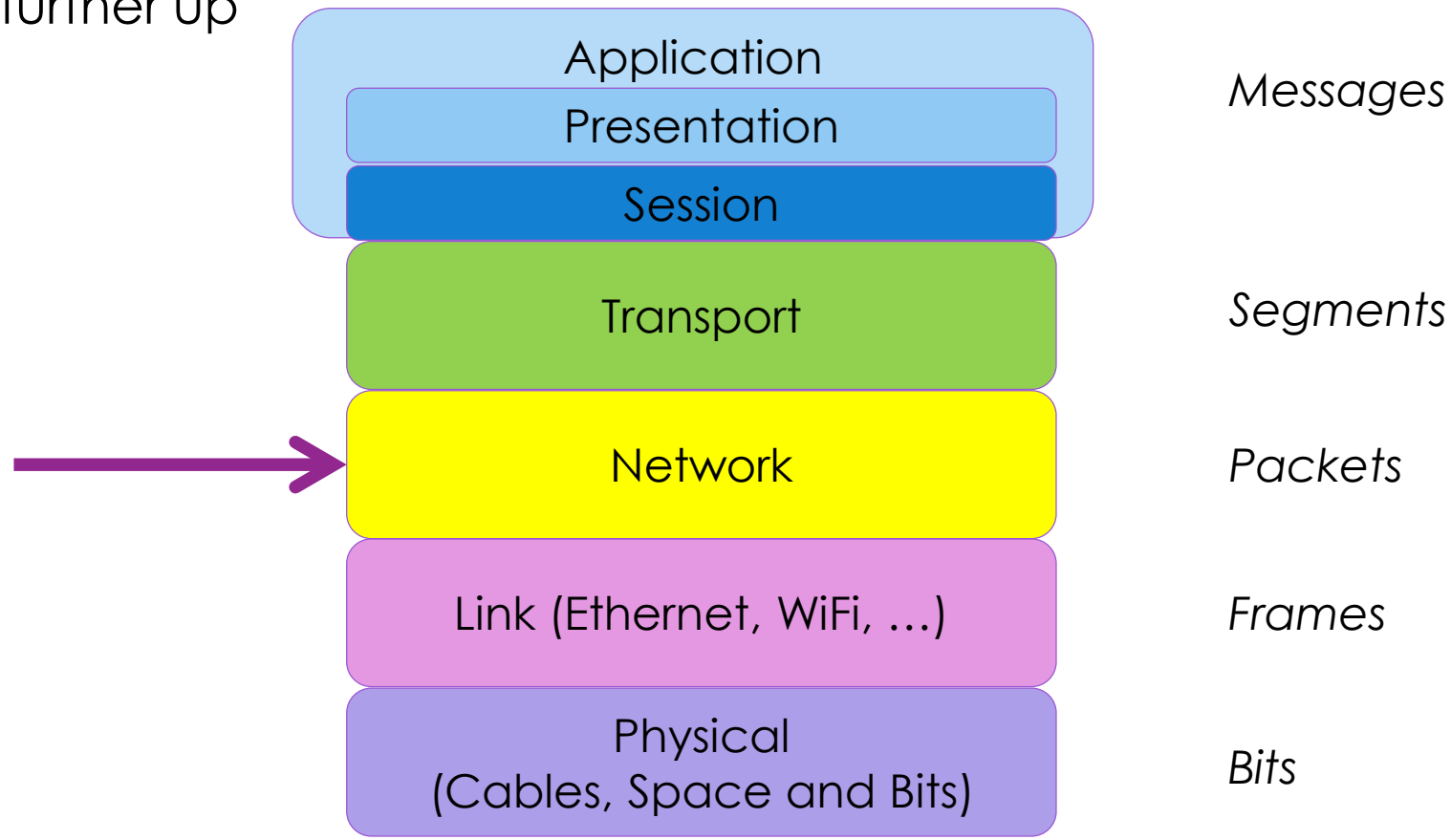
COMP3310/6331 – #10-11

Network Layer: IPv4/v6

Dr Markus Buchhorn: *markus.buchhorn@anu.edu.au*

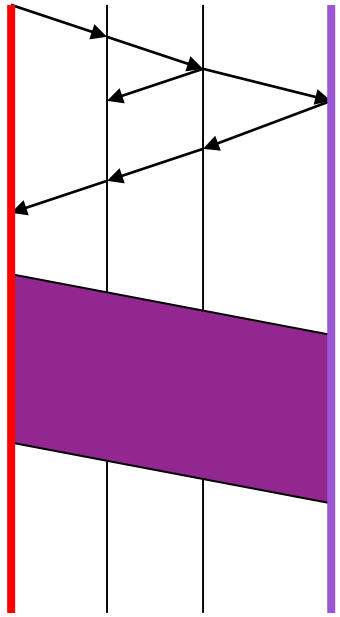
Where are we?

- Moving further up



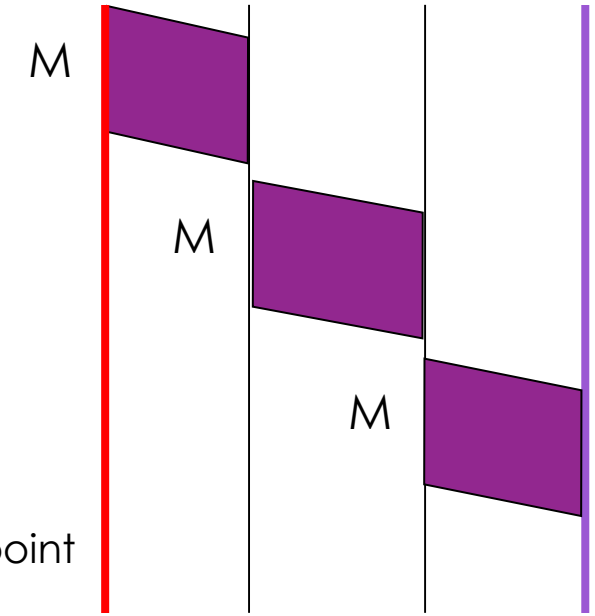
Remember **circuits**?

- POTS/PSTN
- Set up (and tear-down)
- Guaranteed channel
  - But inefficient
  - Solid block-out for duration of a conversation



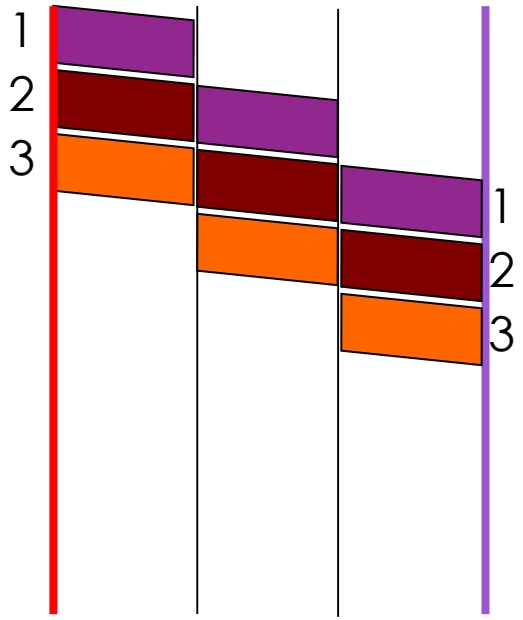
After Circuit switched…

- MESSAGE switched:
  - Postal Service:
    - Put message in container,
    - Address it
    - Put in the network
  - Network (hopefully) takes care of delivery
  - "Store-and-forward" – hold entire message
    - Examine container at each hop before forwarding.
  - Message loss is a large problem.
    - Less than a circuit. Failure along the path is flagged from that point
  - Potentially High Latency
    - Each hop takes time, especially for large messages



After Message Switched…

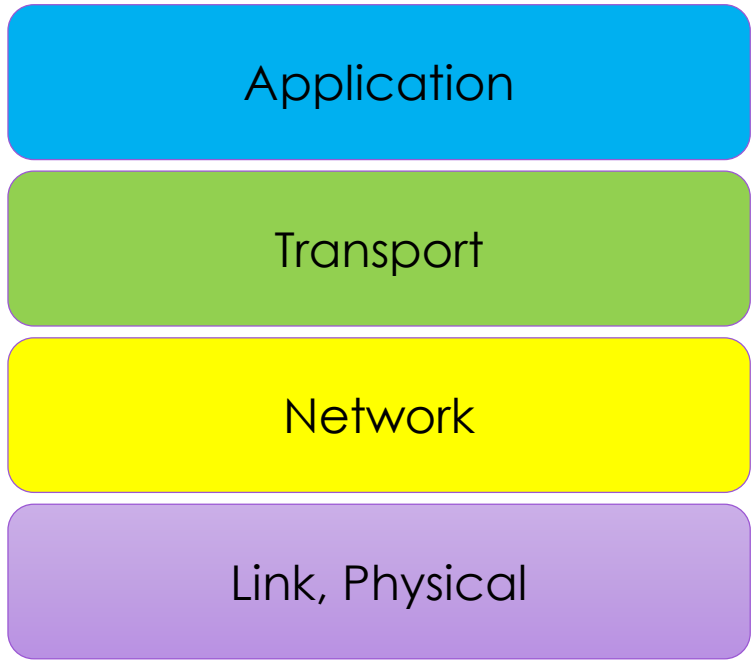
- PACKET switched
  - Put *fragments* of message in *multiple* containers
  - Address them all, and put them on the network
  - Network (hopefully) takes care of delivery
    - Examine each container at each hop before forwarding.
      - Acknowledgement happen sooner, more frequently
    - Packet loss is more tolerable
      - Recovery is a smaller effort
  - Still high latency
    - Each hop takes time, plus overheads
    - But less than for large messages
      - Not waiting for each hop
  - And much better sharing of capacity



Role of the **Network** layer

- Each consumes services (functionality) from the layer below
- Each offers services (functionality) to layer above

- Which functions belong in
  - Link/Physical
  - Transport
  - Network?

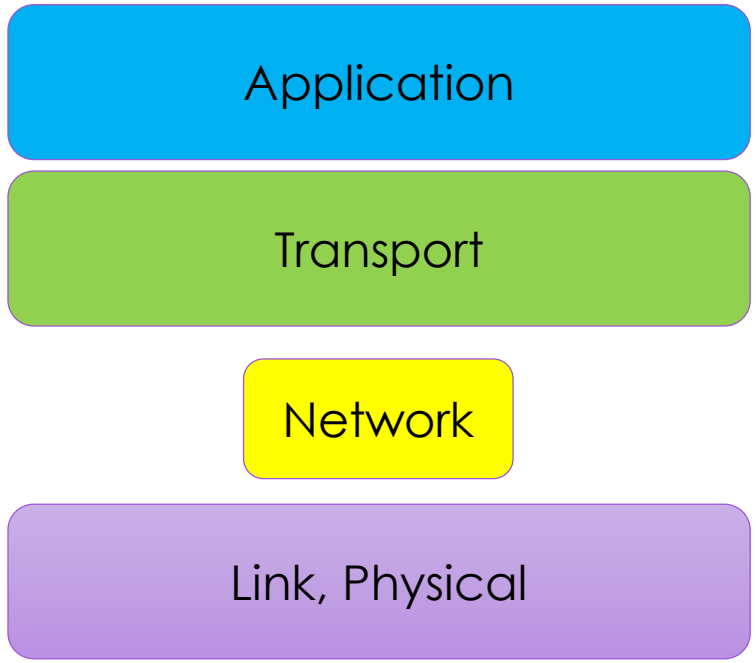


Role of the (IP) Network layer

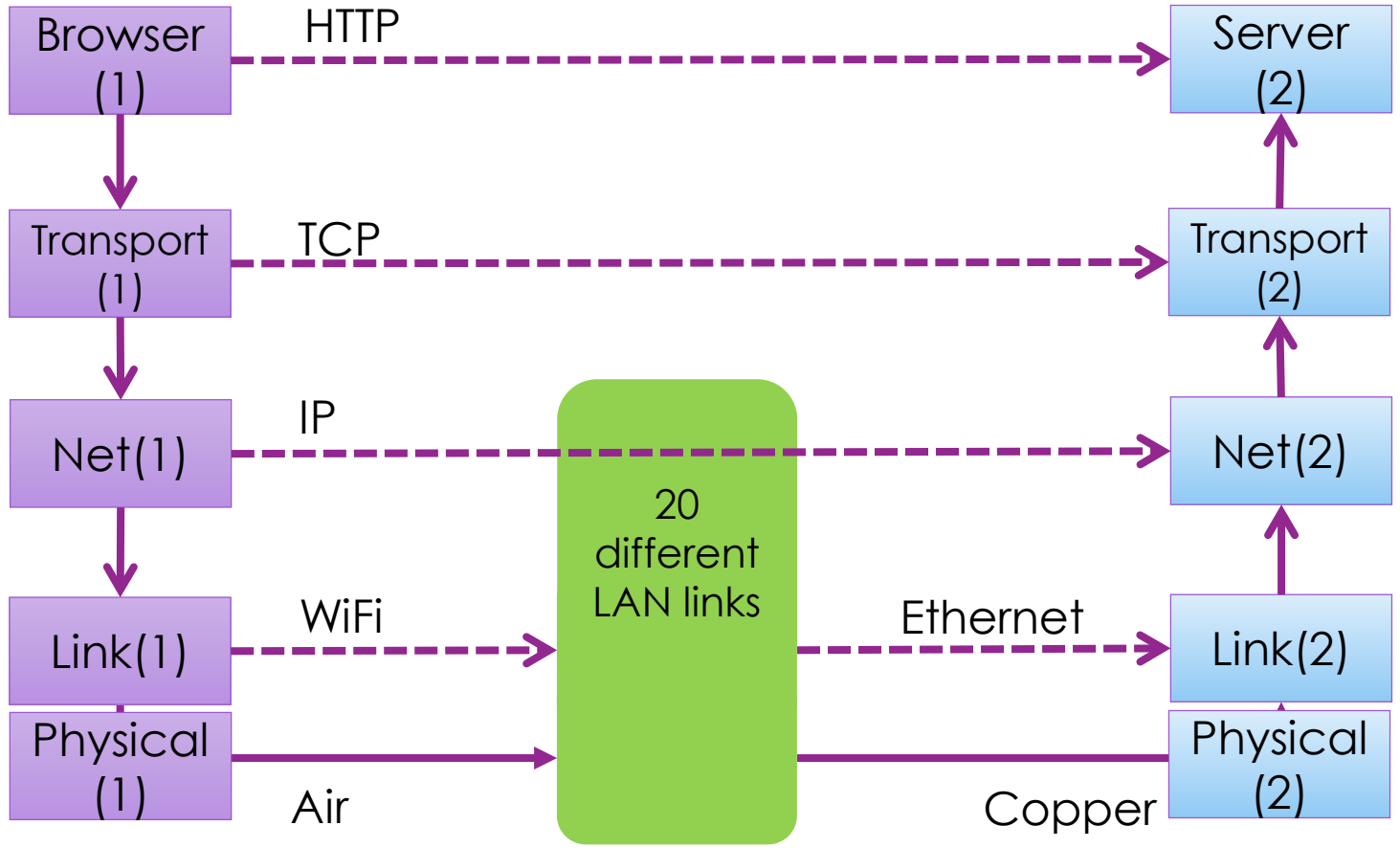
- Simplest common functions
  - Across **many/all** link types
  - Just a glue layer

1. End-to-end delivery of packets
2. Global addressing
3. Cope with evolving network topology

- Consume little from lower layer
- Offer little to higher layers



Hiding **path** complexity



Applications don't know nor care. Unless there is a performance question.

Guiding principles

- Simplicity and end-to-end design
  - Keep connection/conversation state at the edge, not in the network
- Provide '**best-effort**' delivery
  - Minimal Service Level Agreement (SLA)
- Ensure '**reliability**' is delivered (only) where it is needed
  - For specific application needs: file transfers, audio calls, …
  - Can be done at different layers
    - Link (vlan, …), Transport (tcp, …),
    - Network layer? (mpls, …)

Connectionless vs Connection-oriented

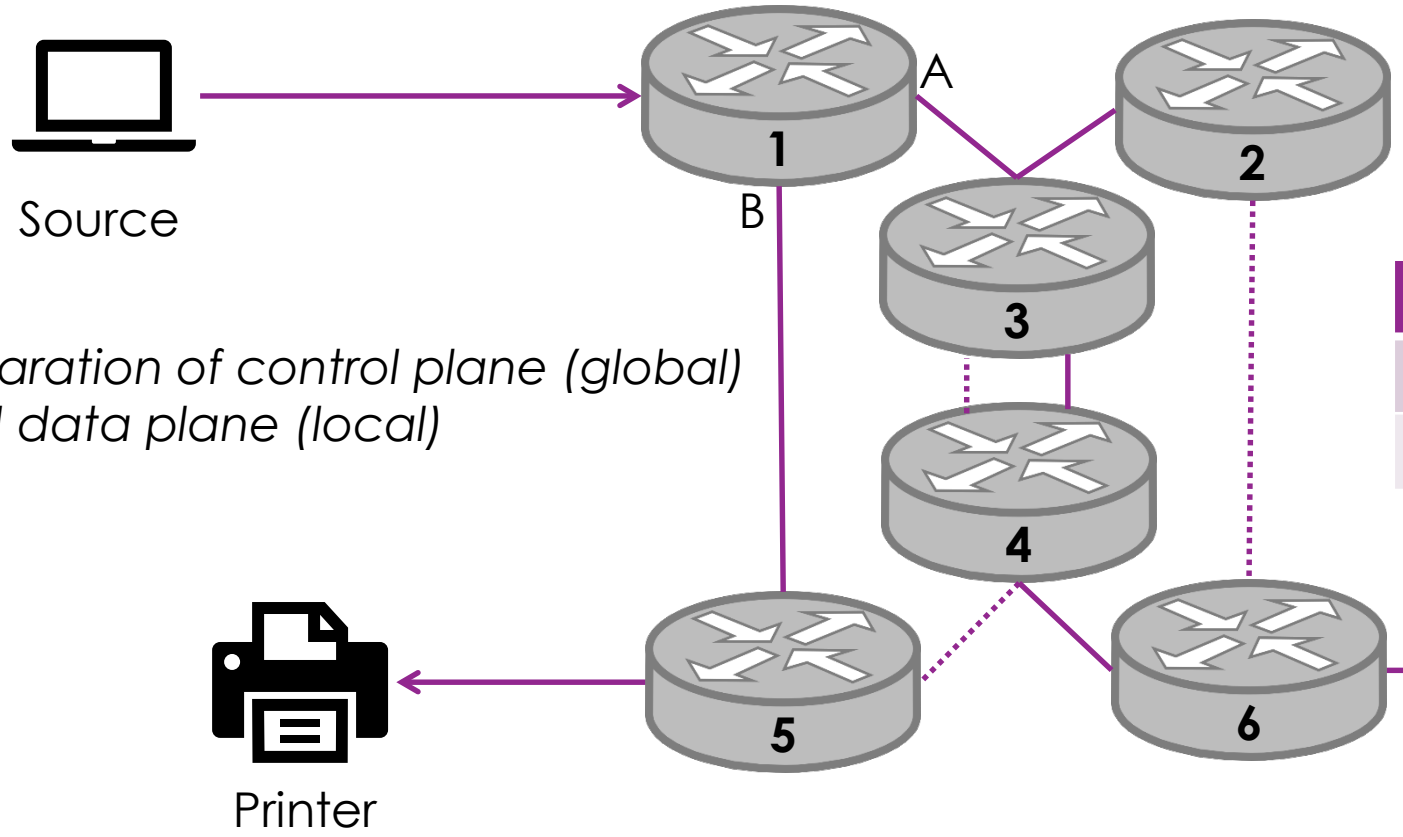
- What guarantees does your application need?
- Which layer provides it?

- Connectionless, packets
  - Go where the network chooses, in realtime

- Connection-oriented, circuits
  - In a packet-switched world – 'virtual' circuits.
  - Go where I configured the network to send them

- Need to understand how packets get sent towards their destination

Packet **Forwarding** and **Routing**



Separation of control plane (global) and data plane (local)

Router 1: Next Hops (now)

Target	Port
Server	A
Printer	B

Forwarding Table

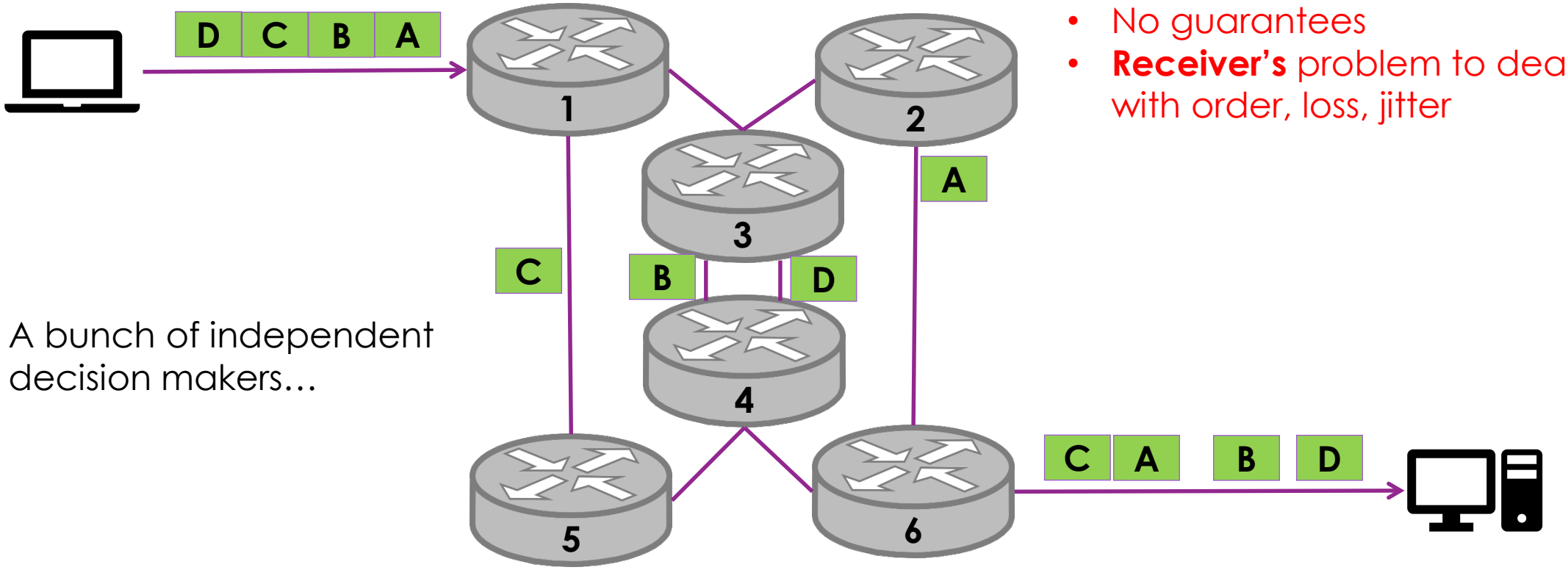
All Routers:

Target	Port
Server	X
Printer	Y

Routing

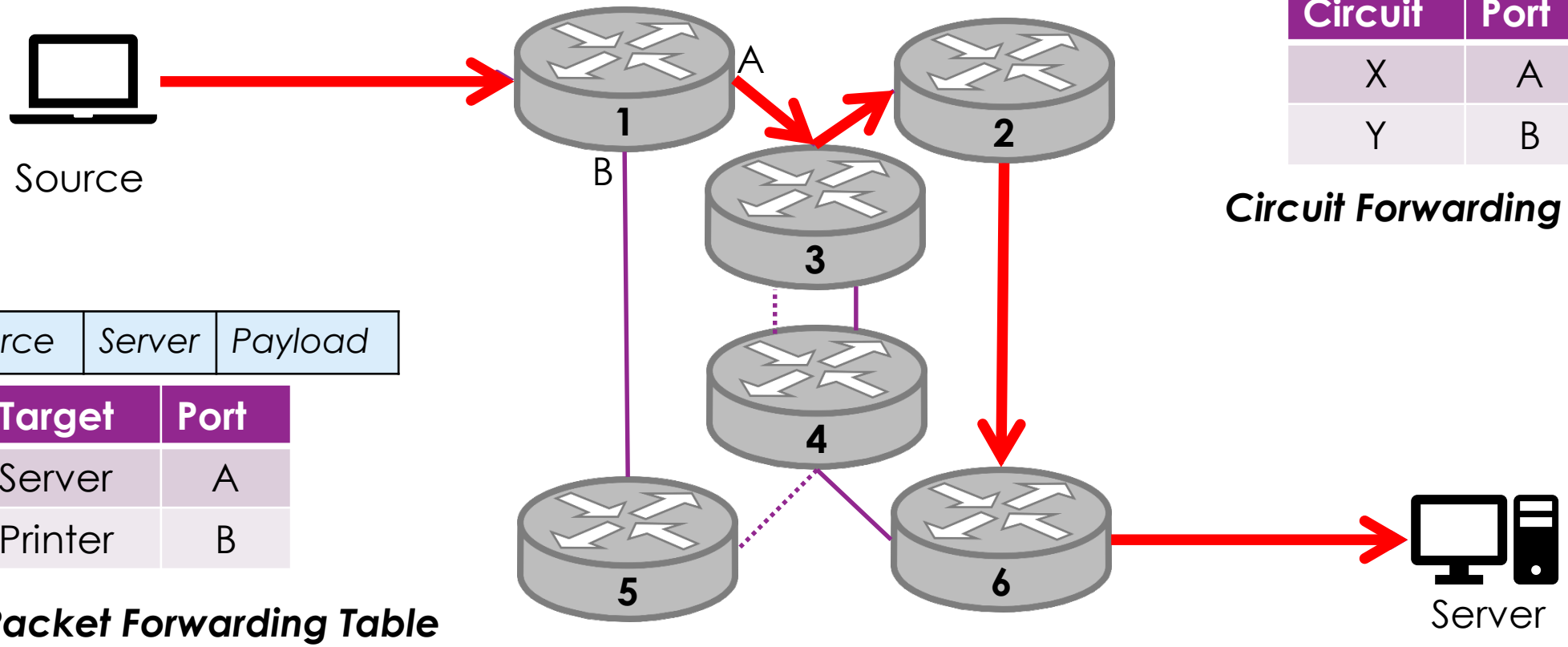
Multi-path packet forwarding

- Statistical multiplexing
- Unpredictable ordering
- Variable delays
- No guarantees
- **Receiver's** problem to deal with order, loss, jitter



A bunch of independent decision makers…

Circuits in a Packet world??



Source	Server	Payload

Target	Port
Server	A
Printer	B

Packet Forwarding Table

Circuit ID	Payload

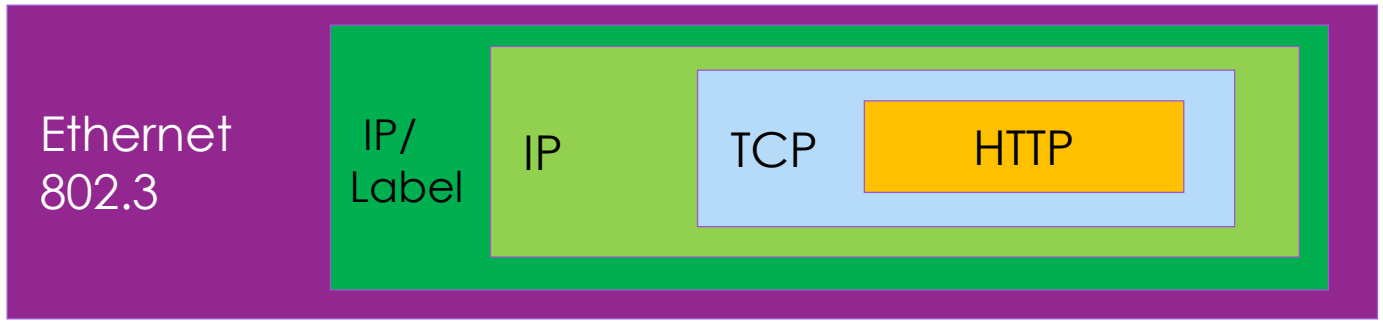
Circuit	Port
X	A
Y	B

Circuit Forwarding Table

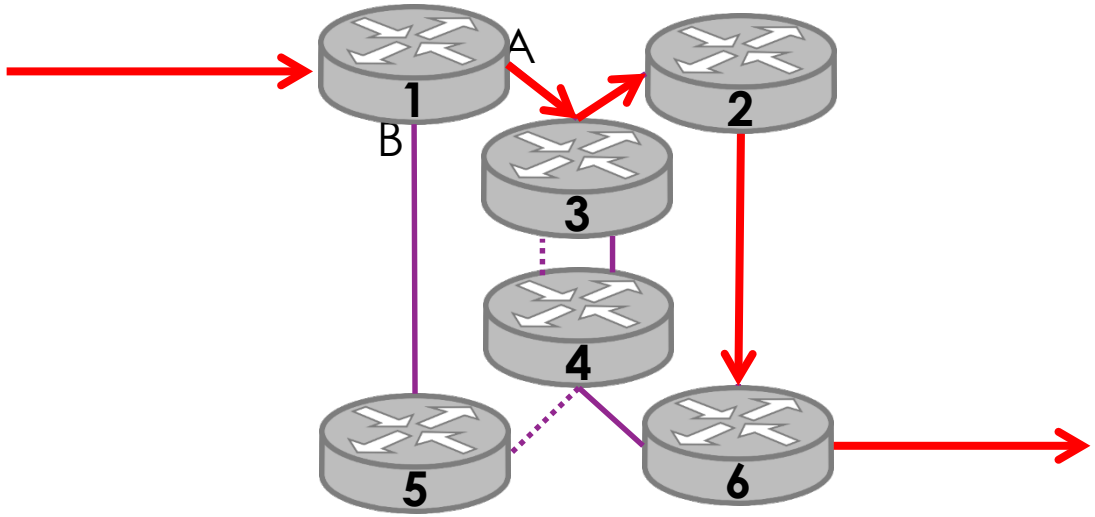
Circuits over packets

- Why???
  - Guaranteed path
  - Guaranteed (maybe) bandwidth/performance

- How?
  - Circuit set-up and tear-down: manual, or on-demand
  - Packets: More encapsulation – *IPinIP*, *Multi-Protocol Label Switching (MPLS)*



Circuits vs packets



	Packets	Circuits
Path router control	Not needed	Required
Prior Setup	Nothing needed	Required
Router State	Per destination	Per circuit configuration
Addressing	Packets carry full src/dest	Packets carry short label
Forwarding	Per packet	Per circuit
Router failure	Packets lost, reroute	Circuit fails completely
Quality of service	Difficult	Easy(*)
Security	Per-packet, other layers	Maybe…

(The) Internet design

- Standards: the **Internet Engineering Task Force** (IETF.org)
  - Just a bunch of people, arguing. Not a company, no board, no members
  - Lots of Working Groups, under Areas
  - Work revolves around 'drafts' and 'request for comments' (RFC)

- **RFC**
  - Strict rules about structure, references, and language (MUST/SHOULD/MAY)
  - Standards Track *or*
    - Best Current Practice, Informational, Experimental, Historic (Lost interest or *Detrimental*)
  - Locked down on publication. Regularly Obsoleted or Updated
  - RFC-0001: April 1969, RFC-8571: March 2019
    - *Watch out for 1 April RFCs…*

Taming the crowds

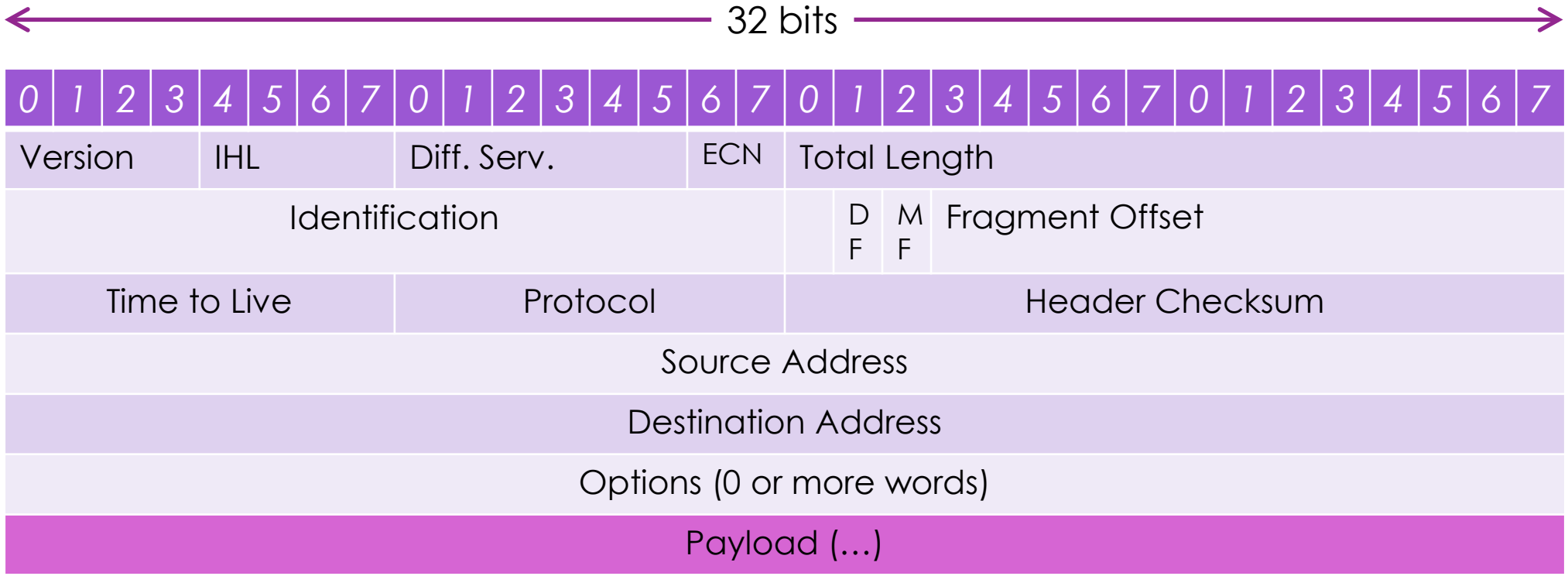
- IETF needs some structure:
  - ISOC: Internet Society – international, non-profit, legal entity
  - IESG: Internet Engineering Steering Group – oversees IETF processes, signs off.
  - IAB: Internet Architecture Board – Big picture view, identify/review issues
  - IRTF: Internet Research Task Force – Researches issues…
    - Overseen by IRSG: Internet Research Steering Group
  - IANA: Internet Assigned Number Authority – Directory keeper
    - Contracted to ICANN: Internet Corporation for Assigned Names and Numbers
  - RFC Editor

IETF "principles"

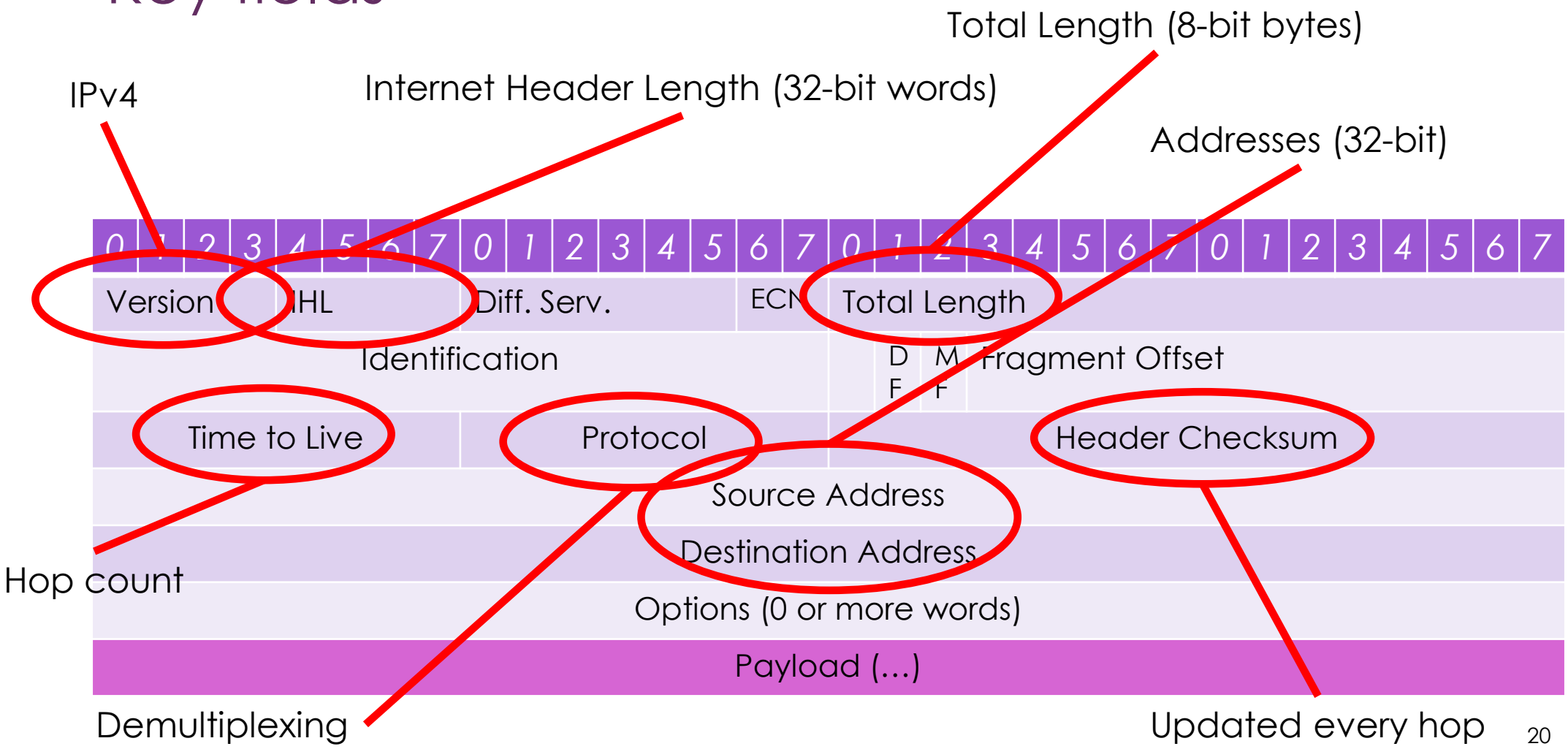
- End-to-end…
- *"Rough consensus and running code"*
- *"Be conservative in what you send, liberal in what you accept"*
- Simplicity, clarity,
  - Fight feature creep, use other layers, offer just one way to do something
  - Don't hardwire too much, let it be negotiated
  - Aim for good, broad design; let others deal with edge-cases
- Think about scalability, non-linearity, heterogeneity, cost
  - Law of Unintended Consequences
  - https://www.rfc-editor.org/rfc/rfc3439.txt *(updates RFC1958)*

Introducing… The Internet Protocol!

Read left-to-right, top to bottom



Key fields


IPv4
Internet Header Length (32-bit words)
Total Length (8-bit bytes)
Addresses (32-bit)
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
Version
IHL
Diff. Serv.
ECN
Total Length
Identification
D F
M F
Fragment Offset
Time to Live
Protocol
Header Checksum
Source Address
Destination Address
Options (0 or more words)
Payload (...)
Hop count
Demultiplexing
Updated every hop

IP addressing

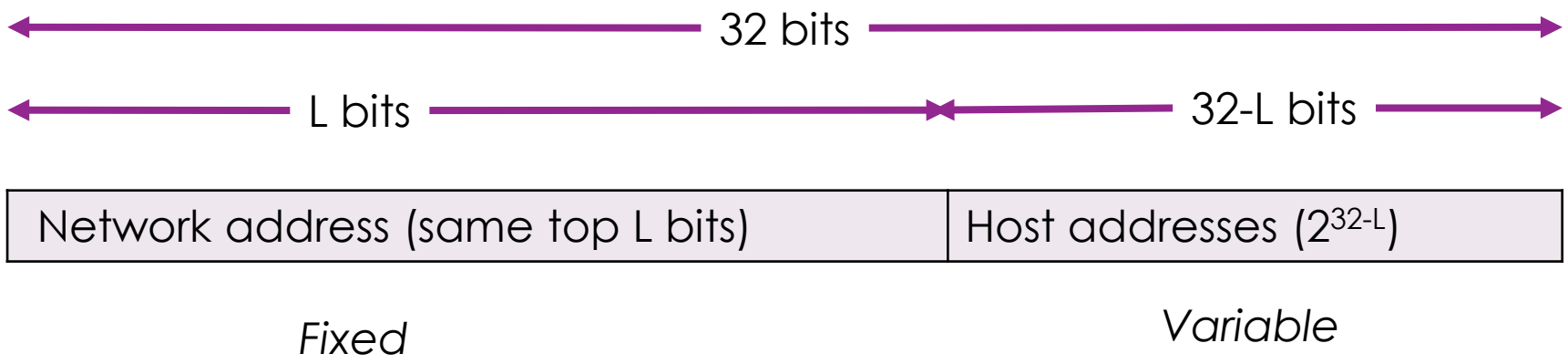
- 32-bits = 2^{32} hosts = ~4billion - in theory
- Written in 'dotted-quad' notation
  - i.e. four numbers, separated by dots

- 11010101|11110000|10101010|00001111

- 213.240.170.15
- Not a host, but an interface
  - *1 IP = 1 MAC most of the time…*

IP prefixes

- Aggregate 'nearby' addresses into a *block* for routing (tables)
- A block of addresses is described by its *prefix*
- Split 32-bits into network and host components using upper **L** bits:



IP prefixes

- Use a '/' ('slash') notation:

- *Network address/prefixlength*:
  - Network address is the first 'host' in the host-range

- For example: 150.203.0.0/16
  - From 150.203.**0.0** up to 150.203.**255.255**
  - 32 bits, using 16 for the prefix: $2^{32-16} = 2^{16} = 65{,}536$ addresses

- A "/24" has 256 addresses *[e.g. 150.203.0.0/24 = 150.203.0.0-150.203.0.255]*
- A "/30" has 4 addresses

IP **subnets**

- *Network address/prefixlength*

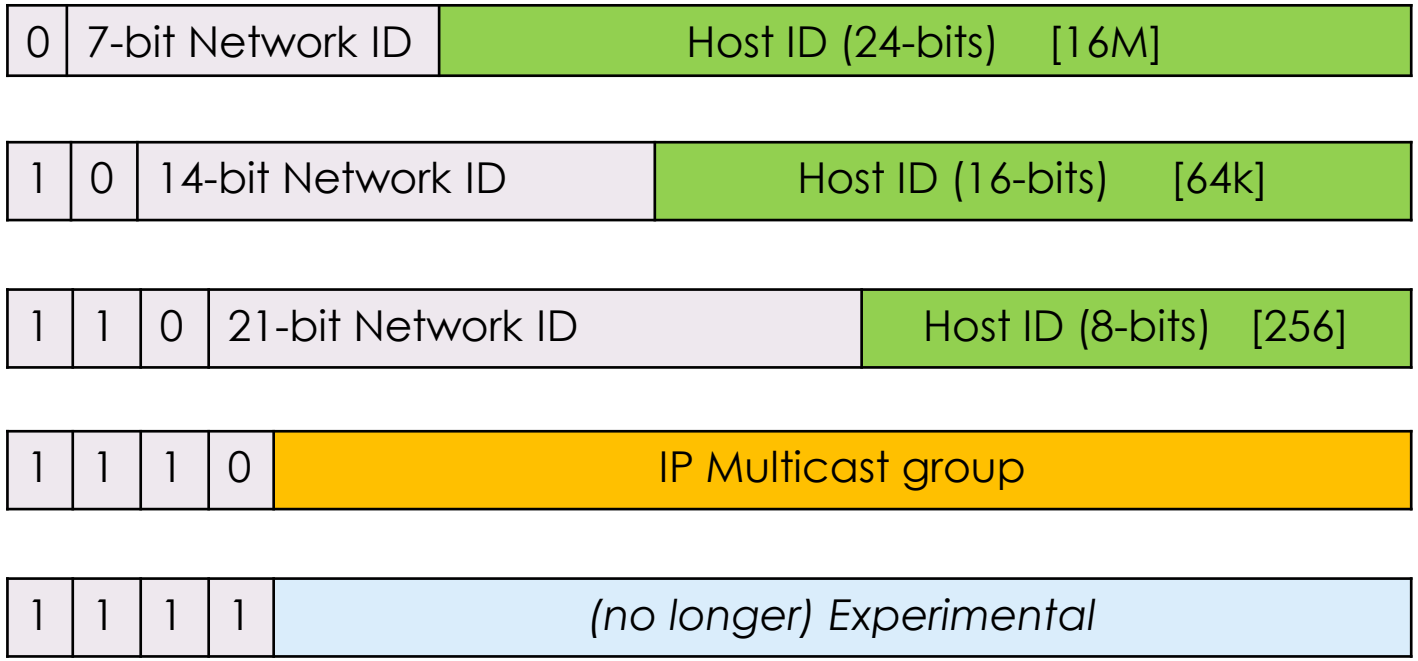
- Network address = a subnet (a block of contiguous host addresses)
- Prefix length = a subnet mask

- For example: 150.203.10.0/24
  - The 150.203.10.0 **subnet**
  - /24 = 24-bit network id so **mask**= 24 1's and 8 0's: 255.255.255.0

IP address classes

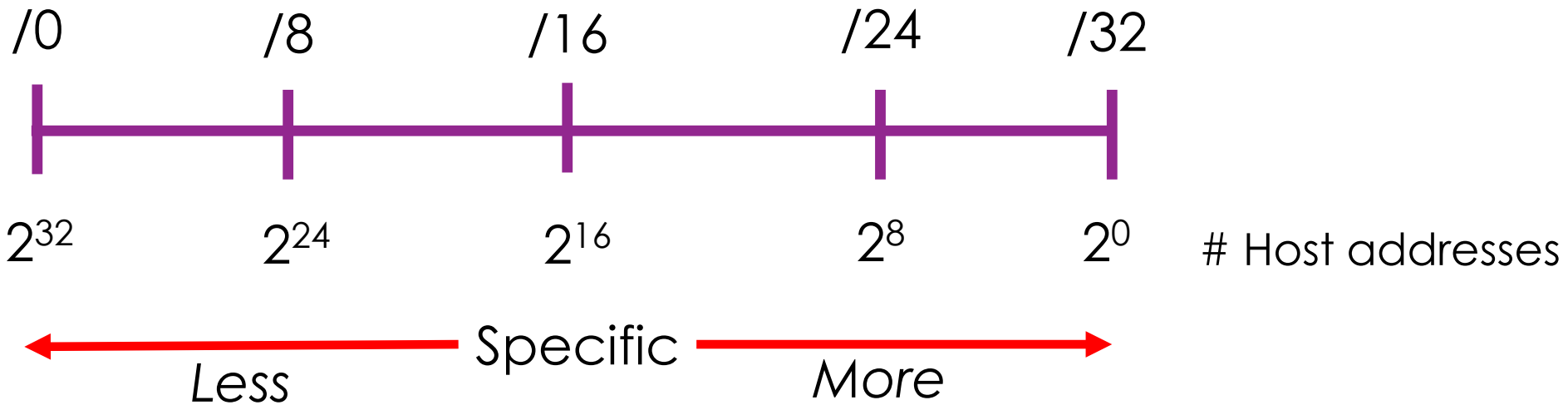
- Largely historical, but still common language

- **Class A**: /8
  - First byte: 1-126
- **Class B**: /16
  - First byte: 128-191
- **Class C**: /24
  - First byte: 192-223
- **Class D**:
  - First byte: 224-239
- **Class E**:
  - First byte: 240-255



More or Less?

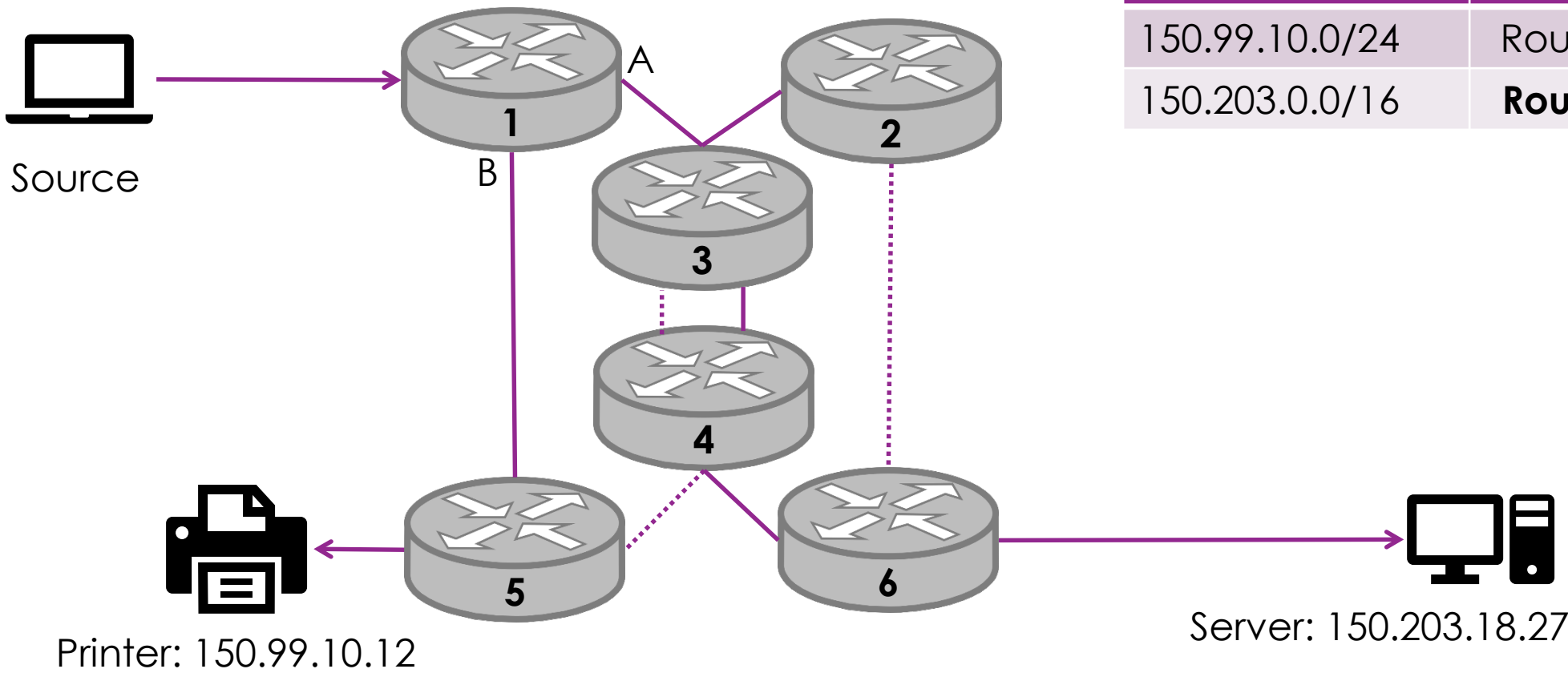
- A "More-specific" prefix = longer prefix = fewer hosts
- A "Less-specific" prefix = shorter prefix = more hosts



Forwarding, by prefix

Router 1 Forwarding table

Target	Next Hop
150.99.10.0/24	Router 5
150.203.0.0/16	**Router 4**



Forwarding by longest matching prefix

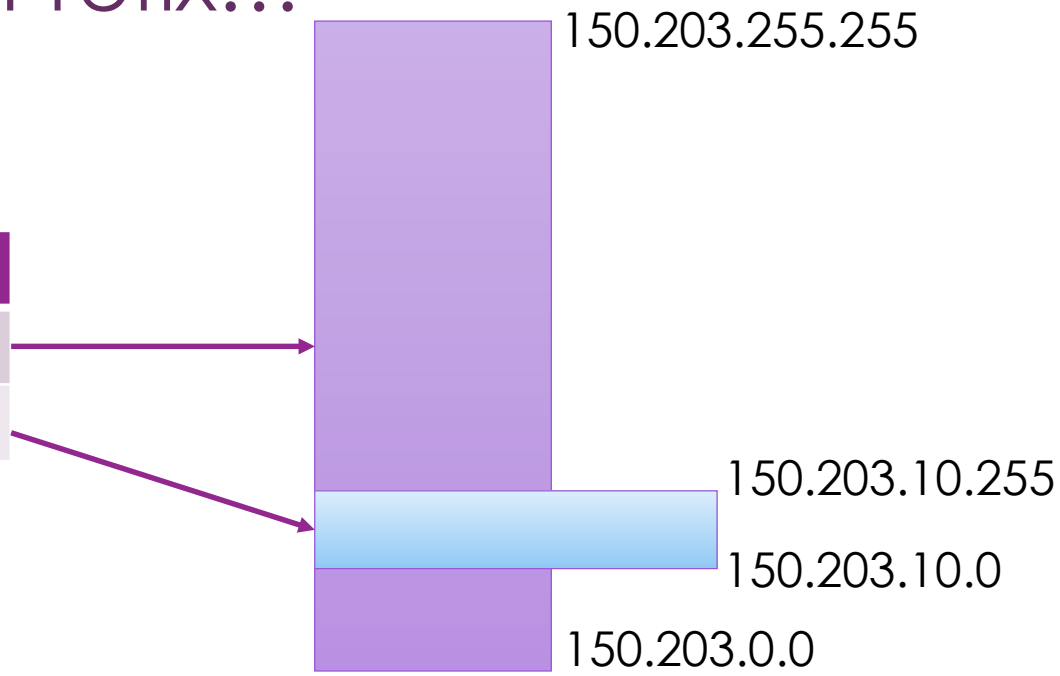
- Prefixes in a forwarding table are allowed to overlap
  - For good reasons!
  - Aggregation benefit of hierarchical addressing *(e.g. a /20 holds sixteen /24's)*
  - As well as flexibility to direct some specific traffic

- Longest matching prefix rule:
  1. For each packet, identify all subnet prefixes that apply
  2. Select the one with the longest matching prefix
     - The 'most specific'
  3. Forward accordingly to the next hop

Longest Matching Prefix…

Router Forwarding Table

Target	Next Hop
150.203.0.0/16	Router 5
150.203.10.0/24	Router 4



- 150.203.8.99 goes to … Router 5
- 150.203.100.6 goes to … Router 5
- 150.203.10.200 goes to … **Router 4**

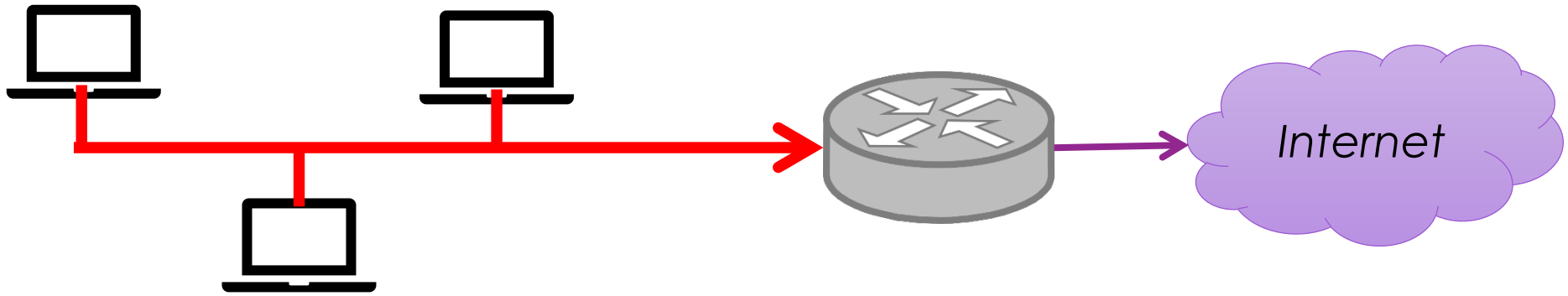
Why?

- Provide default behaviour with shorter (less-specific) prefixes
  - Catches more host-addresses in a single block
- Support specialised behaviour with longer (more-specific) prefixes
  - Key services to be reached via
    - Higher performance paths
    - Lower cost paths
    - More secure paths
    - … (policy reasons)
- Hierarchy generates more compact forwarding tables on routers
  - Cost of lookups vs simple tables is largely optimised away now

Hosts as routers?

- How does a host decide how to send a packet?
  - Assume it has learnt the destination IP address from somewhere
- Hosts are not good routers – keep it simple, let routers route!

- Two types of destination
  - On my LAN – **use LAN services**
  - Beyond the LAN – **use a router**



Host forwarding table

- How to decide?
- Longest matching prefix plus a catch-all address:
- 0.0.0.0/0 = 'the whole internet'

- Host knows its IP address and its prefix (subnet mask):
  - "I'm 150.203.56.99 and I'm on a /24"
  - So my network is 150.203.56.0/24

Target	Next Hop
150.203.56.0/24	Direct on my LAN
0.0.0.0/0	My (default) Router *150.203.56.1*

Longest matching prefix

…which is also on my LAN

My Forwarding table

- Lots of interfaces
- Default route
  - My router
- LAN route
  - 192.168.178/24
  - My subnet
  - All "On-link"
- All learned routes
  - No static or "persistent" routes

```
C:\Users\User>netstat -r
===========================================================================
Interface List
 24...00 50 b6 67 be bf ......USB Giga Ethernet
 22...00 ff db e1 48 f8 ......Speedify Virtual Adapter
 15...02 00 4c 4f 4f 50 ......Npcap Loopback Adapter
  9...7c 7a 91 75 36 46 ......Microsoft Wi-Fi Direct Virtual Adapter #5
 19...7e 7a 91 75 36 45 ......Microsoft Wi-Fi Direct Virtual Adapter #6
  6...7c 7a 91 75 36 49 ......Bluetooth Device (Personal Area Network) #2
  7...7c 7a 91 75 36 45 ......Intel(R) Wireless-N 7260
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0    192.168.178.1   192.168.178.34     25
        127.0.0.1  255.255.255.255          On-link        127.0.0.1    331
      169.254.0.0      255.255.0.0          On-link    169.254.164.48    281
    192.168.178.0    255.255.255.0          On-link    192.168.178.34    281
   192.168.178.34  255.255.255.255          On-link    192.168.178.34    281
  192.168.178.255  255.255.255.255          On-link    192.168.178.34    281
        224.0.0.0        240.0.0.0          On-link        127.0.0.1    331
        224.0.0.0        240.0.0.0          On-link    192.168.178.34    281
[...]
===========================================================================
Persistent Routes:
  None
```

Home on the LAN

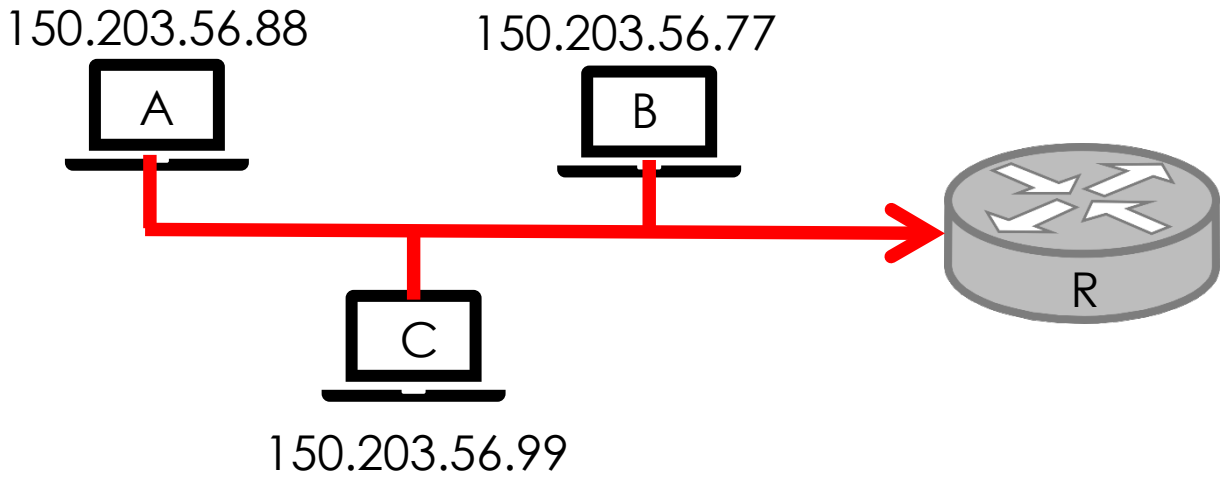
- Network-Layer:
  - *Hey, I'm on the same Ethernet as my target, I can just send this packet directly*
  - *Link layer, send this to IP-address 150.203.56.99*

- Link-Layer:
  - *What's an IP address????*
  - *I need a MAC (Link Layer) address*
  - *Network-layer won't help me*

- Need to cross layers. Need some kind of Address Resolution Protocol

The Address Resolution Protocol (ARP)

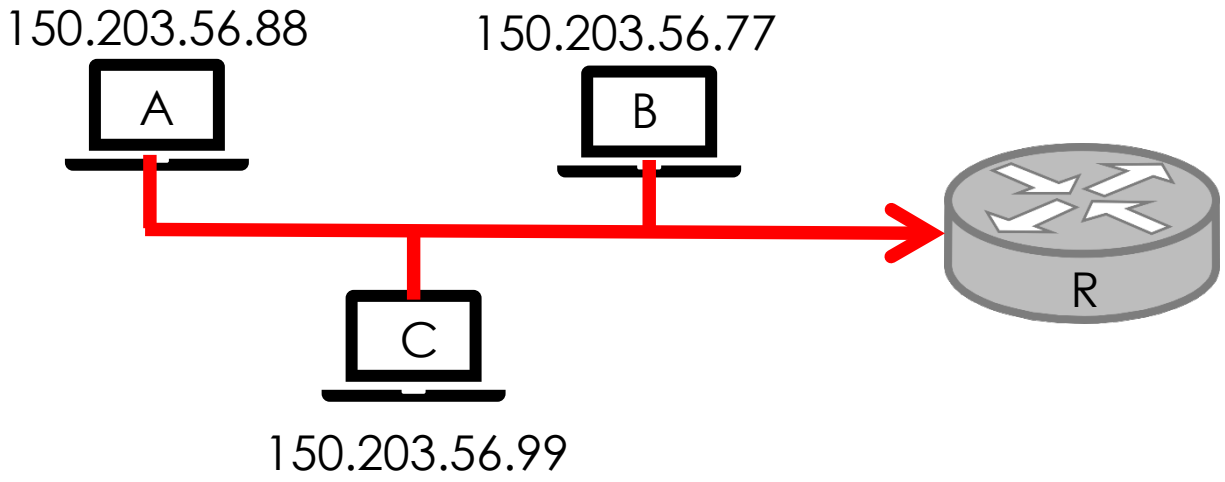
- RFC 826 (and updates)
- *Mapping IP addresses to Ethernet/etc. hardware addresses*
  - Not an IP packet – Link Layer



- A wants to send IP packet to C
  - Send a Link layer **broadcast**
    - Src MAC = AA:AA:AA:AA:AA:AA
    - Dest MAC = FF:FF:FF:FF:FF:FF
    - I am/Tell 150.203.56.88
    - Who has 150.203.56.99?

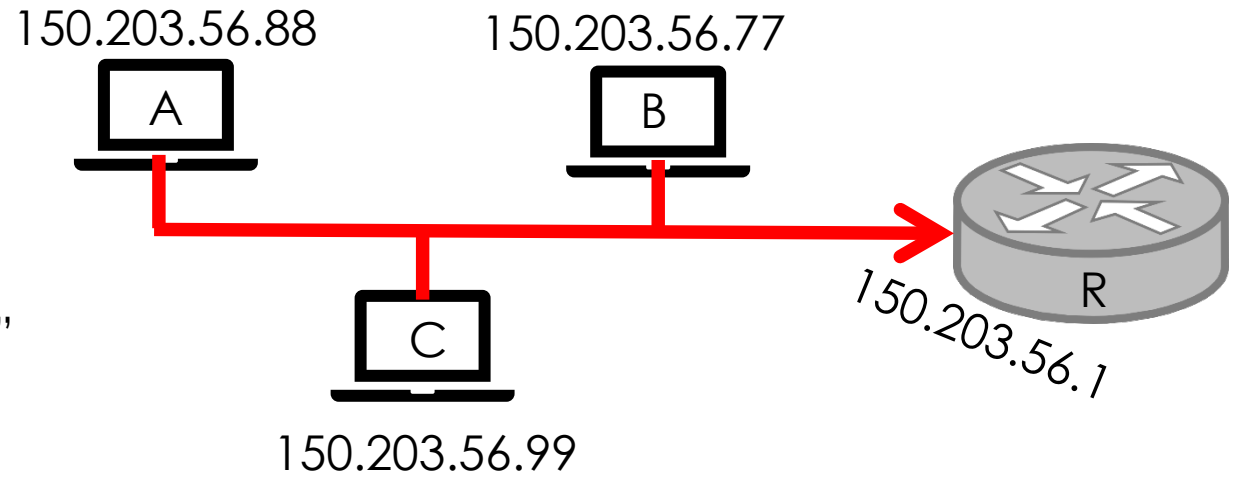
The Address Resolution Protocol

- B receives the broadcast
  - And ignores it…
- C receives the broadcast
  - And replies (Link layer)
    - Src MAC = CC:CC:CC:CC:CC:CC
    - Dst MAC = *AA:AA:AA:AA:AA:AA*
    - Dear 150.203.56.88
    - I am 150.203.56.99
- A gets the reply
  - And can now send directly



The Address Resolution Protocol

- Some optimisations:
  - Caching, with timeouts
  - Catch passing ARP information
    - B doesn't ignore the broadcast
  - Tell everyone of your changes
    - Gratuitous ARP – "look for yourself"
    - Also helps find duplicate IPs
- Also applies to packets going to/through the router
  - *Need MAC address of R*



ARP is a simple Discovery Protocol

ARP cache

- *Dynamic*
  - Learned
- *Static*
  - Configured
- 2 interfaces
  - WiFi, Ethernet
- 192.168.178/24 subnet
- Some special addresses

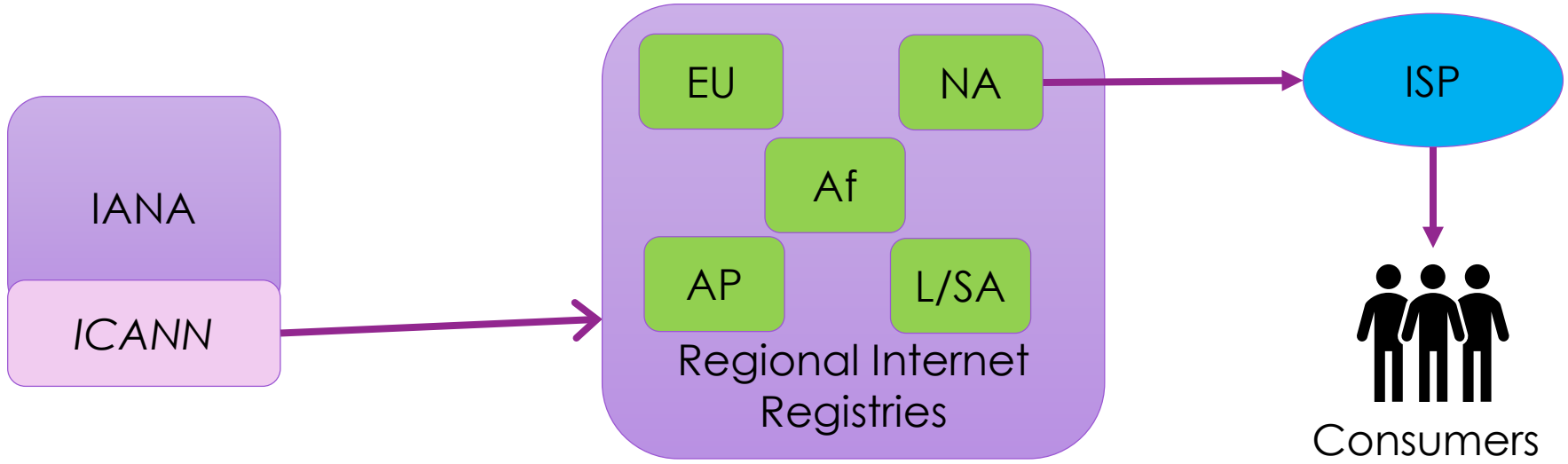
```
C:\Users\User>arp -a

Interface: 192.168.178.47 --- 0x7
  Internet Address      Physical Address      Type
  192.168.178.1         38-10-d5-bc-be-99     dynamic
  192.168.178.32        e8-5b-5b-64-13-62     dynamic
  192.168.178.35        1c-1b-0d-72-01-52     dynamic
  192.168.178.38        b8-ca-3a-e2-5b-6f     dynamic
  192.168.178.39        f0-bf-97-9c-84-7e     dynamic
  192.168.178.44        00-11-32-6d-d8-69     dynamic
  192.168.178.50        00-19-1e-b0-02-87     dynamic
  192.168.178.63        3c-bd-d8-26-f7-ac     dynamic
  192.168.178.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  230.0.0.1             01-00-5e-00-00-01     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 169.254.164.48 --- 0xf
  Internet Address      Physical Address      Type
  169.254.255.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  230.0.0.1             01-00-5e-00-00-01     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Getting addresses (blocks) for your network

- Need to consider
  - Globally-unique allocation
  - Routing aggregation opportunities
  - **Politics**
- Need an authority, which scales



Regional Internet Registries

Addresses are not equal

- IPv4 – 2^{32} addresses – can't use all of them.
- Special allocations
  - **Private Networks**: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
    - Can be used on networks (that are/are NOT) connected directly to the Internet
  - **IP Multicasting**: 224.0.0.0/4 [old class-D]
    - Distribute packets to groups of subscribers
    - Requires additional services on the network
  - **Experimental**: 240.0.0.0/4 [old class-E, 200M addresses!]
    - Still waiting for an experiment
    - Most OS will drop such packets

Addresses are not equal (2)

- Special networks:
  - **This host** on this network: 0.0.0.0/8
    - Only used as a source address
    - Used for 'any interface' or 'I do not know'
  - **Local interface**: 127.0.0.0/8
    - Loopback interface: 127.0.0.1
  - **Link-local**: 169.254.0.0/16
    - My LAN when all else fails
  - **Broadcast**: 255.255.255.255
    - Specific address for a global broadcast
      - In theory…

Address conventions

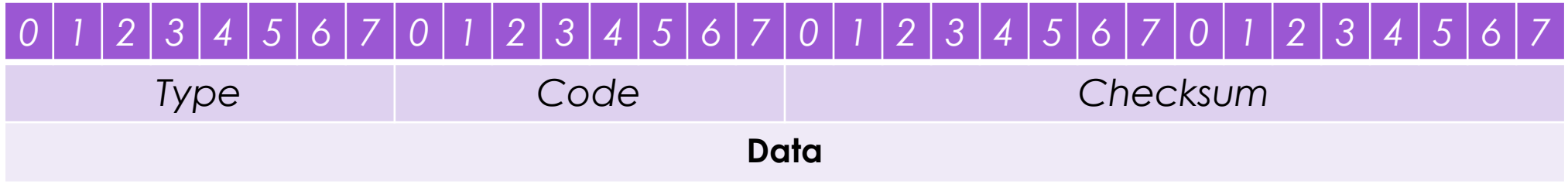
- Subnet broadcast: A.B.C.255
  - All ones in the host field (.255 for /24, .255.255 for /16)
- The subnet: A.B.C.0
  - Aka "the wire", usually followed by /n
  - All zeroes in the host field
- Router/gateway: A.B.C.1
  - A convention. Makes it easy to find
- Note: Host field is shorter than 8 bits, for prefixes >24
  - *150.203.56.0/28: (Sub)Netmask=255.255.255.240, Broadcast=150.203.56.15!*

Getting feedback from the Internet

- Sometimes things happen to packets
  - Along the route
  - Loss, corruption, mistakes, …
    - Wrong addresses, nobody home, packet malformed, …
- Sender needs to know what happened
  - With little/no feedback from receiver
  - Retransmission may be wrong
    - Don't keep making the same mistake: "Internet says NO!"
- Internet Protocol needs some Control Messages

Internet Control Message Protocol (ICMP)

- IP packet – protocol #1
- Designed mainly for routers to inform senders (including routers)
  - Senders listen, but don't (usually) send
- 'Type': category. 'Code': actual problem/question/answer
- Data =
  - Header of packet that caused the problem, and 8+ bytes of payload
  - Other information related to the problem/request

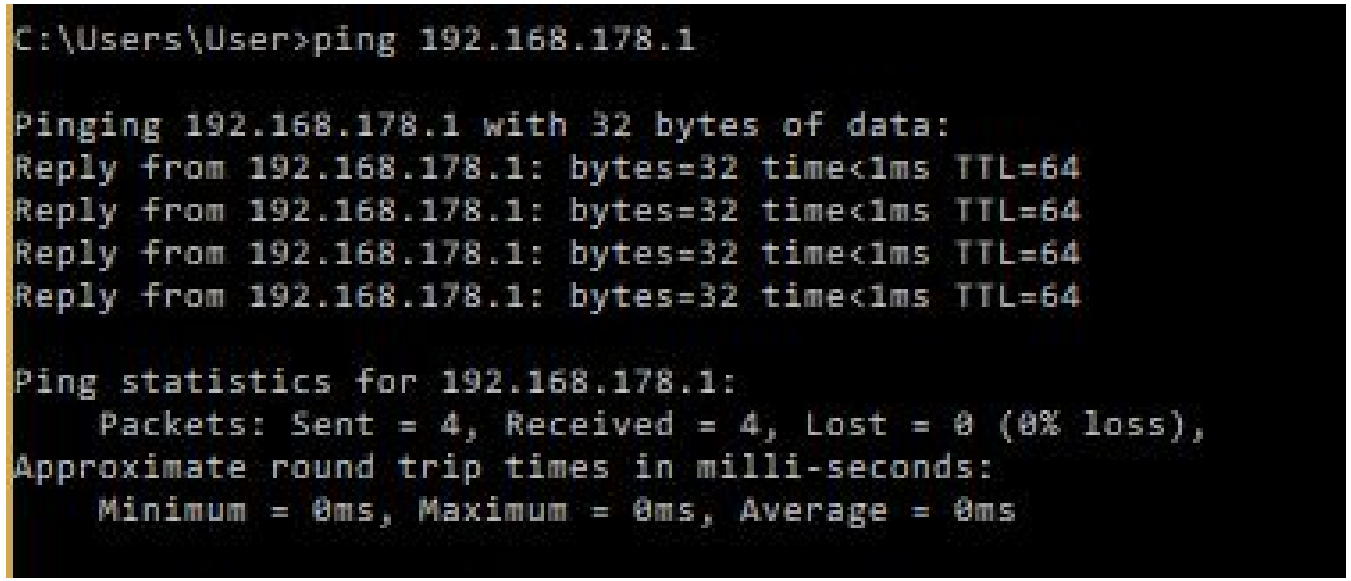


ICMP Types

- Type 0,8 – ICMP Echo
- Type 3 – Destination Unreachable
  - Many reasons, at intermediate routers, final router, host
- Type 4 – Source Quench
  - Please Slow Down! (deprecated)
- Type 5 – Redirect
  - Looks elsewhere
- Type 9,10 – Router discovery
- Type 11 – Time exceeded
  - E.g. TTL has hit zero
- Type 12 – Bad header
- Type 13,14 – Timestamp
- Type 15+ - Deprecated, Experimental, Unallocated and Reserved

ICMP use by hosts?

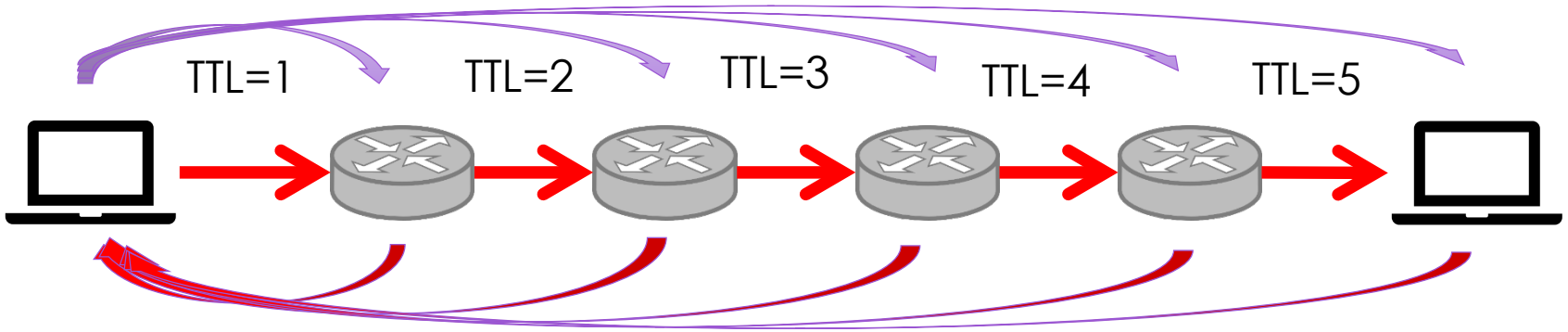
- **Ping**
- Send an ICMP Echo-request (Type 8/Code 0) to an IP address
- If received, receiver sends back an ICMP Echo-reply (Type 0/0)



- Useful for testing (many options) – and for probing…

Traceroute

- Identify all the routers through which your packets are going (now)
- Use ‘TTL decrement’ and ‘ICMP Time Exceeded’ (Type 11/0)
  - Replies include IP of router that hit zero



ICMP TTL expired (11/0) at router X …

Traceroute

- Really useful to identify path, intermediate devices and distances
- And to probe internal networks…

```
C:\Users\User>tracert  -d www.bbc.co.uk

Tracing route to www.bbc.net.uk [212.58.244.27]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  192.168.178.1
  2     5 ms     5 ms     4 ms  150.101.32.77
  3     5 ms     5 ms     5 ms  150.101.34.134
  4     5 ms     5 ms     6 ms  150.101.33.6
  5     9 ms     9 ms     9 ms  150.101.33.102
  6    12 ms     9 ms    10 ms  203.29.134.4
  7   189 ms   189 ms   189 ms  213.248.86.188
  8   255 ms   255 ms   255 ms  62.115.119.228
  9   363 ms   331 ms   331 ms  62.115.136.184
 10   316 ms   340 ms   317 ms  62.115.136.193
 11   330 ms   332 ms   323 ms  62.115.141.197
 12   321 ms   320 ms   320 ms  62.115.144.159
 13     *        *        *     Request timed out.
 14   323 ms   323 ms   323 ms  132.185.254.105
 15   323 ms   323 ms   323 ms  132.185.255.148
 16   321 ms   321 ms   321 ms  212.58.244.27

Trace complete.
```

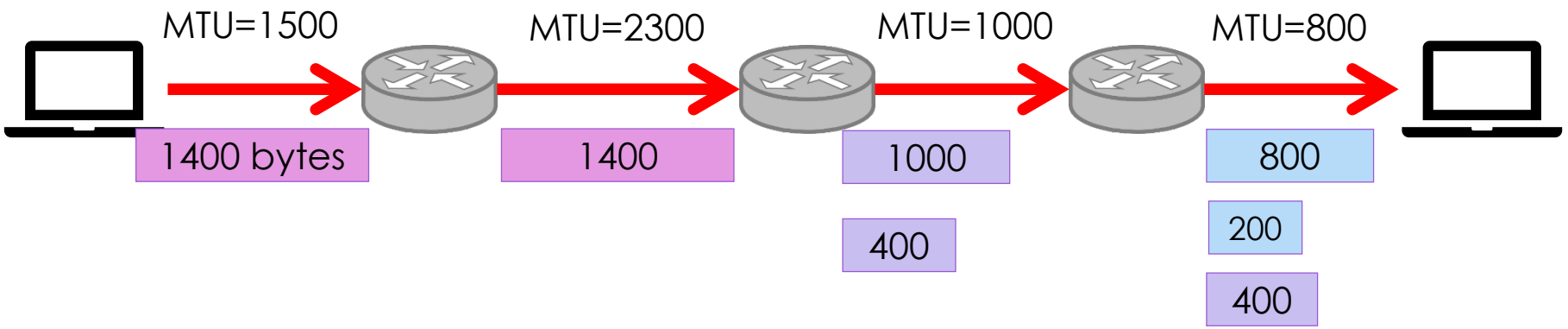
3 attempts each hop

RTT increases in jumps (within variations)

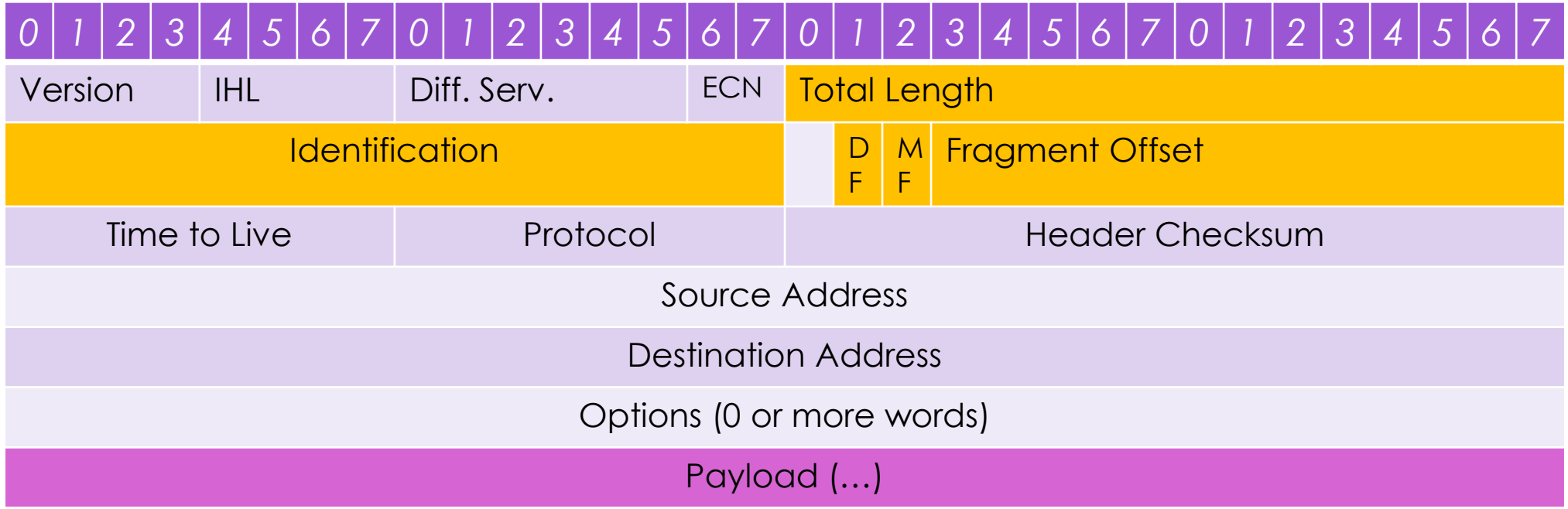
#13 is a bit shy…?

Big packets

- Bigger packets are more efficient, but can be 'too big'.
- What's a 'big' packet?
- Something bigger than the payload of your LAN
  - **Maximum Transmission Unit (MTU)**
  - Ethernet: 1500bytes, WiFi: 2300bytes
  - Leads to **Fragmentation**



IP Fragmentation



- Identification = key to identify a packet uniquely
- MF = More Fragments (flag)
- DF = Don't Fragment (flag)

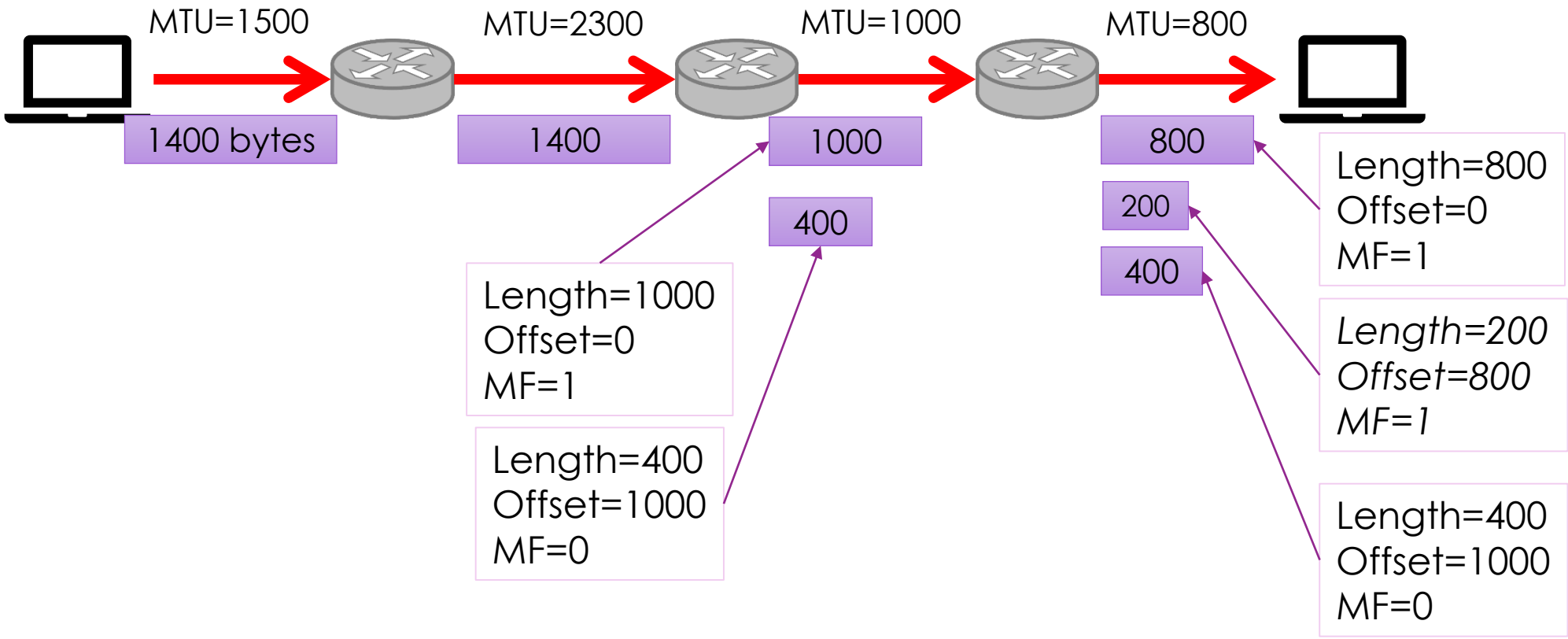
- Total Length = of **this** packet
- Fragment Offset = position within original

Router Fragmentation Process

- Incoming packet of size > outbound MTU
- Split packet into (large) new packets
- Copy IP Header to each new packet – including the Identification
- Adjust Length field for each packet
  - And Checksum, and TTL
- Set Offset to identify location within overall packet
- Set MF flag on all packets, except the last one

- Receiver collects all fragment-packets and reassembles

Fragmentation example



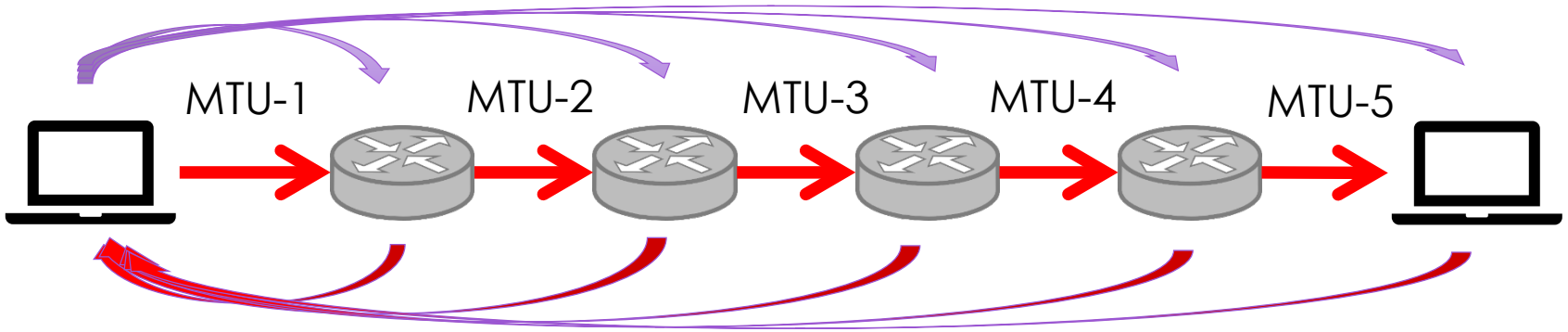
It works, but…

- Has been used since the beginning of IP, and works well

- Creates performance issues
  - More work for routers and receivers
  - Increased probability of (total) packet loss
    - No retransmission of fragments
  - Security issues
    - Easier to hide malicious traffic
    - Harder for Deep Packet Inspection

Better approach

- Test the network and send the smallest big-packet you can
- **Path MTU Discovery**
- Looks like traceroute – but use packet sizes and DF=1

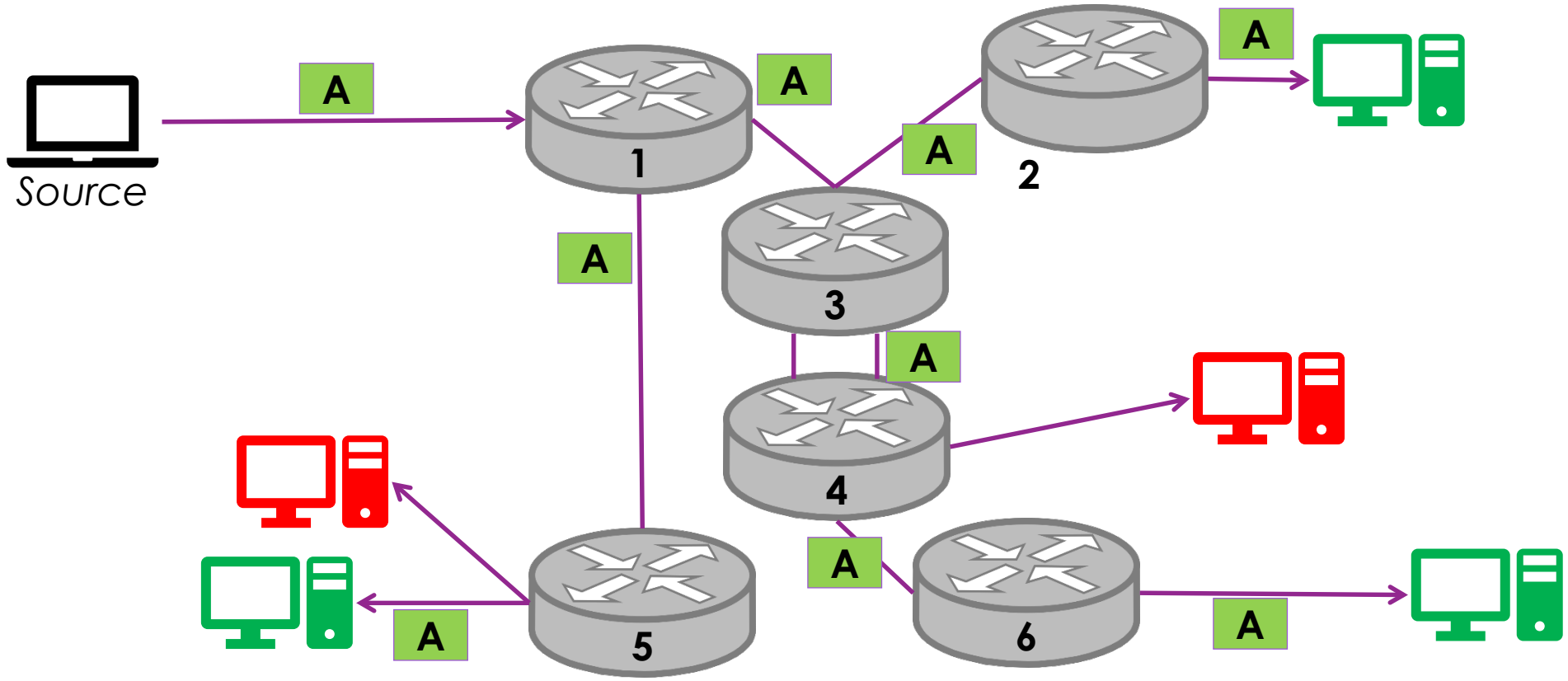


ICMP Destination unreachable (Type 3)
Fragmentation required, and DF flag set (Code 4)
Data = next-hop MTU

IP Multicast

Packets sent to **all** ***group*** subscribers (224/8)
- sender sends once

Internet Group Membership Protocol (IGMP)



IPv6!

- When IPv4 just won't do it anymore

- IPv4 designed in a smaller, more scalable and way more trusting world
- Never considered planetary wide participation
- Never considered major infrastructure role
- Never considered IoT, smart devices, mobility, …
- Never considered bad people

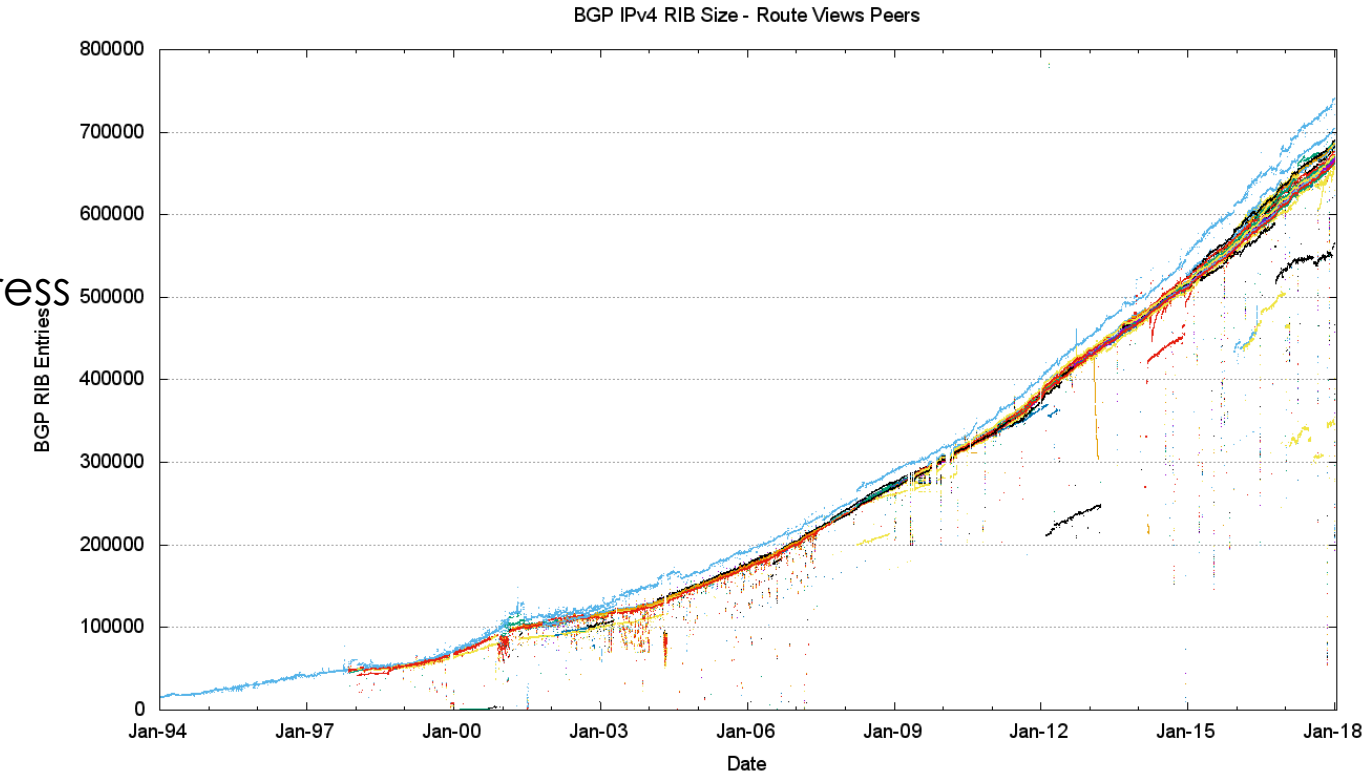
- We now have a problem
  - Several
  - But especially the need for more than 4 billion devices

IPv4 is over

- A common catchphrase
- However…
- Addresses are largely exhausted
  - RIR's ran out 2011-2015
  - Lots of wasted address space
  - Re-allocating ever-smaller chunks (/30)
    - With tighter rules
  - Can't aggregate address blocks for routing
    - Forwarding tables are immense

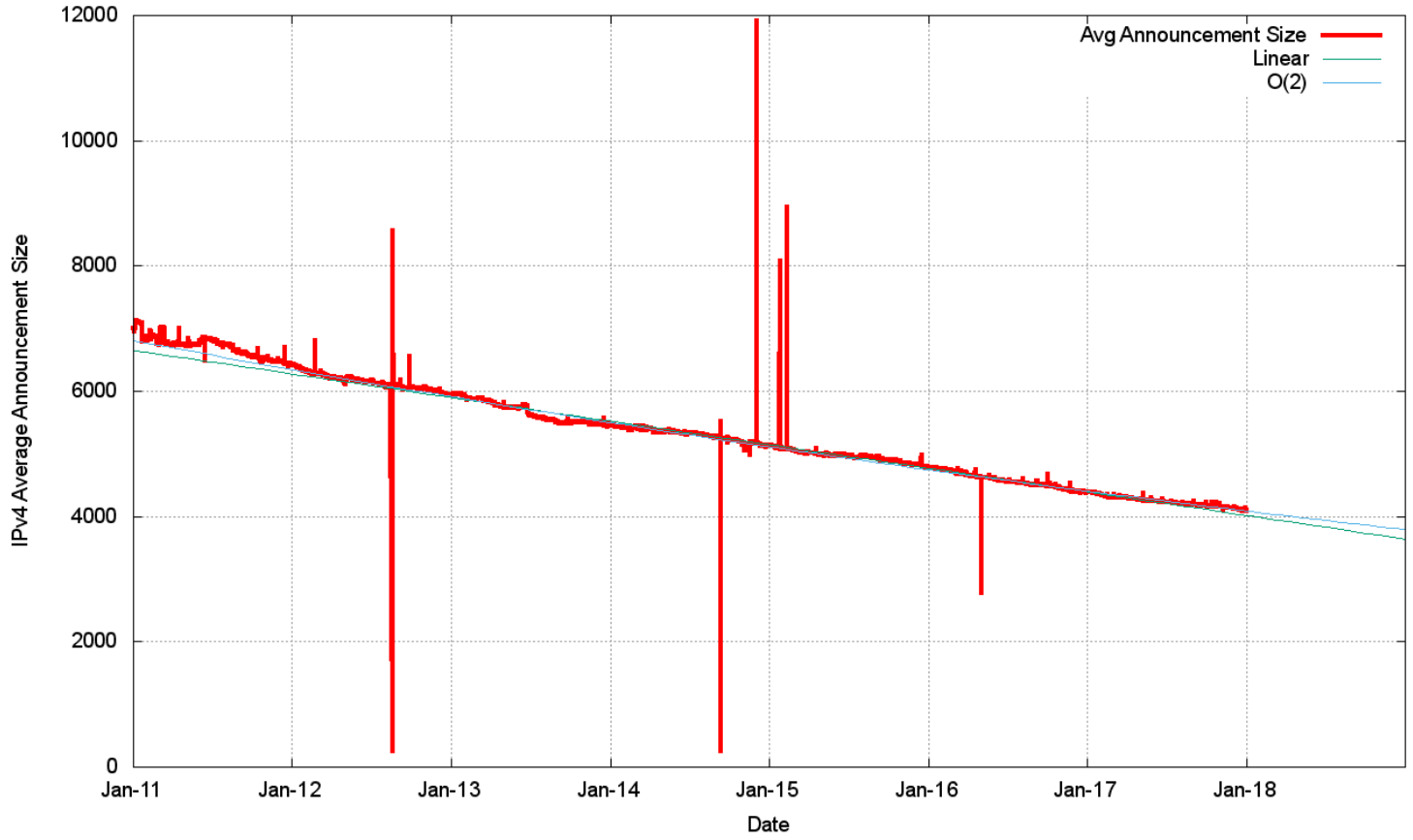
The routing problem

- Every router has a forwarding table
  - Fortunately only 10-100 interfaces
- Across the whole internet
  - Lookup tables of ~1M entries
  - Fuzzy matching on 32bit address
  - In under 5ns (100Gb/s)
- And 170k updates/day
  - 2 per second



The routing problem

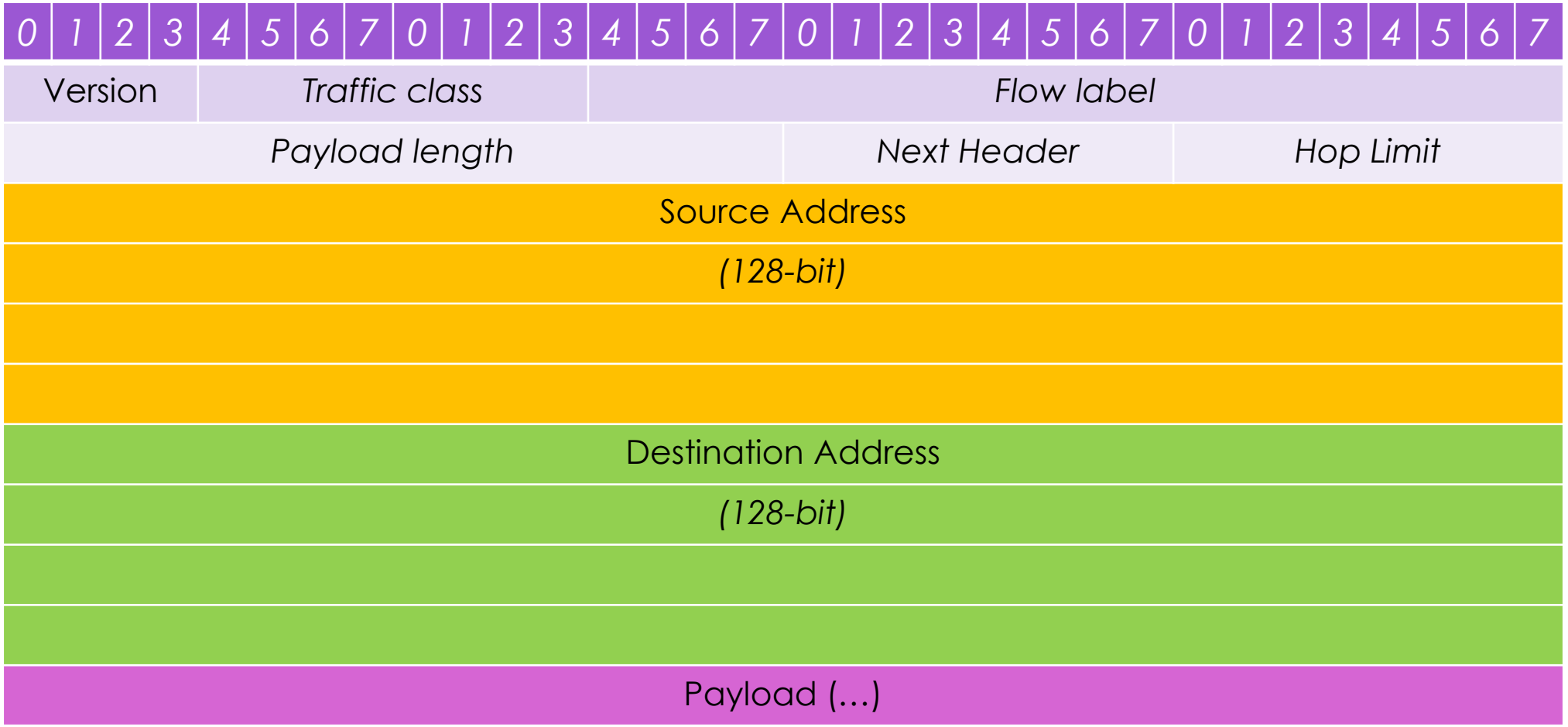
Address blocks are getting much harder to aggregate



IPv6

- New effort from ~1994
  - Address exhaustion was long predicted
  - Note around the rise of WWW
- Standardised around 1998, OS support from 2000
- And till recently, limited effort
  - *1983.1.1 Internet flag day: Comply or disappear. Can't do that now!*
  - Hampered by deployment issues
  - Lacking incentives
    - Nobody does homework till there's a deadline
- What do we get?
  - Bigger addresses
  - And other stuff…

IPv6 packets



IPv6 addressing

- 128-bits = 2^{96} more than IPv4
  - $6x10^{23}$ per square meter on Earth
  - A few thousand for every atom on the surface

- 'Coloned hex-quad (with compression)'
  - Instead of 'dotted quad'
- 8 groups of four hexadecimals (8*16bits)
- For **visuals**, compress
  1. Drop leading zeroes
  2. Drop consecutive zero blocks

1. **3018:0ae8:0000:0000:0000:ae00:0098:8ac2**
2. **3018:ae8:0000:0000:0000:ae00:98:8ac2**
3. **3018:ae8::ae00:98:8ac2**

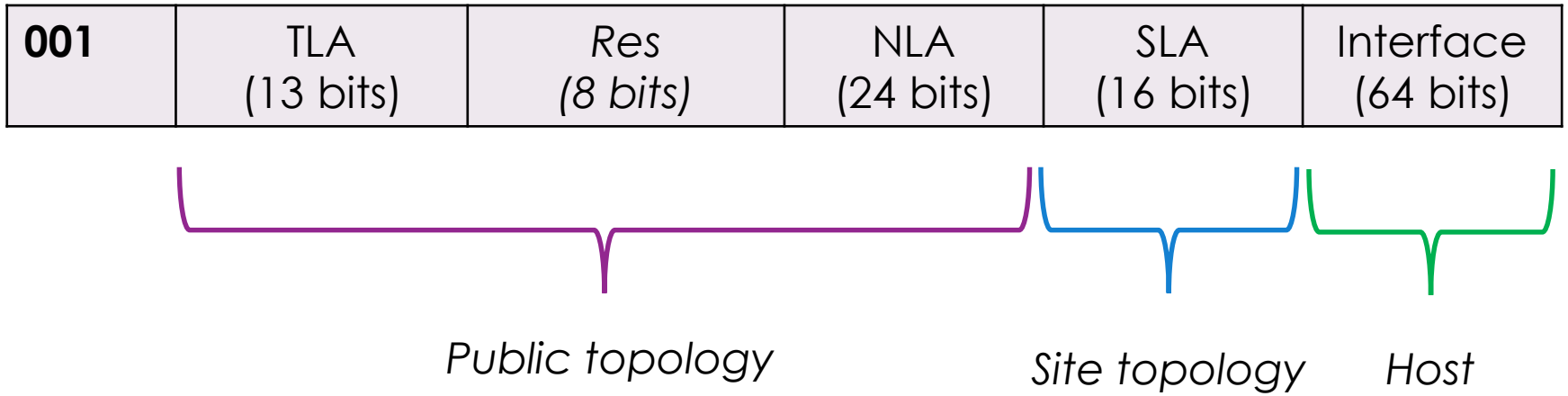
IPv6 address types and scopes

- Types:
  - Unicast – to one
  - Multicast – to a group
  - Anycast – to the nearest in a group
    - Note – no broadcast!

- Scopes: (except multicast)
  - Link-local – my subnet
  - Site-local – my organisation/site
  - Global - everywhere

IPv6 address semantics – 1 example

Unicast-Global

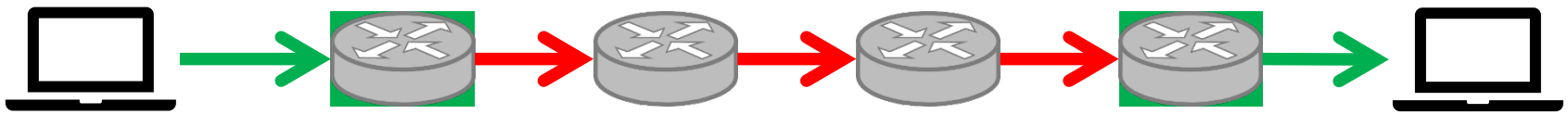


TLA: Top level aggregator – IANA-> global ISP
NLA: Next level aggregator – global ISP->site
SLA: Site level aggregator – site->subnets

Res: reserved

Moving to IPv6

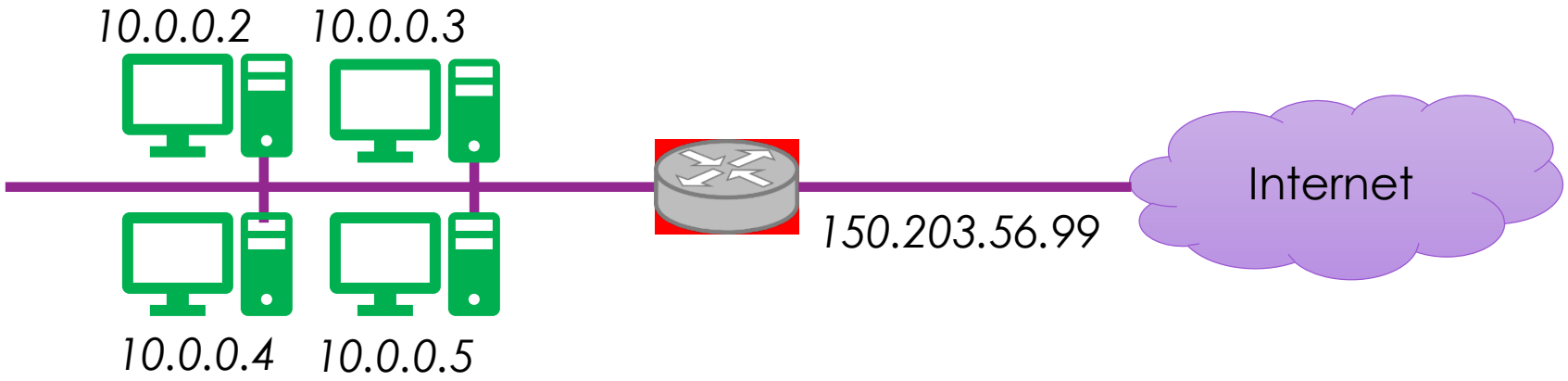
- Will probably have both for another decade or more – around 10-20% now
- Transitioning is a large problem…
  - Bottom-up, top-down challenges – leaves islands of addressing
- Dual stack (run both)
- Translate – convert IPv6 <-> IPv4
  - But how do you handle those addresses
- Tunneling – IPv6 inside IPv4
  - V4 is everywhere



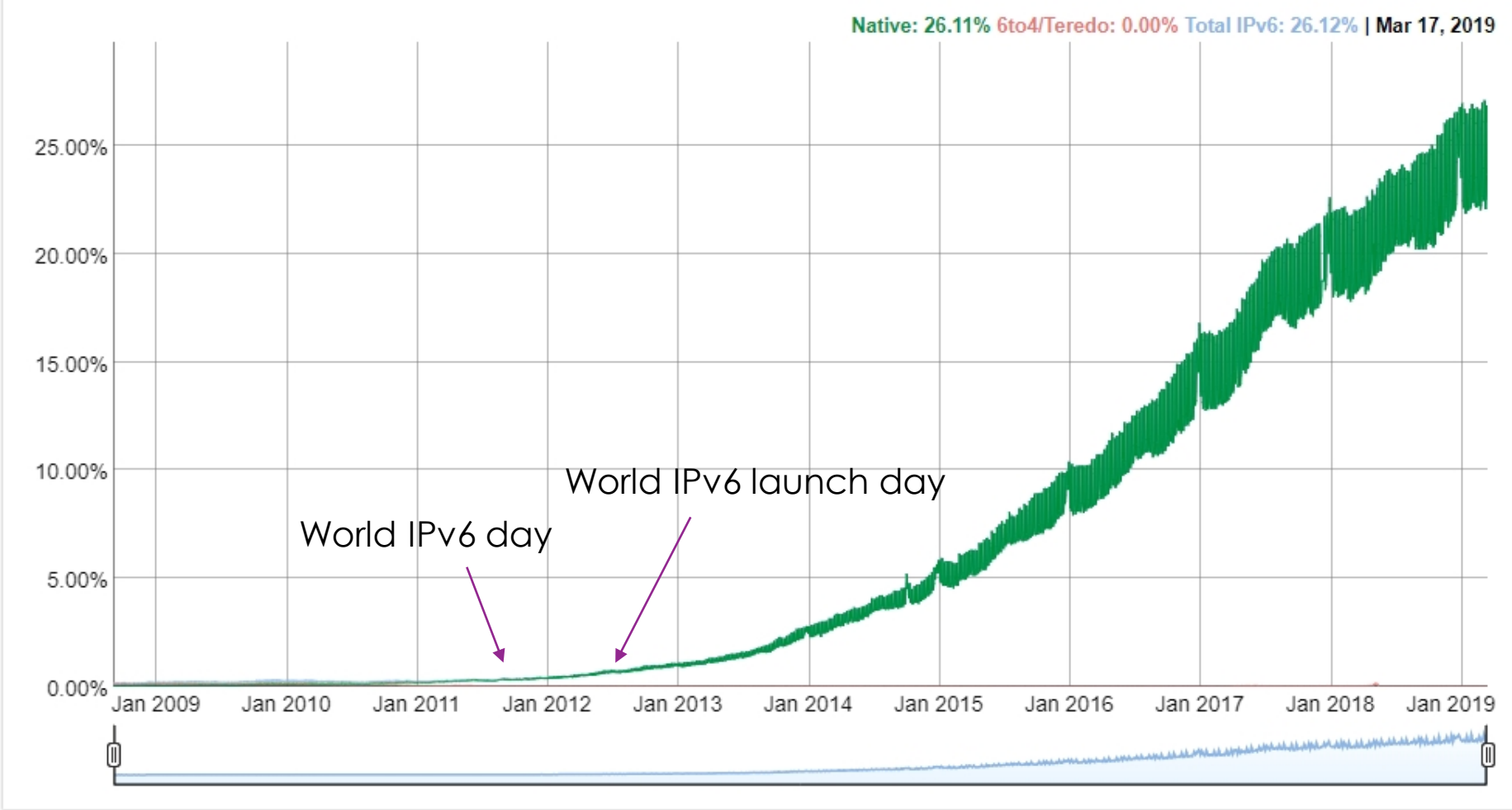
IPv6 killer – NAT – *Network Address Translation*

- Use of private address spaces inside:
  - Homes
  - Mobile networks
  - Organisations

- All 'hiding' behind a single public IP address



Still…


Native: 26.11% 6to4/Teredo: 0.00% Total IPv6: 26.12% | Mar 17, 2019
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
World IPv6 day
World IPv6 launch day
Jan 2009
Jan 2010
Jan 2011
Jan 2012
Jan 2013
Jan 2014
Jan 2015
Jan 2016
Jan 2017
Jan 2018
Jan 2019

Around the world