COMP2221 Networks

David Head

University of Leeds

Lecture 17

Previous lectures

In the last lecture we looked at the Transport layer, which exists in **end systems** or **hosts**, and the two primary protocols it provides:

- TCP (Transmission Control Protocol), which is **reliable**, *i.e.* guarantees no errors, lost packets, or packets arriving out of the order in which they were sent.
- Supports **connection management** and **congestion control**.
- UDP (User Datagram Protocol), which is an unreliable, connectionless service that is usually faster than TCP.

The simplicity of the service provided by UDP is reflected in its much shorter header.

Today's lecture

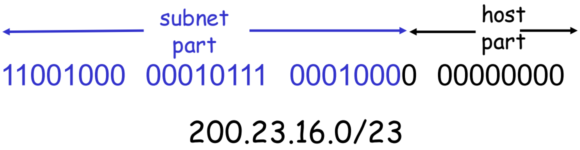
In this second of four lectures looking below the Application layer, we will start to look at the **Network layer.**

- Where IP addresses 'live.'
- Also referred to as the IP or Internet layer.
- Responsible for **routing** packets through possibly many intermediate stages.
- See how packets are **forwarded** at routers.
- **Generalised forwarding**, which has given rise to **software defined networking**.

Classless Inter-Domain Routing

Recall from Lecture 5 that IP addresses now follow **CIDR**: Classless Inter-Domain Routing.

- Subnet portion of address of arbitrary length.
- IPv4 address format: `a.b.c.d/x`, where `x` is the number of bits in the subnet portion of the address.
- Similar for IPv6.



IP addresses: How to get one?

Question: How does a **host** get an IP address?

Could be **hard-coded** by system administrator in a file.

- UNIX: `/etc/rc.config`, `/etc/hostname` (old).
- Windows: TCP/IP properties in control panel / system properities.

More common is to use DHCP = Dynamic Host Configuration Protocol:

- Dynamically gets address from a server.
- 'Plug-and-play.'

Question: How does a **network** get the subnet part of an IP address?

Answer: Gets allocated a portion of its ISP = Internet Service Provider's address space.

ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

ICMP: Internet Control Message Protocol

- Used by hosts and routers to communicate network level information.
- *e.g.* error reporting ('unreachable host'), echo request/reply.
- ICMP messages carried in IP datagrams.
- **ICMP Message:** type, code plus first 8 bytes of IP datagram causing error.

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

traceroute and ICMP

Source sends series of UDP segments to destination.

- First has TTL=1, second has TTL=2, *etc.*

When *n*th datagram arrives at the *n*th router:

- Router discards it, returns ICMP message type 11 code 0.
- When received, sender calculates RTT = Round Trip Time.
- Three times for statistics; '*' denotes timeout.

Stopping criterion:

- UDP segment eventually arrives at destination host.
- Destination returns 'port unreachable' (ICMP message type 3 code 3) because the port does not exist.
- When source gets this, it stops.

IPv6

Initial motivation for IPv6:

- The 32-bit address space of IPv4 was ‘running out.’

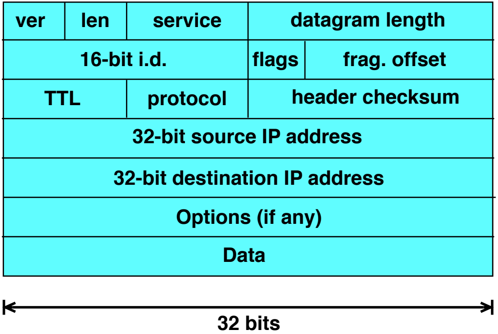
Additional motivation:

- Header format helps speed processing/forwarding at routers.
- Header includes **Quality of Service**.

IPv6 datagram format:

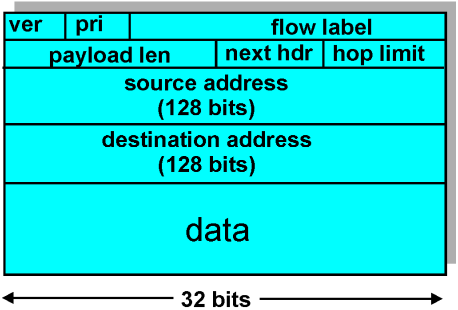
- **Fixed length** 40-byte header, with no optional field as in IPv4.
- No **fragmentation** allowed at routers, *i.e.* breaking one datagram into smaller datagrams. Instead, returns an error message (ICMP type 2).

IPv4 Header



len: Length of header, including options (20 bytes if none).
service: Type of service (TOS), *e.g.* real-time *versus* file transfer.
16-bit i.d., flags, frag. offset: Used for **fragmentation** *(breaking large messages into smaller ones at a router).*

IPv6 Header



Priority: Identify priority among datagrams in flow.
Flow label: Identify datagrams in some 'flow' (concept of flow not well-defined).
Next header: Identify upper layer protocol in the data (*e.g.* TCP, UDP, Options).

Other changes from IPv4

Checksum:

- Removed to reducing processing time at each 'hop.'
- Still checked at the Transport layer (TCP and UDP), and possibly the Link layer.

Options:

- Still allowed, but outside of the header, as indicated by the **Next header** field.

ICMPv6:

- New version of ICMP.
- Additional message types, *i.e.* 'Packet too big.'
- Multicast group management functions.

Transition from IPv4 to IPv6

Not all routers can be upgraded simultaneously.

- *i.e.* there is no 'flag day' when the entire internet would switch to IPv6.

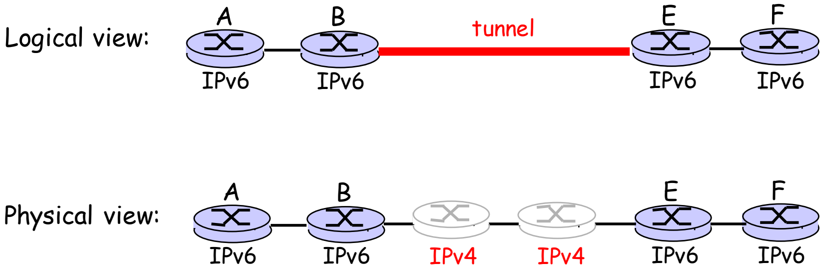
Therefore IPv4 and IPv6 must co-exist, for some time at least.

- How can the network operate with mixed IPv4 and IPv6 routers?

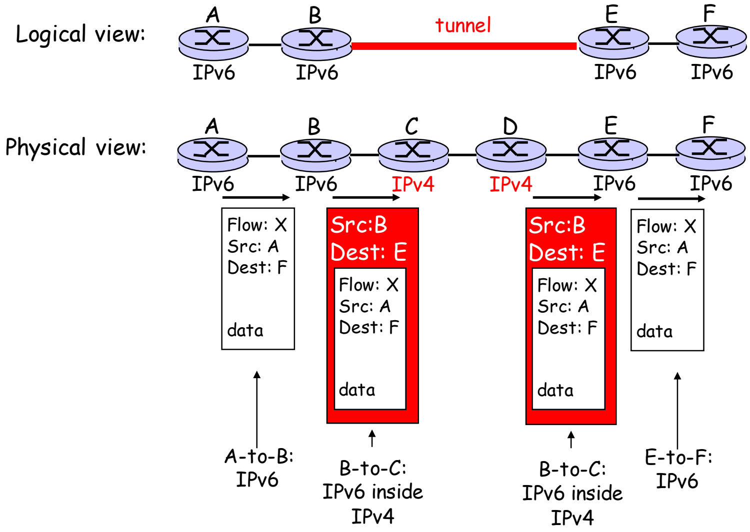
Tunnelling:

- IPv6 carried as **payload** in IPv4 datagram among IPv4 routers.

Tunnelling (1)



Tunnelling (2)



Routers

A **router** performs two functions:

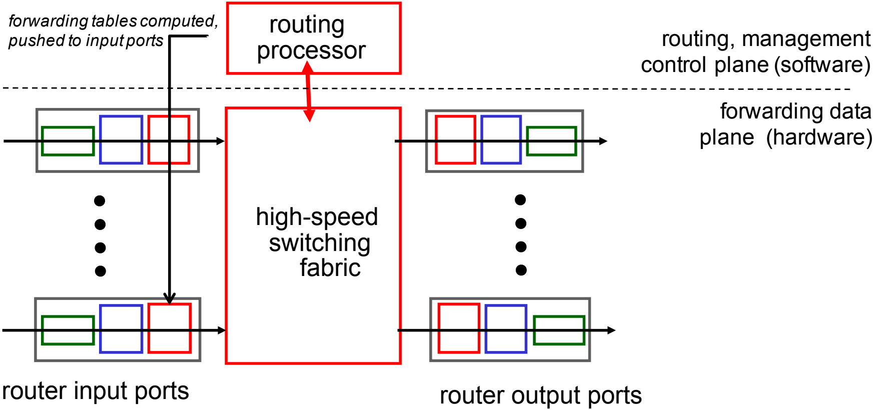
1. Run **routing algorithms** to determine an efficient onward path.
   - We will cover these in the next lecture.

2. **Forward** datagrams from an incoming to an outgoing link.
   - Using a **forwarding table** determined by the routing algorithm.

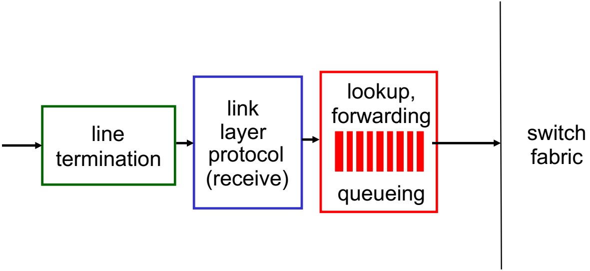
Note that although an idealised router has no Application or Transport layer, in reality these higher layers are sometimes used.

- Technically breaks the layered architecture model.

Router architecture overview



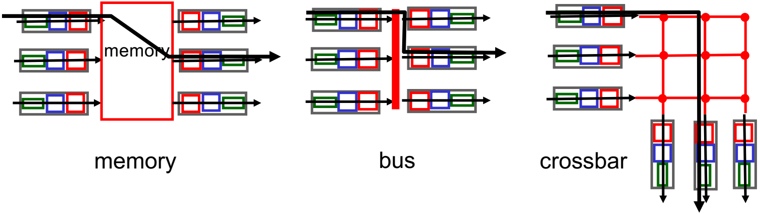
Input port functions



- **Line termination**: Physical Layer: Bit-level reception.
- **Link layer protocol**: *e.g.* Ethernet (*cf.* Lecture 19).
- **Lookup, forwarding**: Inspect packet to determine output port as per a **forwarding table**.
  - If cannot perform processing at line speed, must **queue**.

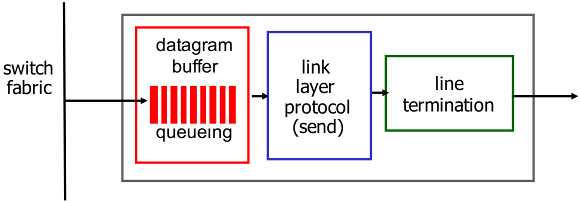
Switching fabric

The **switching fabric** transfers packets from input buffer to appropriate output buffer.



- Early routers used **memory** under CPU control; slow.
- Shared **bus** faster, limited by bus speed (*e.g.* 32 Gbps bus)
- **Crossbar** (*i.e.* interconnection network) overcomes bus limitations; fastest currently in usage.

Output port functions

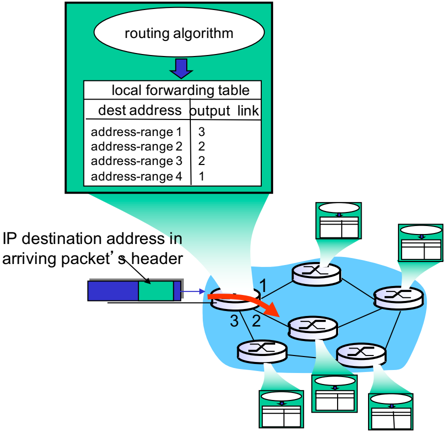


- **Buffering** required since fabric may be faster.
- If buffer is exceeded, packets are lost.
- Can also **schedule** packets, giving higher priority to certain packets based on some criterion (net neutrality?).

Forwarding tables

The **forwarding table** selects the output port for each packet.

- Too many IP addresses to consider each one.
- Therefore maps **ranges** of IP address destinations.
- Uses **prefixes** `a.b.c.d/x`.
- If multiple entries, use the **longest** prefix `x`, *i.e.* the smallest range of addresses.



Generalised forwarding

Early routers only used the destination IP address to determine the onward path for each packet.

Greater control possible by using **generalised forwarding**:

- **Match** incoming packets to **actions**.
- Can use **all header fields** from Transport, Network and Link layer headers.
    - Ports (source and destination), IP addresses, protocols, ...
- Actions can include dropping packets — **firewalling**.

Forwarding based on IP destination address only is now seen as a simple case of **match-action forwarding**.

SDN: Software Defined Networks

The demand for generalised forwarding has lead to the development of devices that support **SDN** = Software Defined Networks.

- **Flow-based forwarding** based on **any** information in the header fields.
- Separates the control from the data.
- Software can exist on separate servers to the router and be broken down in **modules** that can be developed **independently**.
- Network is **programmable**.
- Widespread standard is OpenFlow.

Overview and next lecture

Today we have started looking at the Network layer:

- ICMP, IPv4 and IPv6.
- **Tunnelling** to allow IPv6 messages to pass through IPv4 routers.
- Generalised forwarding and SDN.

For more information see Kurose and Ross, *Computer Networking: A Top-Down approach*, Chapter 4 (7^{th} ed.).

Next time we will see how the forwarding tables are constructed.