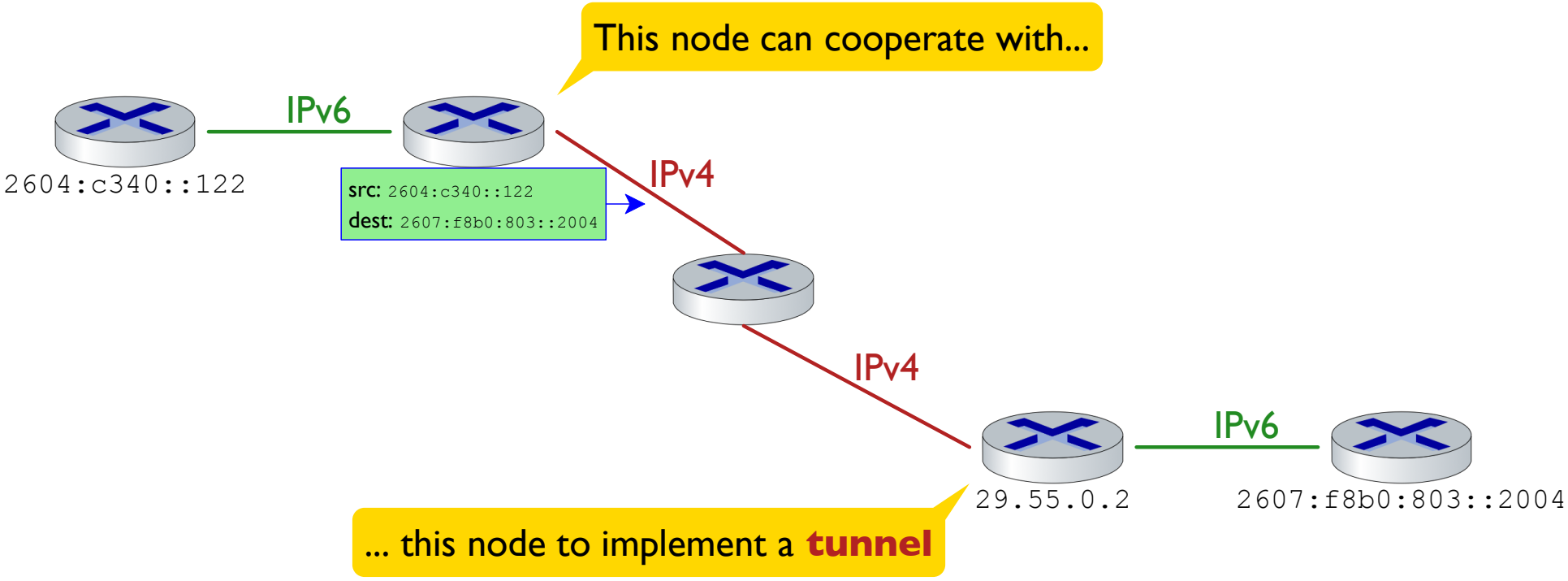
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



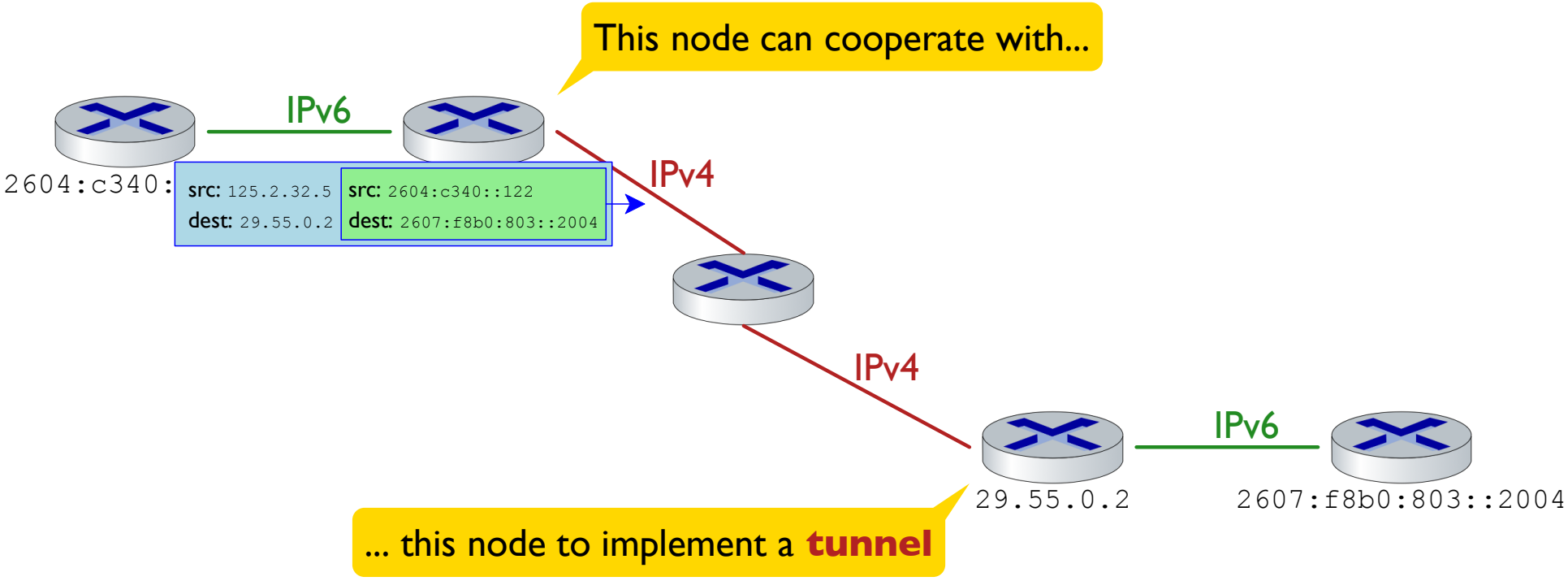
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



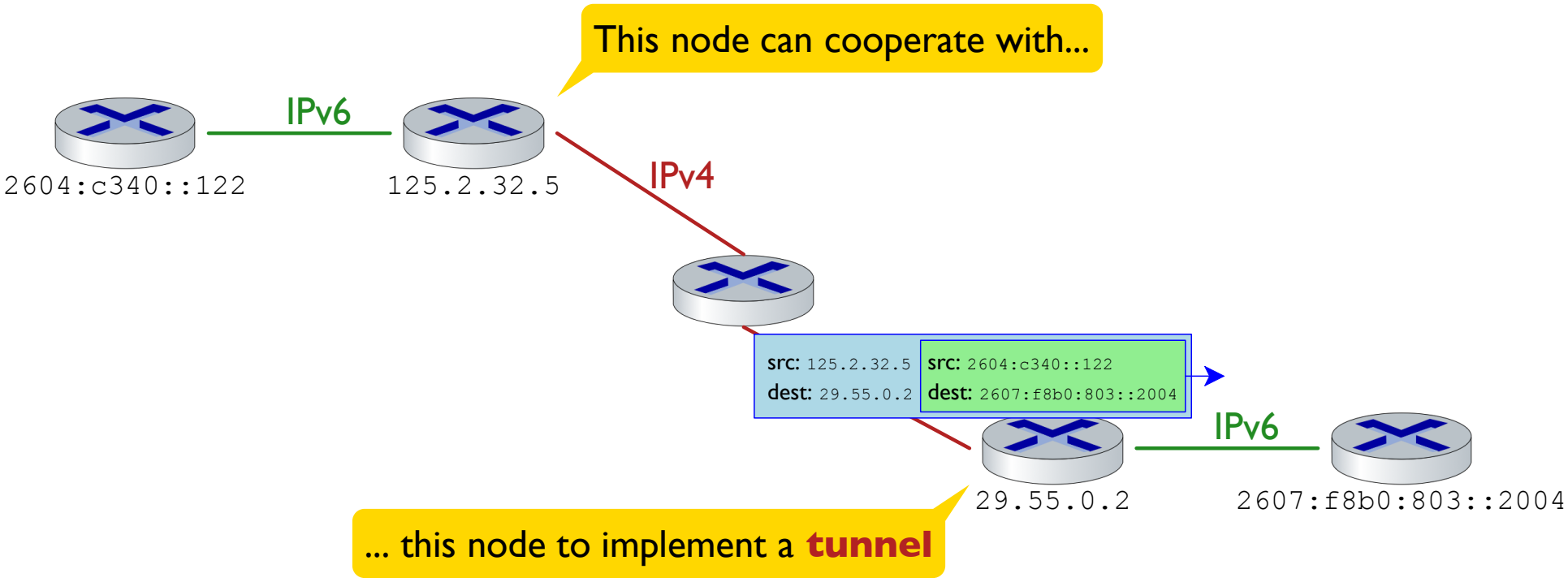
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



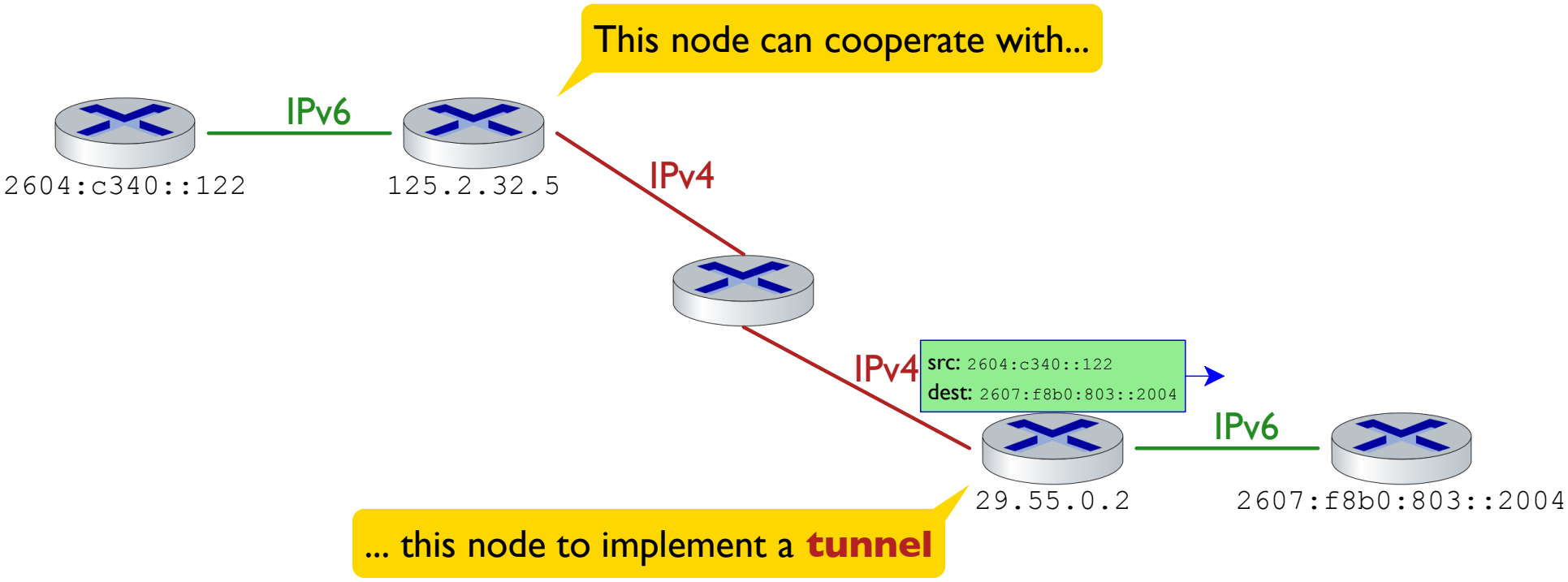
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



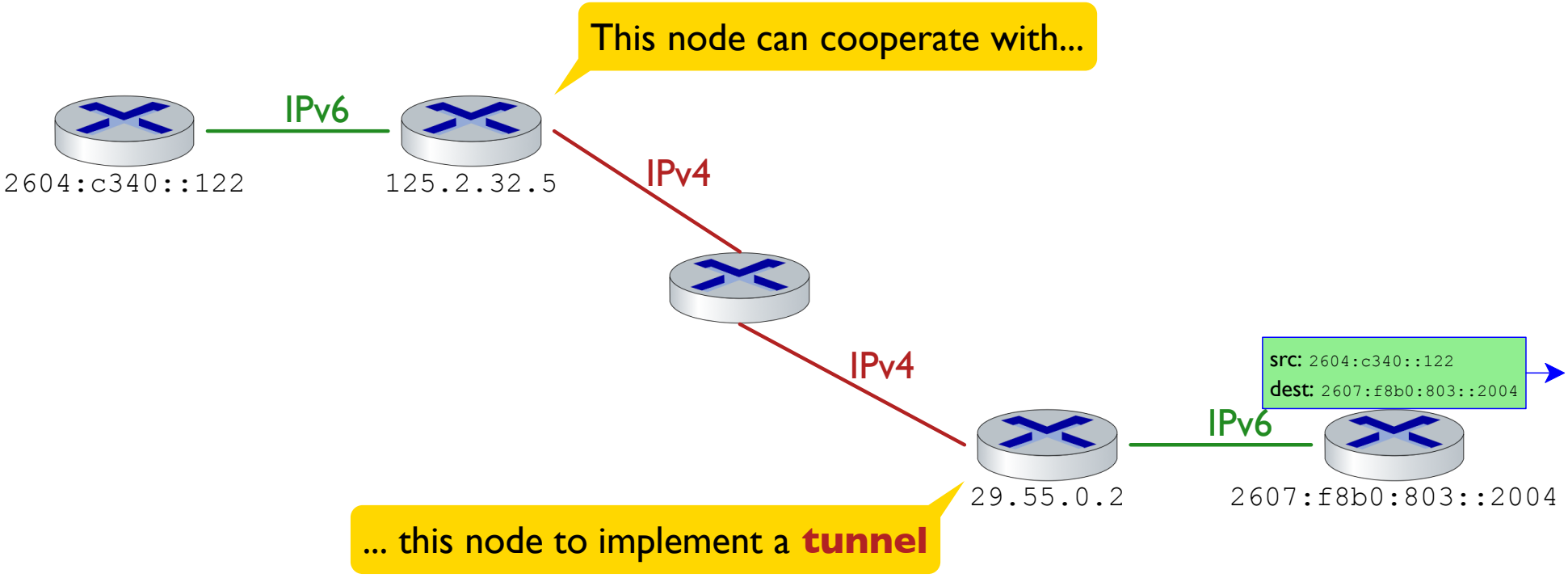
Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



Tunnelling

What if you have hosts that are happy to talk IPv6, but all routes involve IPv4?



Summary

The network layer **data plane** is IPv4 or IPv6 packets

IPv4 uses 32-bit addresses

IPv6 is 128-bit addresses ...and a simpler packet header

Network Address Translation (NAT) delays need for IPv6

Tunneling IPv6 through IPv4 helps the transition