Advanced Networking
Autumn 2021: Post-Covid Edition

Ian Batten
i.g.batten@bham.ac.uk
…!mcvax!bham-cs!igb
igb@uk.ac.bham.cs

Health Warning on Slides

- I am re-recording the lectures because I've got better at it, and last year's are all murky shadows and dubious microphones.
- I am joining the 21st century and switching to 16x9 slides, and saving electrons by switching to Helvetica Light, the prince of fonts. Apologies for any jumpy bits where I've laid this over existing material.

What this course is

- Some networking **theory**
- Some networking **practice**
- Some networking **business and reality**

What you will learn

- The main components of networking 2021-style, hopefully with an eye to 2030 (or at least 2022).
- Some background to explain why networking today looks like it does, much of which is rather contingent, and how people make money.
- Some theory and science to justify the technology.
- Last year we merged this course with a network security course, for reasons of resources. We've un-merged them, but I'm going to still talk about some security stuff because Network Security and Cryptography is not available to (m)any of you and this module isn't available to the Cyber Security MSc.

The key question

- Both GCHQ and Google lament that when they ask applicants the question “you click on a link, what happens?”, people can’t really give much of an answer beyond the GUI.
- This course should get you from the click to the wire.

Who are you?

- I’m expecting people from
  - 3rd/4th years of our BSc/MSci/MEng programmes and parallel joint honours programmes
  - MSc Advanced Computer Science
  - Is there anyone else? ICY? Conversion MSc?
  - Is anyone taking/has taken C&NS?
  - Is there anyone who cannot program in one of C, C++, Java or Python?

Extended Course

- This course is a derivative of the historic 20 credit Networks module. It’s a pretty full module, so in practice everyone ends up doing the “Extended” version.

Lecture Formats

- Pre-recorded material, which will replace lectures in which I stand at the front of the room.
- In person sessions, at which I will engage in the famous “story time with Ian” and discuss the nuances, details and edge cases, with hopefully some practical demos if I can work out how to do them in person.
- Zoom sessions, for people who cannot attend the in-person sessions.
- You only need to attend one of the face-to-face or Zoom sessions, and I would really like as many as possible to come to the face-to-face.
- I intend to run **Wednesdays 11–12 Law LT1** in person, **Zoom Fridays 10–11**. I have Friday 3–4 in Biosciences 301 available to do not intend to use it, at least not at the outset.

Some Basic Assumptions

- Spoiler: Apollo 13 did get back to earth
- Spoiler: The Titanic sinks before the end of the film
- Spoiler: Ethernet lower layers and TCP/IP upper layers **have won**. There are no serious contenders on the 10–20 year horizon. Everything else is history or weird curiosity.
- So this course is unashamedly Ethernet and TCP/IP focused, as there are no other games in town right now, nor in the foreseeable future.

Week 1–2:

- This Introduction
- Packet v Circuit Switching, layer models (DoD 4/5, ISO 7), what's in the subnet, transport, application layer.
- Network Hardware: Switches, Routers, data/control/management plane. Software defined networks.
- Background: LAN/WAN split, Arpanet, DoD, OSI. Why OSI Failed.
- Lower Layers: FDDI, Slotted/Token Rings, **Ethernet** (in its various forms).

Week 3

- IP: addressing, routing, concepts. **Why IPv6 is needed.**
- IP: address allocation, bootp, DHCP, SLAAC.

Weeks 4–5

- UDP, RTSP, other transports. QUIC, as it's currently very trendy.
- NAT and its evils. IPv6 as cure. IoT.

Weeks 6–7

- TCP 1: history, basic concepts. Windows, acks.
- TCP 2: Detailed operation
  - This might take more than one lecture
- TCP 3: Options (scaling, PAWS, etc, timestamps)
- TCP 4: Nagle, silly-window.
- This might be a very busy week, with a couple of extra lectures

Week 8–9

- Application: HTTP, SMTP, IMAP, POP3, SNMP, NTP, **DNS**
  - For all the application lectures, I will deal with deployment and server issues as well as the protocol, so you can go out and do stuff with them.
- For the good of everyone's futures, I'm going to spend some time on security issues within these protocols, but that will be in the second half.
- [[DNS added as I only realised I'd left it off when recording the lecture]]

Weeks 10–11

- Routing inside the enterprise: Interior (RIP, OSPF, IPv6 analogues). VLANs, LACP, trunking and bonding
- Routing outside the enterprise: Exterior (BGP, tech and politics). PPPoA/E, VLAN stacking

Office Hours

- TBD: I will put this up in Canvas once we have sorted out timetabling tutorial sessions.

Books

- “Distributed Systems: Concepts and Designs” by Coulouris, Dollimore, Kindberg and Blair is also used for Distributed Systems and covers a lot (was used in the past when the two courses were combined)
- TCP/IP Illustrated, Volume 1 by W. Richard Stevens is the essential book on TCP/IP
- TCP/IP Illustrated, Volume 2 is **not** necessary (it documents kernel implementations) but is a fascinating read (really)
- Systems Performance: Enterprise and the Cloud 2nd Ed, by Brendan Gregg. The networking stuff is brilliant, but mostly outside the scope of this course.
- I will also expect you to read RFCs as we go along.

Things I’ve left out

- There is hopefully some spare time in which we can fit in some of the following.
  - Wireless networking (802.11[abgn])
  - Network management (SNMP in more detail)
  - Network design issues

The basic problem

- To get us going, let's look at the most basic problem of modern networking and see if we can think of some solutions. I'll return to this as something like a running example.

- It is always risky to quote Ken Livingstone, even when he is quoting someone else
- But…
- ***As I was saying before I was so rudely interrupted***

A problem

- An author has written a novel, and wants to send a copy to their US publisher 3000 miles away.
- They want to be certain the publisher receives a complete, unaltered copy.
- But the only communications mechanism they have is airmail letters, as we have been transported back to 1985.





PARK
ROW
BOOKS

This is not wildly unrealistic

- As we will see, all modern computer networking functions by dividing data into small (relative to the volumes we are moving) chunks and sending them as separate messages. An airmail letter is ~1000 words, ~4Kbytes, which is the right order of magnitude.
- There is a finite limit on how fast data can cross a network, set by (as an absolute floor on delay) the speed of light — 186 000 miles per second means that a 6000 mile round trip is a minimum of 32ms.

A basic approach

- Here is one slow, basic approach.
  1. The author sends a page of the novel
  2. The publisher copies it and sends back the copy.
  3. If the author hears nothing for twice the usual delay, they a letter was lost and re-send the most recent page.
  4. If the author receives back a copy which is faulty, they re-send.
  5. If the author receives back a copy which matches, they send the next page, or if they have sent the last page, they stop.
- There are some details to tidy up about page numbers, marking the last page, what “copy” means, etc, but the general idea stands.
- There are significant security holes in this in the face of a capable attacker, but let’s leave that for Network Security next semester.
- For our purposes we can assume the post office to be unreliable, but not hostile.

This is ridiculous as an analogue example

- 384 pages by (at a minimum) four day turnaround = about **five years**.
- In ~~my lifetime~~ the dim and distant past, people played Chess, Diplomacy and other board games by post, taking years.
- There is also a genre called “epistolary novels” told in the form of exchanges of letters. However…

It’s bad electronically too

- Assume:
  - each page is 4KByte,
  - and we can send a page and receive back the reply (let’s say “acknowledgement”) in 32ms (ie, no delays other than the speed of light).
- It therefore takes 12.28s to send 1.5MB, which is 125KB per second, or 1Mbps.
  - (384 messages * 0.032 seconds) = 12.288 seconds
  - (384*4000 B)/12.288s = **125000 bytes per second**

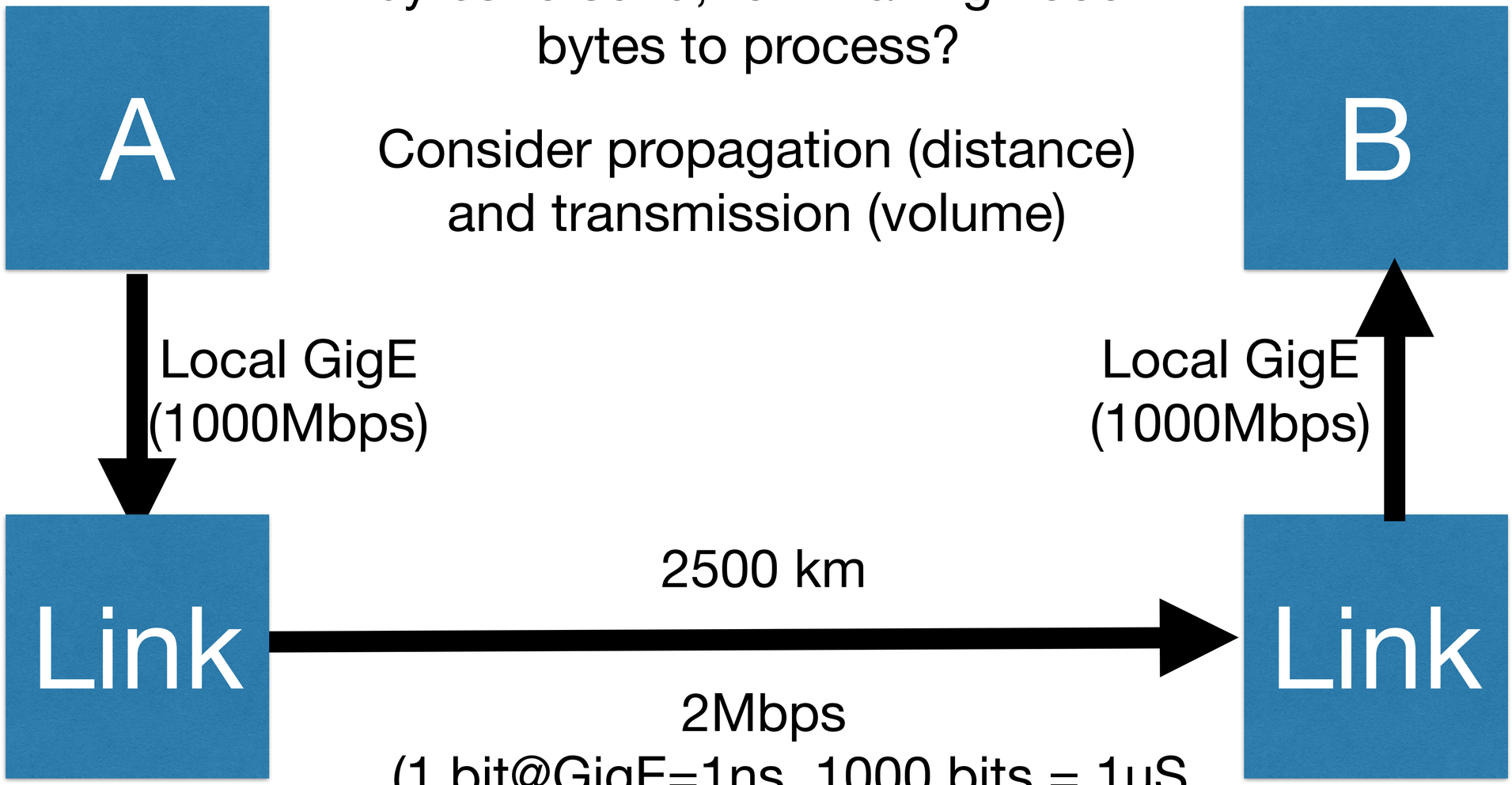
So for Wednesday…

- Stick to the analogue example: sending a novel using airmail envelopes
  - How might you reduce the time taken to send the novel?
  - What fresh problems might your solution cause?
  - What conditions might you relax to get more performance?
- Assume that you **cannot** make the envelopes bigger or otherwise improve the performance of the post office.
  - This is again realistic: the maximum packet size of a network is not a fundamental limit like the speed of light, but the ~1500 byte limit of Ethernet circa 1979 is still very much with us.

- How long does it take a packet of length 1000 bytes to travel over a link of distance 2500km, propagation speed 2.5×10^8 m/s, and transmission rate 2 Mbps?
- Propagation: $(2500 \times 10^3)/(2.5 \times 10^8)=0.01s = 10ms$
- **Clock the packet: $(1000 \times 8) / 2 \times 10^6 = 4ms$**
- **But how many times?**

What is the latency in this set-up?
How long is it from A having 1000 bytes to send, to B having 1000 bytes to process?

Consider propagation (distance) and transmission (volume)



- Takeaway: most systems in 2021 are store and forward, it’s increasingly rare to see packet-based systems which “cut through”.
- That means every hop costs you the time taken to clock the packet plus the propagation delay.
- In modern, complex networks hop counts of 30 or more are not uncommon.
- We want to check checksums at each point.
- The story of failed network projects is the story of underestimating latency (and other things, to come).

- Also bear in mind:
  - Short distance links are dominated by bandwidth
  - Long distance links are dominated by propagation delay
  - “Elephants” (Long Fat Networks) were a problem when “Fat” meant “tens of megabits per second”, are a massive problem when it means “tens of gigabits (terabits?)” per second.
  - We will see later that “filling” a network link is a key issue.

- Suppose Host A wants to send a large file to Host B. The path from Host A to Host B has three links, of rate R1=500kbps, R2=2Mbps, and R3=1Mbps.

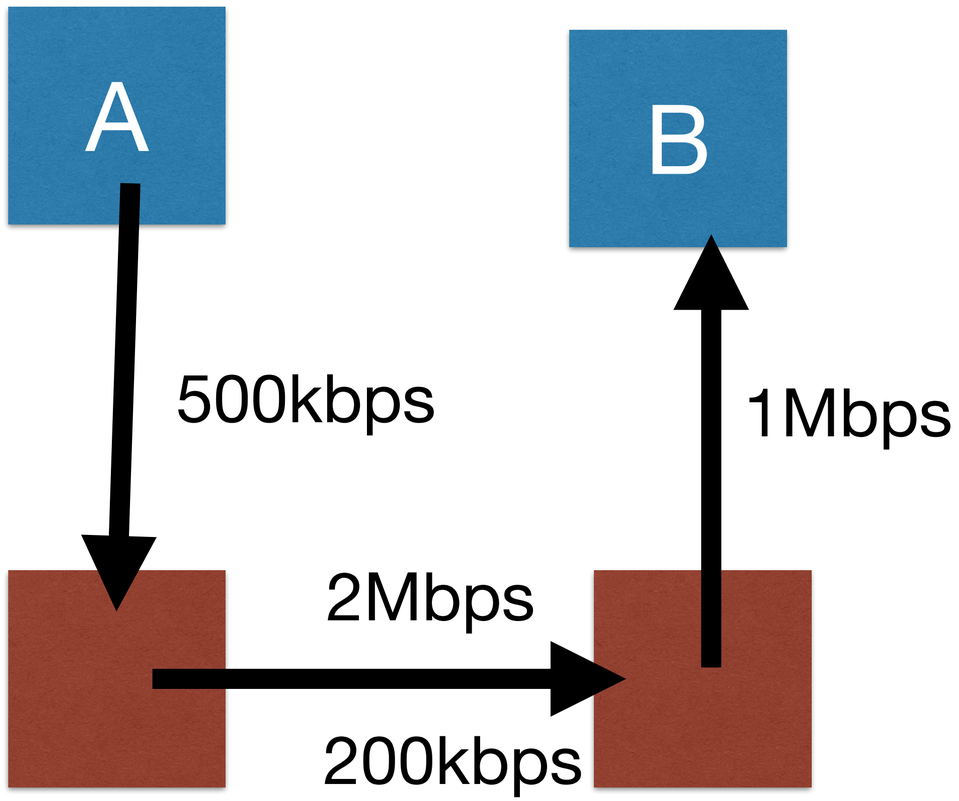
  A. Assuming no other traffic in the network, what is the throughput for the file transfer?

  B. Suppose the file is 4 million bytes. Dividing the file size by the throughput, roughly how long will it take to transfer the file to Host B?

  C. Repeat (a) and (b), but now with R2 reduced to 100kpbs.

Simple Problems become Complex Problems

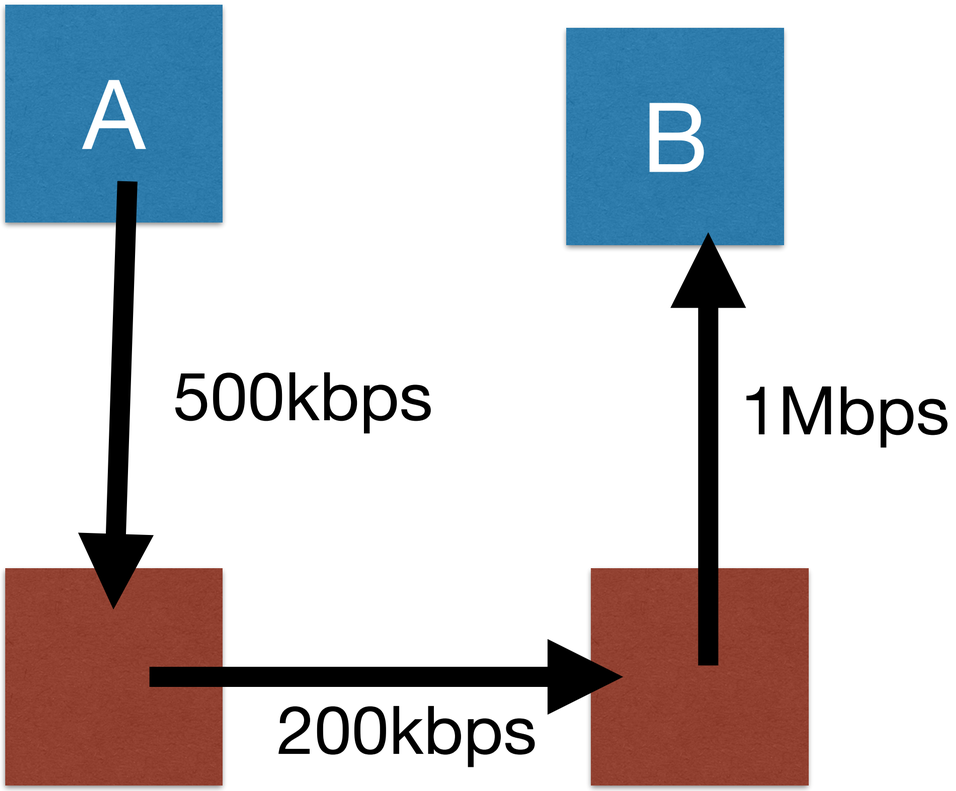
What is the throughput?



- A textbook says the throughput is the slowest link, so 500kbps (64s) or 100kbps (320s).

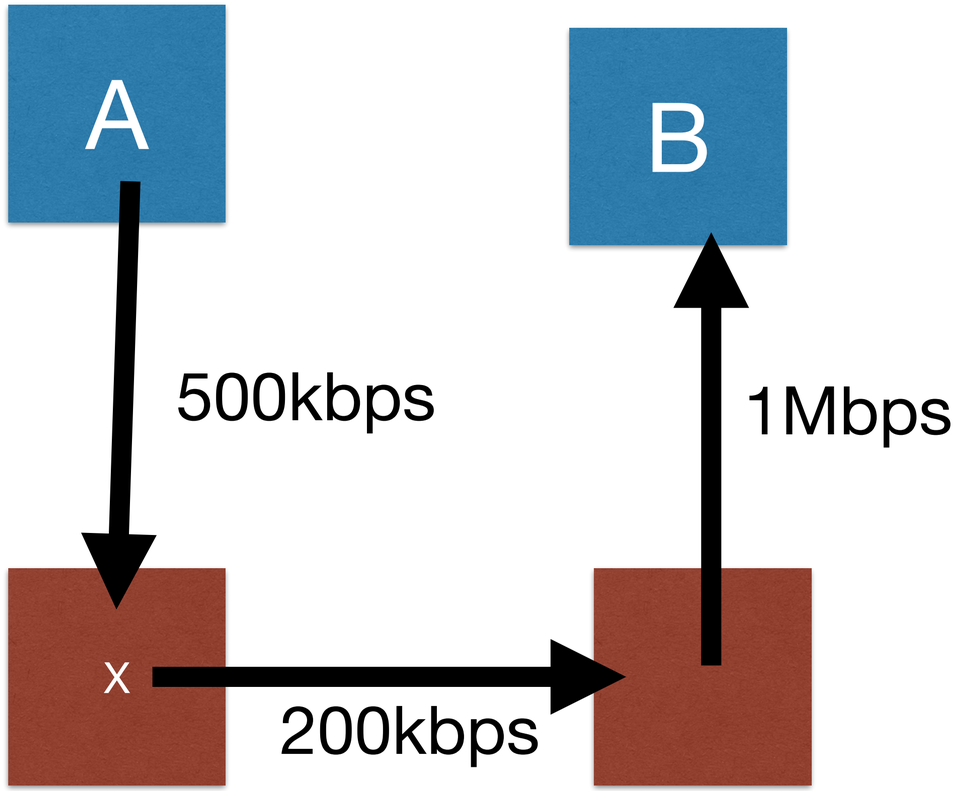
Simple Problems become Complex Problems

What is the throughput?



- But this assumes that "somehow" the sender knows that even though the adjacent link is 500kbps, it should only send 100kbps.
  - How does it know this?
  - And if it doesn't, what happens?

What is the throughput?



- After A has sent 4MB (64s), the router has only sent 1.6MB onwards.
- Where has the other 2.4MB gone?
- The buffers on X. Or not.

Buffering

- We will talk about this later, but core transmission equipment does not have large (we say “deep”, in the manner of storage tanks) buffers.
- Fast, error-checked, double-ported RAM is extremely expensive, and designing core switches with large buffers to handle scenarios like this is difficult.
- So what needs to happen is that “by magic” we need A to throttle its transmission to match the slowest link.
- Something to think about as we go forward.

- Suppose users share a 2 Mbps link. Also suppose each user transmits continuously at 1 Mbps when transmitting, but each user transmits only 20 percent of the time.

  A. When circuit switching is used, how many users can be supported?

  B. For the remainder of this problem, suppose packet switching is used. Why will there be essentially no queuing delay before the link if two or fewer users transmit at the same time? Why will there be a queuing delay if three users transmit at the same time?

  C. Find the probability that a given user is transmitting.

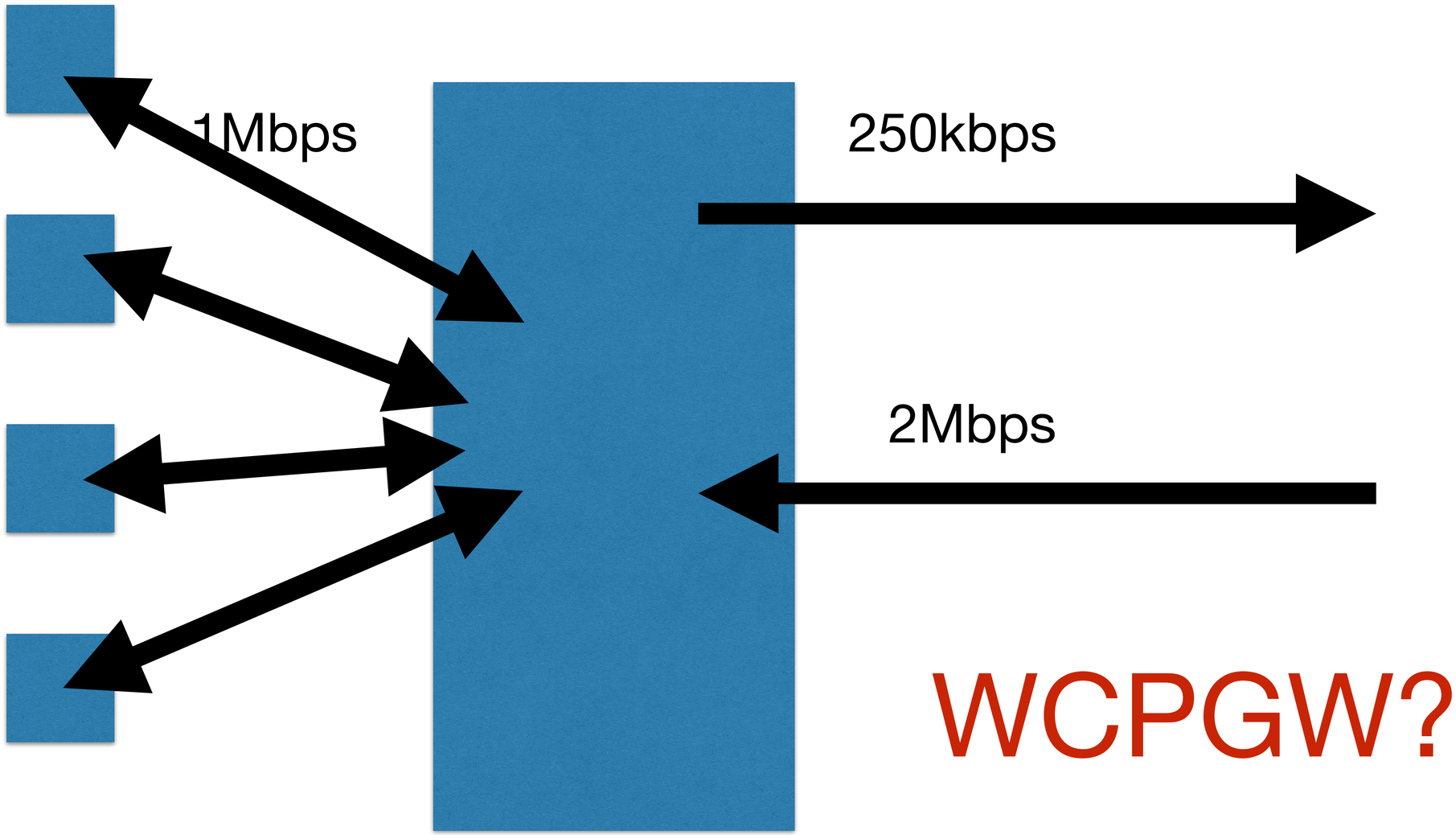
Storytime with Ian



- Most res/SOHO broadband is asymmetric: more downstream (network to user) than upstream (user to network).
- 80Mbps down, 20Mbps is a common setup for VDSL / FTTC.
- 350Mbps down, 50Mbps up is a standard Virgin offer.
- Back in the mid-noughties, 2Mbps down, 256kbps up was rocket science.
- Meanwhile, at a disused airfield in Northamptonshire, just off the A46, trouble is brewing.

ADSL

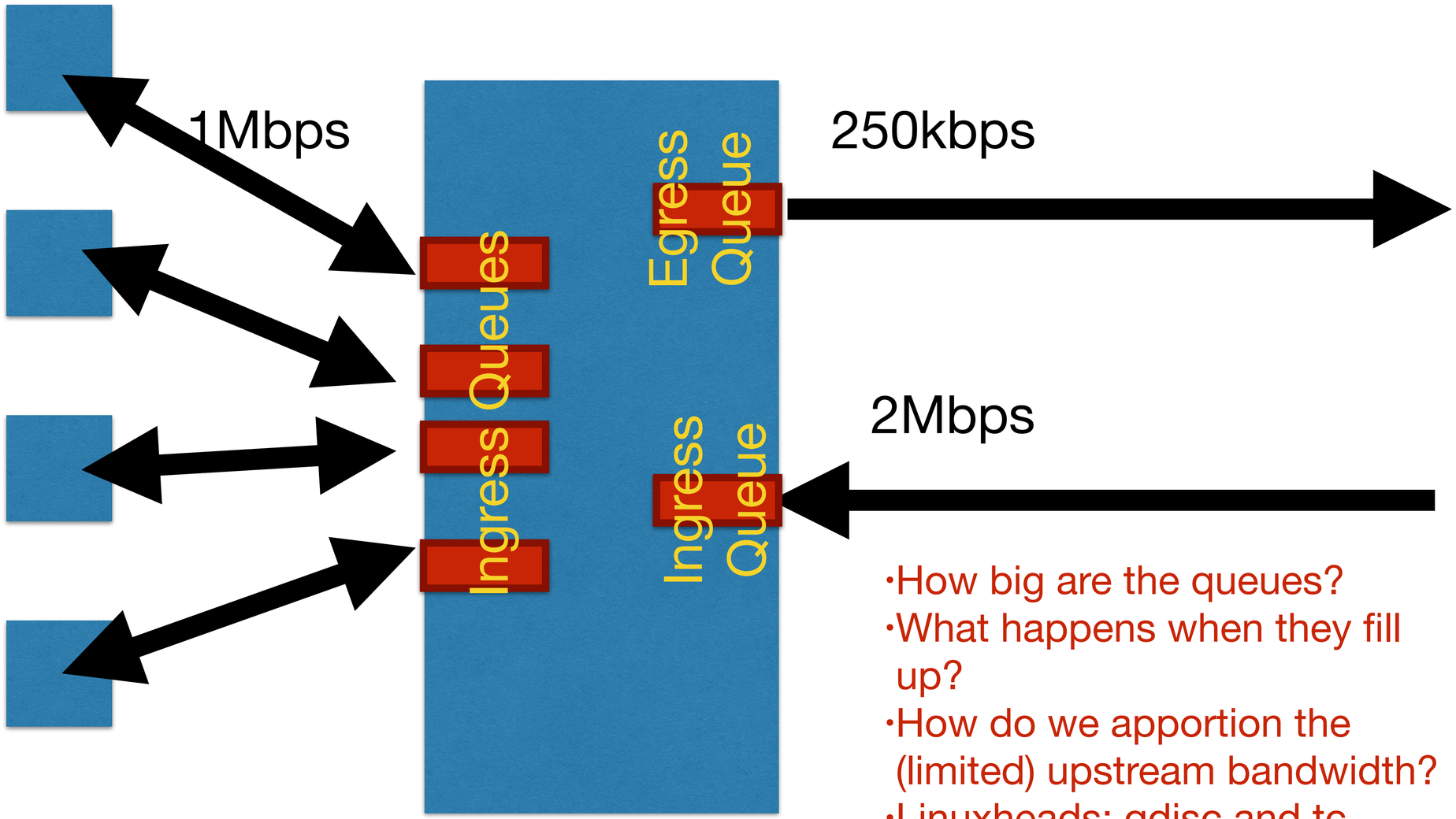
- We will talk about the innards of ADSL later, but the key point is the A for asymmetric.
- It’s 2003 (ask your parents). Just last year, Kodak have realised that these digital camera thingies are catching on, and for the first time didn’t bring a portable E6 processing line for developing slides (ask your grandparents) on site.
- Last year, BT made bank selling one-year contracts for phone lines to photographers to use for three days.
- Now they’re charging £50 for three days’ access to WiFi.



WCPGW?

Nightmare

- System completely collapsed on Thursday, during testing.
- I ended up picking up a call because I happened to be in the office and fancied a trip to the Grand Prix (and ended up sleeping on the floor of the exchange that year and for the following two years)
- Eventually we diagnosed it on Friday and overnight converted Silverstone exchange to SDSL and laid in multiple 2Mbps symmetric circuits.
- What had happened? And what if we hadn't had parts, people, and a willingness to upset the regulator if he found out?



- How big are the queues?
- What happens when they fill up?
- How do we apportion the (limited) upstream bandwidth?
- Linuxheads: qdisc and tc

Queuing

- You cannot fix everything with queues: eventually, people need to see traffic shipped.
- But without effective queuing on ingress, goodput will drop to zero: a random assortment of packets make it through, never enough to send useful amounts of traffic.
  - We will discuss fragmentation later.
- Edge systems: long, deep, well managed queues.
- Core systems: fast, reliable, deterministic, small/no queues.