Patience

Jon Crowcroft
University of Cambridge
jon.crowcroft@cl.cam.ac.uk

ABSTRACT

In this brief note I reflect on my experiences of 60% of the 50 years of the ACM SIGCOMM Experience. This represents very personal views, and I encourage people who were around for any of this time to disagree.

CCS CONCEPTS

• **Social and professional topics** → *Historical people*; *Computing organizations*;

KEYWORDS

SIGCOMM

1 INTRODUCTION

31 years ago (at the time of writing), I attended SIGCOMM 88 at Stanford. We had a couple of papers, but I was paying more attention to the new things I was hearing about – papers on the Internet Architecture (Clark), Spanning Trees (Perlman), Multicast (Deering), congestion control (Jacobson).

2 SIG CHAIR

I acted as ACM SIGCOMM chair from 1995-1999 with Sally Floyd as co-chair. Three things I'd like to reflect on that were, I think, important things we decided back then:

Avoid Pangloss We considered, and rejected the idea of becoming like SIGGRAPH. At the time, SIGGRAPH ran a budget of 5-10M per year, and had a joint equipment show, film show and technical conference. We thought about the equivalent, which would have been to combine SIGCOMM, Interop (a large trade show at the time) and the IETF. This was scary, and when SIGGRAPH had a slight under-attendance one year (in Vegas, one of the few places with conference facilities big enough for their 50,000 attendees) they nearly bankrupted the ACM. I regard it as very lucky we didn't go that way.

Embrace Globalism SIGCOMM had been in mainly in the USA, with brief excursions to Quebec and Mexico City and, in 1990, in Zurich. It was time to engage more fully with folks around the world, especially since the systems under study (e.g. the Internet) were pretty International, so we started to locate the conference regularly in places with a local community, not just within North America. I think this has been proved justified multiple times over.

Have Values We had some notions based on actual value judgements. I make no apologies for this example, but we had an intuition that distributed congestion control really mattered. We discussed with theory and economics folks and really though that there was a real and present danger of the Bad TCP – in some sense, we became a congestion control thought police. There are other examples of things that people asked us to avoid, which we did not.

3 SIG AWARD

With great honour, I received the 2009 ACM SIGCOMM Lifetime Achievement award. (a bit daunting as I've done some other stuff in the decade since then, but that's another story). Here are some reflections on things that I feel are important to do, despite them being awkward and possibly even unpleasant:

Contrarian I believe you should work on things you don't like (QoS, ATM) so you can be an honest, informed critic. You also hone your taste/judgement/skills at making good calls. I currently put blockchain and quantum computing in this space.

Boxing Get out of the box as often as possible - if this means you write lots of papers, but have to cultivate a tolerance for rejection, learn to turn this into refining your explanation of your apparently outlandish ideas, instead of getting stressed. While this may damage your H-Index, it can also be a lot of fun. One year, I had six Hotnets rejects.

Patience Be aware that many ideas take inordinately long timescales to have an impact. One area I did some work on was IPng requirements capture (ended April 1996). Twenty years and more later, IPv6 sees use in about 15% of the Internet - I'm still waiting for multicast to see a bit more use. I am not holding my breath. Sometimes, things learned along the way turn out to be very useful in other contexts; often, in fact.

4 CONCLUSIONS

There are many stories that can be told - here are a few that never made it into a SIGCOMM paper.

Why seventeen seconds? Back in the early 1980s, Europe was connected to the Internet via SATNET, the Atlantic Packet Satellite Network. The links were driven by a version of the ARPANET IMP (interface message processor), called a SIMP (a Satellite IMP). At some point, UCL (where I worked from 1980-2000) measured occasional ping times to typical hosts in the US of upwards of 17 seconds. It turned out that there had been a memory leak in one version of the code, and the solution someone had implemented was to walk through all the area of RAM used for packet buffers, and stick anything that look like it contained an IP packet back on the output link drivers “just in case”. This opposite-of-garbage-collection ran every 17 seconds. The rest is, as they say, packet history. (Actually, I might be mis-remembering this - perhaps it was 23 seconds?).

Another SIMP story was that at some later stage, the source code for the SIMP was lost. The only way to change things (e.g. to switch on or off a different MAC layer, which could be a significant change to performance on a shared geostationary satellite channel with a physical RTT of 0.72 seconds at the speed of light alone), was to patch the binary (e.g. to choose FTDMA versus CPODA). This was (as far as I recall) a relatively rare skill as the CPU in the SIMP was a Honeywell C30 processor – not something that used a terribly widely used or familiar instruction set.

Reboot SATNET For a while in the mid-to-late 1980s we used to be the best place to test new end-to-end algorithms in the Internet, due to the delay (discussed above) and loss (e.g. during thunderstorms in Western Europe or Eastern US), and general heterogeneity of the path. I used to install new kernels in the Sun workstations used as the measurement testbed machines then. It was always quite exciting to reboot a bunch of machines the other side of a network, without any other way of reaching them (there wasn’t another satellite path or a fail-over terrestrial link, or even dial-up to these machines at this point). Luckily (or else because I usually tested stuff locally before doing remote kernel installs) I didn’t have the nightmare scenario of it never coming back to life.

We did have one near miss, where we built a way to update the TCP congestion control algorithm by compiling the new function, and writing it (the binary code fragment) to a control socket to an experimental kernel, which then just did a function indirect call to this new code (sort of Software Defined Networking in 1988). The error was to not check the length of the compiled code when passing it through a socket, and to assume it would fit in an `mbuf` and `bcopy` it there - at some point, overwriting the next buffer pointer, and eventually causing a test machine to crash and trash its filesystem. Oops. Especially since that machine was one of the researcher’s developer machines and had his codebase on (luckily, it was backed up:)

9.6 Kbps! Around this time, we rolled out a 2Mbps terrestrial net around the UK for a research project called Admiral. I wrote an IP router implementation for it. I put in all sorts of debugging and tracing capabilities. One time, someone called me up from the University of London Computer Center to complain that the speed of the net was down to about 9.6Kbps. I asked them to turn on some debugging so I could see maybe what the problem was - they said they had every level on full. I asked them if that meant they had full packet display on the console of the router, which was connected via a 9.6kbps sync blocking serial (RS232) line, and magically, they, and the problem went away.

Multicast goes viral In the early 1990s, there was a workshop about multicast at Stanford. During the talks, there was an

Figure 1: Breakfast over Vancouver

alert that the net was looping multicast traffic. It turned out that a new release from a major router vendor had a very interesting bug which caused the routing system to self-replicate – not as intended. The extra excitement was that a new release had to be deployed/installed and enabled *everywhere* to fix the problem: a single remaining instance would just re-propagate the problem. It was touch and go whether the net would come back again without some rather tricky global coordination. Luckily nothing like that happens these days.

Share prices and elections I remember sitting at the back of another workshop near the end of the 1990s, while many people in the room were looking at online web pages reporting the count in a US Presidential election, whilst other people in the room were looking at their (router vendor employer) share price graph against time. I wondered which if these was really more important than the talks in the room - perhaps both? Verily the Internet was the new rock-and-roll.

Zeppelin versus Seaplane At SIGCOMM in 1998, in Vancouver, we were off to dinner from the conference hotel and the house band were playing Stairway to Heaven (no sign forbidding that). As went to get in a limo, my colleagues recognised the two gentlemen getting out of the same car to be, yes, Page and Plant. As they entered the hotel foyer, the house band jaws dropped, slightly. Even more rock-and-roll The next morning, we went for a ride in a seaplane over the city and nearby lakes – so much faster than a Led Zeppelin: in the photo with the author are Kid Multicast, Dr SIP, Sprint, Professor Fry, and some surfer dude.. Later, we’d feel less cool when we heard Vint Cerf had had the MCI Lear jet scrambled to get him to come to the conference in 1996 to get the SIGCOMM award that year.

Since 2000, however, everything in communications systems research has been going downhill: Pressure on authors to generate reproducible research, to share software and data artefacts, to present incredible game-based animations of their work at the conference, has led to ossification of the work. Now, instead of grand new ideas, everything is just a new version number or a semitone increase (5G, IPv6, F#). More seriously, in fact, this is a good thing, as the community has grown up and take collective responsibility, since the systems we research are often, shortly after, incorporated into the world's critical infrastructure.

5 FUTURE WORK

There are still some thing to finish. One example is that I think the compromise design for IPv6 was a mistake, and we should revisit IEN #1 (available in the RFC archive site), written back in 1977, which clearly addressed (literally) multihoming, topology changes and potential mobility.

The net is now critical infrastructure, but we haven't really thought about how to apply some of the Internet Architectural type thinking to some pieces – for example, we need to have a cooperative/competitive organisation so that cellular data network providers can share infrastructure (cell towers, spectrum etc) as this is technically a win-win in terms of capacity, but would need the moral equivalent of BGP at scales that already change inter-domain policy and algorithms.

The pitchforks and torches of the software revolution in networks needs to be carried to every corner of communications so that we can improve security and safety through design and verification techniques which are being deployed in other parts of the computing industry, and I'd still like to see multicast (widely) deployed.

REFERENCES

See the 50 year bibliometric analysis elsewhere in this issue, in which we try to dispel the notion that there should be a ranking between conferences and journals in this community, using evidence.