The ISP Column
A column on things Internet

November 2025
Geoff Huston

Internet Evolution

This article is based on a presentation I made to the ARIN 56 meeting in October 2025.

Here I'd like to elevate the typical Regional Internet Registry policy conversations above the day-to-day mundanities of address allocation policies with its vocabulary of address block sizes and needs-based justifications, fairness and efficiency and look more broadly at the context of the industry we operate in, and try to gain an understanding of where we are right now, and speculate on where it's all going.

Where we came from

We are caught up in one of the most prodigious marriages in all human history, namely the marriage of computing and communications. Since the inventions of the transistor in December 1947 and the integrated circuit in 1958, the world has been fundamentally changed. Until then all human endeavours had been limited by their geography. The industrial age and the explosive introduction of the railway in the mid-19th century heralded a significant societal change as nation states shifted their foundations of wealth and power from land and agricultural activities to industrial production. Facilitating these changes was firstly the telegraph and then the telephone, allowing these new industrial companies to project their power and influence across larger expanses, harnessing greater volumes of production and increasing their wealth. When we added computers into the communications realm the rate of change increased dramatically. It took us a decade from the invention of the transistor to that of the integrated circuit, and a further decade to reach the point where computers passed through the transition from esoteric research endeavours to essential tool for data processing and communications.

In the late 1960's, Bell Labs programmers, Ken Thompson and Dennis Ritchie, had devised the Unix operating system. It was written in the C language, a so-called high-level language and its associated compiler could produce assembler code for a variety of computers. It was one of the first of the "open" operating systems of the age. This was not exactly an unforced choice: Under a 1956 consent decree in settlement of an antitrust case, the Bell Systems (the parent organization of Bell Labs) was forbidden from entering any business other than common carrier communications services and was required to license any patents it had upon request. Unix could not, therefore, be turned into a product. Bell Labs shipped the source code Unix system for the cost of media and shipping to those who asked, allowing universities and other organizations to modify and extend it. This open model, both of the code itself and the refinement and development of the system, facilitated the development of many variants in the ensuring years, including the highly influential Berkeley Software Distribution (BSD).

In 1973 Bob Metcalf, working at the Xerox Palo Alto Research Centre published a memo describing "X-Wire" a 3 million bit-per-second (Mbps) common bus local area network, which became known as "Ethernet". It was notable because Ethernet was the absolute minimal form of

computer networking. It really was just a wire. It had no internal switch, no packet framing, no clock, no controller, no network state. Nothing! Just a wire. And in many ways Ethernet changed the way we thought about computer networks and communications architectures for computers. There was no technology inside the wire. It was the simplest network you could ever have. It was a wire. Everything happened in the connected computers at the network's edge. And that's why Ethernet took off because every conventional function of network control was pushed off the network into the connected computers and implemented as a collection of distributed algorithms. The network itself, this wire, was just transmission and only transmission. "Dumb network, Smart devices," as we constantly reminded ourselves. What this also meant is that the money to operate the network wasn't the network's problem. That's up to the computers that connected to this network, The connected computers were operating the software and so the network was just a packet transmission medium.

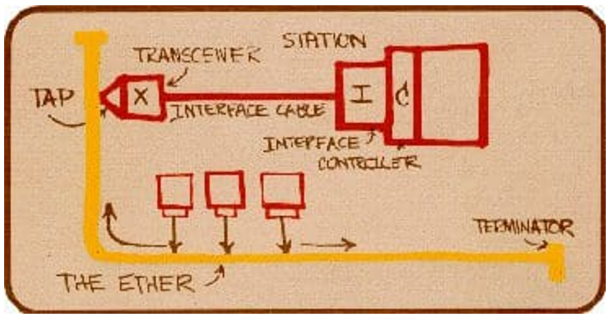

Figure 1 – X-Wire framework

So, the collective "intelligence" for an Ethernet local Area Network (LAN) was placed into the computers that connected to this common wire. All Ethernet packets were self-clocked, using a 64-bit preamble to allow all receivers to synchronise their receiving data clocks against the data rate of the packet being transmitted. Packets were between 64 and 1,518 bytes in total length. There was no centralized contention control – each station used a CSMA/CD protocol to negotiate exclusive sending access to the wire, and all connected computers received all packets that were passed onto the wire. It used a unique station addressing plan in the form of a 48-bit MAC address that is still with us today! But most importantly, like Unix, it was an open standard specification.

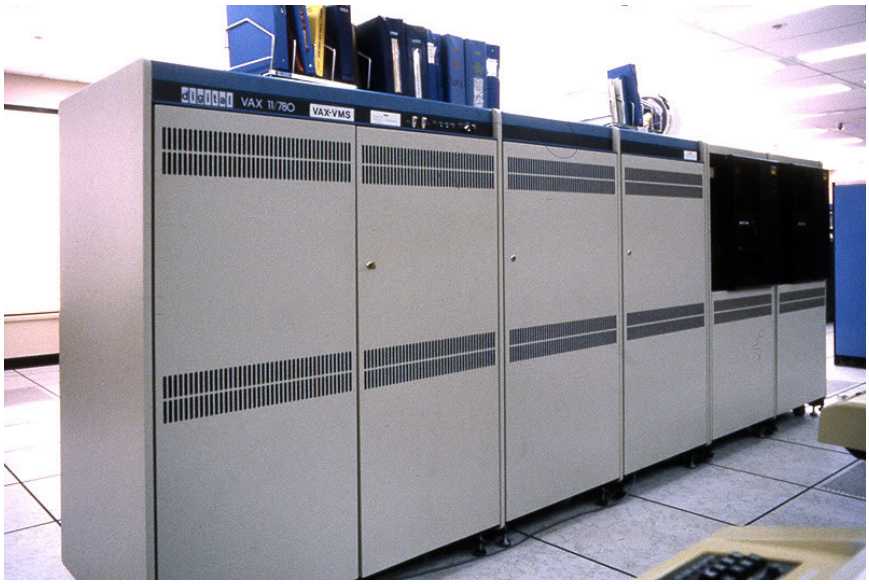

Figure 2 – DEC Vax 11/780

The next image (Figure 2) takes us back to 1977, with the release of the Vax 11/780 computer, made by the Digital Equipment Corporation. This was a medium-size mainframe computer capable of executing 1 million instructions per second (MIPs). This was a very popular computer used in many corporate environments as well as universities and research institutions.

If you look at other 50-year old technology artefacts, many of these items don't look all that different from what we use today. Yes, cars today are lighter, more efficient and come with a whole lot less chrome, but they still have a wheel at each corner and some internal engine for propulsion. And I guess that there still 50-year old cars around today, and doubtless some of them are still on the road. But this is not the case for computers. There are very few 50-year old computers left in today's world, and those that still exist are in museums. Why has the computer industry been so ready to cast off its past lives and embrace change so readily?

The essential difference here it's that the pace of *Moore's Law* in the improvements in the production of integrated circuits on silicon wafers has been truly miraculous. Moore's law, originally an observation by Intel co-founder Gordon Moore in 1965, is the observation that the number of gates on an integrated circuit doubles approximately every two years, while the cost of fabrication of the chip has increased by a far lower factor, if at all. The cost and quantity of compute capability have improved inexorably over the same 50 years (Figure 3).

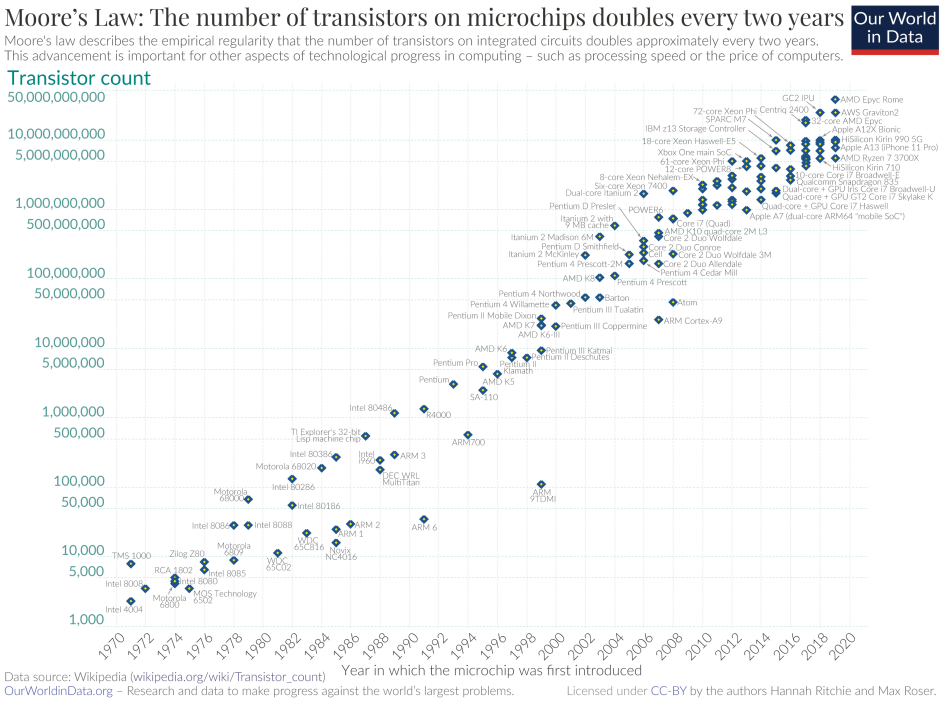

Figure 3 – Moore's Law – from https://commons.wikimedia.org/w/index.php?curid=98219918

Exponential growth is often hard to come to terms with. It has its own momentum and raft of unintended consequences. Initially, this was all about making these lumbering mainframe computers faster, smaller, and cheaper. The mainframe computers of the early 1960s that took up an entire data room with multiple racks of circuitry, power and air conditioning changed every few years. These machines had assured obsolescence from the moment that they were made, and their useful service life was best measured in units of months rather than decades!

This dynamic pace of evolution of computing capability has impacted many aspects of the computing environment. Network protocol design was certainly part of the impact zone of these improvements in processing capability. One of the objectives of a computer network was to allow any connected computer to communicate with any other. For that each computer needed to have a unique identification field, or an "address" to distinguish it from all the other connected computers. The proprietary network protocol used by the VAX Systems was DECnet, and at that time Phase 3 of this network architecture was in widespread use. It used a 16-bit address field, allowing a maximum of 65,535 computers in a single DECnet environment. In the late 1970s the concept of a single network populated with tens of thousands of these large lumbering computers was an unaffordable fantasy. But that thinking did not factor in exponential growth through Moore's Law, and the associated exponential drop in size and cost. The computing environment rapidly shifted in the 1980s into personal computers, and the multi-million dollar mainframes of the 1970s with their specialised environments of highly curated clean rooms transformed into small boxes that found a home of every desk, both in the workplace and in the home. What was an esoteric fantasy of running out of 16-bit DECnet addresses in the 1970's was an obvious reality within just ten years.

But not every computer networking protocol had such a severely constrained design. The folk working on the Internet Protocol (IP) took a far more radical step and used a 32-bit device address architecture. Instead of an inbuilt ceiling of some 65,000 connected devices the IP protocol had a ceiling of some 4.3 billion connected devices! This was crazy thinking, even during the period when computers escaped from their clean rooms and invaded offices and homes. This was a larger number than the world's population at the time. This was a computer protocol that matched the inexorable progress of the expansion of the computing resolution.

We now had a triumvirate in the combination of the advances in the speed, size and cost of computing, an openly available highly efficient operating system for these computers in the form of Unix and a freely available computer networking protocol in IP that had this massive inbuilt capacity through its crazy decision to use a 32-bit addressing architecture!

This was why the IP design was so revolutionary. A few years after that photo in Figure 2 was taken, there were a few hundred thousand of these Vax computers in the field. And we looked at the IP address space, with some 4 billion addresses, and said to ourselves: "What's your problem? We're here forever!" Awesomely wrong, as it turned out!

Moore's Law continued to play havoc with this industry. At the same time as the VAX 11/780 was released, over on the West Coast of the U.S. the Apple Corporation released its first Apple computer. This device was never going to challenge the mainstream computer industry. It cost hundreds of dollars, not millions, it lived on the desktop, and it played games. But Steve Jobs, Apple's founder, appreciated the true effects of Moore's law. In the search for untapped markets that had could absorb the production capacity of the silicon chip industry Apple was a pioneer in the consumer market and applied design principles to personal computers. This was a computer that not only was at home, but it looked like it was truly part of that home. But Steve understood more than just the effects of Moore's law in terms of cheaper and smaller. He understood Moore's Law in terms of using that computing capacity to elevate his machines to interact with its human user on human terms. Apple developed the Mac, a computer with an intuitive look and feel, to the extent that it did not need a geeky user manual. (Coincidentally, this graphical interface was also developed at the Xerox Palo Alto Research Centre, and while Xerox was highly influential in these landmark inventions, the company stubbornly stuck to being a photocopier company!)

When you powered up an Apple Mac, it displayed a friendly "hello!" The human on the other side of the keyboard didn't need to memorise some arcane Job Control Language to make this machine sing and dance. It used a part of its compute capability define an interaction with its users that was entirely phrased in human terms. And in the consumer market this worked! The personal computer market quickly outpaced the mainframe market and dominated the entire computing environment.

Something else was also happening here. In these early days of computer networks, we viewed the network in the same terms as we viewed the telephone network. Human users of telephone both spoke and listened. The telephone network was symmetric and invisible. In computer networks the connected devices both provided digital services and accessed the services of others. Connected computers were both client and server in the networking environment. But that's not really thinking about Moore's law in a useful way. Sure, there was a driving need for more and more personal computers, but these were not both client and server computers. They were positioned as simple clients. There was an emerging segmentation of connected computers into dedicated client and server roles. The large mainframe computers didn't disappear. They were used to provide services to the millions, then tens of millions of users behind these client personal computers.

It was no longer a computer network built in the image of a telephone network, but a network that was modelled on broadcast television. With slightly better programming control that allowed every client to perform their own programming. Users at home were not providers, and they did not actually want to host services and certainly did want these service delivery platforms to be located in their homes. They just wanted the computing equivalent of the television set. Consequently, we constructed the emerging Internet network of the late 1990's around that model of clients and servers. At the time this was convenient because we were still building the internet on the underlay of the telephone network. We applied this entire asymmetric behaviour of clients and servers back into the network architecture of the internet itself. If you were a client, you couldn't do much at all, but you could make contact with servers. The connected edge client computer was the thing that simply said, "Lets' look at the data out there." The dial-up world of the 1990's and the DSL/Cable modem world of the 2000's in the last mile access network were a good fit to the demands of client/server networking at the time and the Internet rapidly expanded by repurposing the existing last mile infrastructure, avoiding the need for hefty up-font capital investment by Internet Service Providers.

There were millions of clients who were eager to own their personal computers and use the Internet to access digital services. The industry continued to grow. By around the year 2000 we saw the use of specialised data centres that were intended to coalesce all these servers into a common curated environment with power, cooling and a couple of human attendants to scurry about and replace dead or dying server units. This server world also saw role specialisation. There were web services, mail services, data services, archival services and so on.

Compared to the current AI-scale data centre designs, this was a small-scale activity. These Internet service data centres were only a room or two with a few hundred megawatts of power. All of those online services were now crammed into these dedicated service delivery points. As we continued to expand into the consumer space we turned to large scale "network exchanges" and "network peering points" to service these concentration points where data intersected with service delivery capability. Clients never exchanged data with other clients. Even transaction was a transaction between a client and one or more servers.

And then the network moved into the whole thing about network traffic engineering. This was interesting in that the silicon industry was working like crazy hard to expanding the network's

client base, but at the same time we were spending a lot of time and measure to control and even ration the use of the network to ensure that the limited resource, network capacity, was used fairly and efficiently.

This network was now a long way away from being "just a wire". It was replete with hardware and software to perform all kinds of functions relating to resource control, variable service response, defensive measures and of course data collection. The progress of Moore's Law enabled further functionality to be added, as the processing capability of these service delivery platforms increased. But this progress of silicon chip capability extended to more than the capabilities of the data centre services and in-network middleware. When you shine a laser light through a fibre optic cable and modulate the light to carry a digital signal with on/off keying then you can work with a simple light detector. If you want additional data capacity from a fibre circuit then the logical approach is to use the same signal modulation techniques we previously used for analogue modems, namely using phase and amplitude modulationof the light signal. While this can radically improve the carrying capacity of a fibre system, the practical limits of this approach are strongly dependant on the signal processing capabilities used at either end. The reason why you we can push up to almost a terabit per second of signal through fiber optic cable is not solely because we are doing anything radically different with glass fibre, but the increased capability of the digital signal processors (DSPs) we're using at either end. If you're using three nanometre chip technology, it appears that you can achieve some 800 Gbps from a single polarisation of a light signal through fibre. It's not necessarily easy, but its achievable. At the same time, we've lifted the common network bearer capacity to 100 - 200G with these latest generation of DSPs

A Network of Abundance

These developments have heralded a huge shift in the role of the network. When transmission capacity was a scarce resource, then rationing access to resource across competing demands was the role of the network. Rationing scarcity is a high value task, as in economic terms the controller of the resource can demand a "scarcity premium" from competing users. But if capacity is abundant to the extent that it overwhelms all demands, then the network operator's role shifts into a basic undistinguished commodity broking role. We've seen this shift in the commercial world, where the share value of network carriage operators has plummeted in line with this shift from scarcity to abundance. The response from the network operations sector has been an effort to add further functionality to the network, in the form of variable service responses, network segmentation and customisable forwarding functions, but to date this has increased costs in the network but has failed to attract sufficient interest from the network's client base. The same basic improvement in technology capacity also mean that connected devices are increasingly capable of managing their own requirements for service responses from an amply dimensioned abundant network. In this environment of network abundance, the network service itself quickly becomes an undistinguished commodity good, and is priced accordingly.

The silicon industry then turned its attention to handheld and embedded devices, and once more, it overachieved! Billions of devices were added into the Internet in the short span of a couple of years as a result. The cumulative result is that we are now in a digital world that is defined largely by abundance and scale. Abundant processing, abundant storage, and abundant carriage capacity,

What happens in an environment of abundance? This is a very relevant question, as abundance is the driver of today's internet. Up until today scarcity drove this industry. And now we're in a different industry where almost everything is being commoditized and being overwhelmed with capacity. And where we used to use pricing to ration a limited resource when performing the distribution function, we are looking for a new model. Pricing isn't an answer to this new challenge So if you're a carrier, life is now grim. The carrier's historical role is to take a single resource and

share it amongst the folk who want to use it. But that's over and the underlying value of the network infrastructure is extremely low. Didn't T-Mobile sell the Sprint wireline network to Cogent for a single dollar? They probably paid too much.

Abundance appears to have destroyed the network completely! The role of the network was a "just-in-time" service that connected users to service on demand. This has been largely replaced by Content Delivery Networks (CDNs), whose delivery model is to position replicas of the service content across a large set of service points. This could be characterised as a "just-in-case" model of service delivery, where content is pushed out to the edges, close to potential consumers, just in case they ask for it.

This has had its own ripple effect. As we move content and service closer to users the network becomes smaller. We are eliminating distance as the dominating characteristic of the network. We don't grab content from far away any more. The packet-miles, or the distance a packet travels from the server to the client, are now tiny. When you shrink the network from the circumference of the globe or across a continent down to the dimensions of traversing a single city, then everything is easier, faster and cheaper. If you can build an internet that looks and performs like a Local Area Network, then where's the scarcity? Where's the performance issues? They're largely gone. When you shrink distance in networking, everything else becomes easier.

Bigger, Faster and Cheaper

Today's distance-shrunken Internet is, oddly enough, bigger. We are talking terabits of capacity in the core systems and gigabits out to the edges, both in last mile fibre and last mile radio. We just publish all the content in all the places, all at once, because that's what abundance says. All of this has allowed us to move to a network which is phenomenally big because it's not "distance-big" it's "capacity-big" because when you shrink distance you can improve protocol performance and decrease cost.

This network is now fast. Really fast. Not only do I get a gigabit per second to my house, but I can actually pull down content with a cooperating server or two at that same line speed. What enables this increased performance is shorter network distance. Network transport protocols are feedback-controlled protocols, and when you bring the two ends closer together you increase the number of potential feedback cycles per unit of time, and this increased information flow provides better guidance to the transport protocol to optimise efficiency of delivery.

And, yes, it's cheaper. If I took a gigabit per second internet service and pushed it back to the currency unit of 1975, fifty years ago, then the $100 per month that users pay on average for such a residential service would cost $17 of these 1975 dollars. And not only is this cheap by historic standards, but its reach has also extended all the way to the most remote corners of the surface of the earth, with the advent of low earth orbit satellite-based services.

We seem to have achieved the elusive triple of better, faster and cheaper. The engine that has got us here is this continuous refinement in the fabrication of silicon chips.

The End of Moore's Law?

So far, we've looked at the upbeat story of the past fifty years. But while Moore's Law has carried the burden of the progressive improvement in this industry for the past sixty years, are we reaching the end of this journey? Down on the silicon chip we're down at feature sizes of 3 nanometers, and it looks like 2 nanometers is achievable. How much further can we take this continuous improvement in silicon chip manufacture? Or, to put it more bluntly, when will we reach the end

of Moore's Law? To put that into context, a silicon atom has a diameter of 0.23 nm. This implication is that a chip feature of 2nm in size is only 10 silicon atoms wide

Can we go to one nanometer feature sizes on silicon chips? Well, you can try. We're using extreme ultraviolet rotation already to produce a wavelength of 13.5nm for chip lithography and the process is extremely complex. Even if you can achieve a reliable and scalable lithography process, there is also the problem of electrons, and let's assume they're particles even though there is the duality of quantum mechanics. In small enough spaces electrons have a mind of their own. It's hard to get this chip feature size even smaller using today's planar integrated circuit technology. That thing we did in 1957 to create an integrated circuit has been progressively refined year after year, and we've made the features smaller and smaller as a result, but it's possible that this process of continuous improvement will grind to a halt. Nobody is announcing a 4 trillion gate silicon chip this year, and I'm not sure they ever will. This technology as it stands is close to that physical edge of impossibility and it appears that the next innovation needs to be really off-the-wall clever and do something that none of us have thought possible.

That's not the only limitation we've encountered in chip capability. The clock speed of processors has topped out in around 2000. Processors may have more gates, and CPUs may have more cores, but the processor's clock speed has remained pretty constant since then (Figure 4).

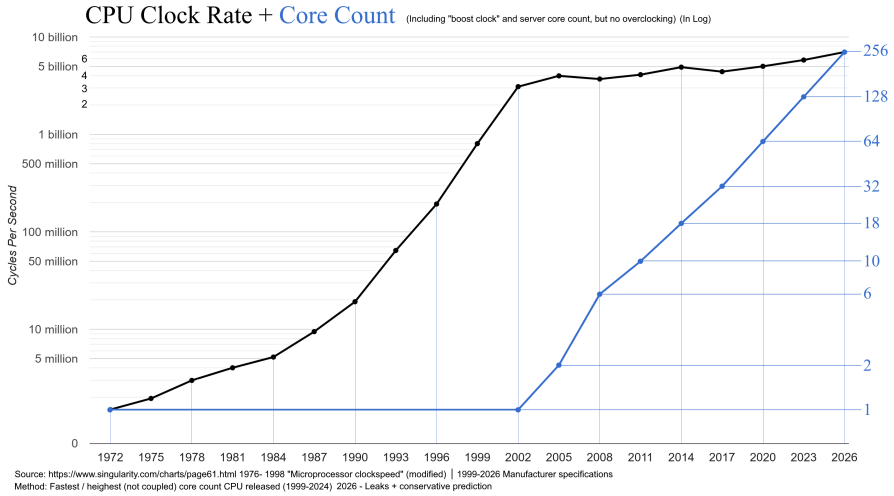

Figure 4 –CPU Clock Rate– from https://en.wikipedia.org/wiki/Clock_rate

It's way too early to proclaim that Moore's Law is over. But perhaps it wise to contemplate what it means for us if Moore's Law becomes the gift that has effectively stopped giving

IPv6 and IP addresses

How does all this relate to IPv6 deployment?

The effort of the early 1990's intended to provide a path through the situation where the population of connecte3d devices exceeds the capacity of the 32-bit address plant. The response, a redefinition of the IP protocol using a 128-bit address field, preserved all the behaviours of the Internet, but the lack of any form of backward compatibility implied that all the network's users and operators were exposed to the marginal cost of adopting a new network protocol, but in fact gained none of the marginal benefits that a completely different network protocol might offer. It was still just IP.

The alternative option, which was also available was to deploy Network Address Translators (NATs) in the network's infrastructure. To make up this shortfall in IPv4 addressing, NATs were adding greater processing capability into the network's infrastructure. NATs are a client-side response and are viable in the long term only if we can scale processing efficiency in line with demand growth. The purpose of NATs was bridging that gap between the capabilities of the technology we had, and how size of the network that we wanted to deploy. As the network grew, the size of the gap increased, and the cost in terms of infrastructure complexity just got higher and higher. (Figure 5)

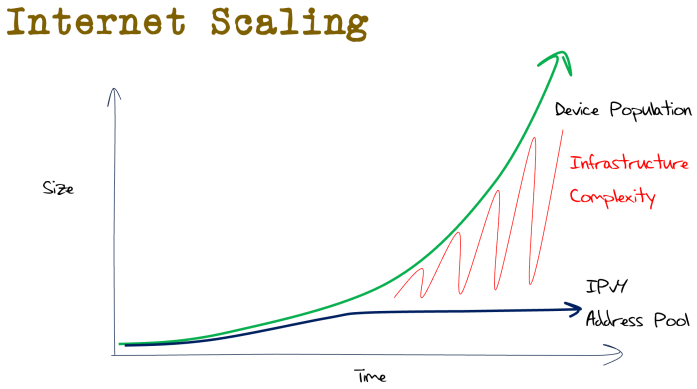

Figure 5 – Bridging the IPv4 Address Gap

At some point this embedded processing in the network becomes problematical. It's a cost element, and costs in a commodity network are intolerable in the long run.

The NAT response to the IPv4 address exhaustion situation gives us little leeway to reduce network complexity. Which means that if you want to continue to deploy digital services at scale, contain cost escalation to keep the service affordable and improve network robustness and security, then you have few choices left other than to reduce the network complexity burden. Deploying IPv6 is one obvious response as a way of achieving this.

Who received this memo? India got the memo in 2017 and rolled out a V6 network across a billion users in 12 months. That's getting the message at the scale and speed that they needed to achieve to be a part of today's world. (Figure 6)

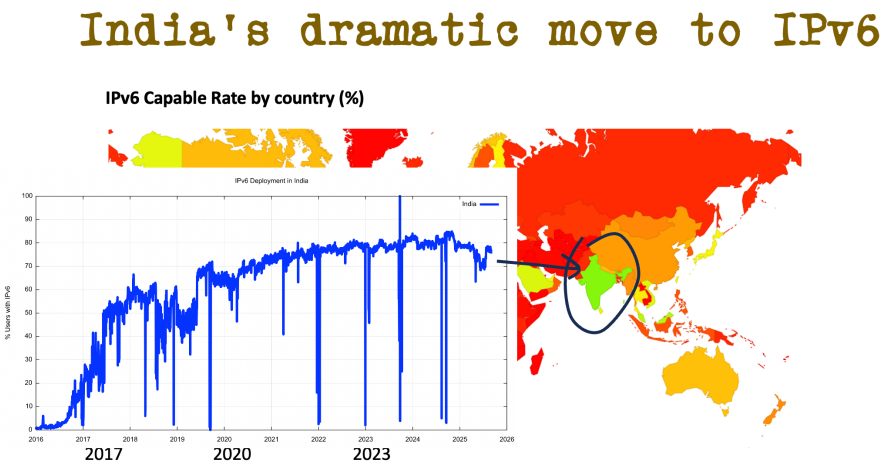

Figure 6 IPv6 in India

China is facing a similar set of problems but are proceeding at a more leisurely pace with their program of IPv6 adoption (Figure 7).

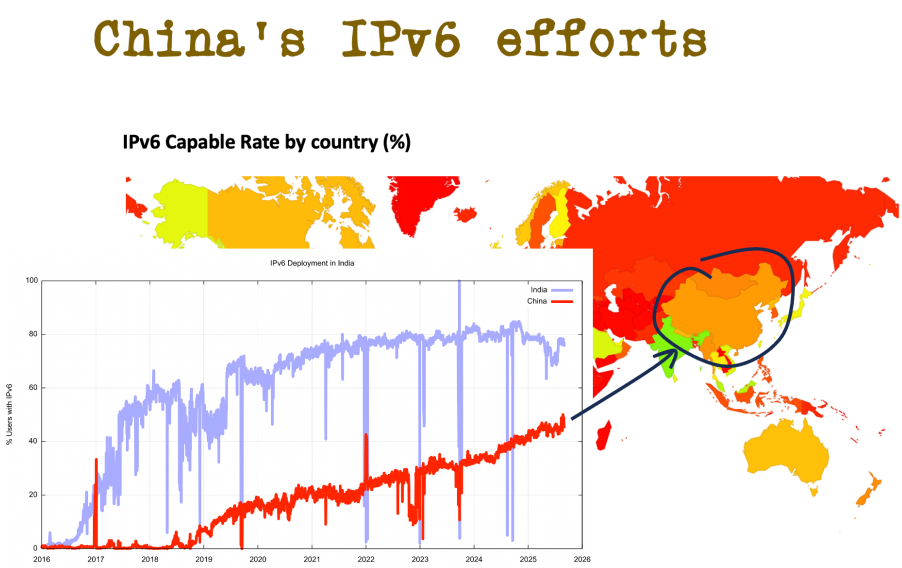

Figure 7 IPv6 in China

There's another tension going on here and it's the vertical tension inside the protocol stack. All the value is moving up the protocol stack into the application layer. The application is now the service.

The Client / Server network needs accessible rendezvous points for servers, or to be more correct, for services, but not for clients. In the public Internet clients do not need to be associated with a persistent IP address, and the QUIC transport protocol takes this a step further with the observation that the network can change the IP address of the client at will and as long as the address assignation is stable for a couple of round-trip time intervals, all will work! For servers' IP addresses persistence is not a strict prerequisite either. As long as it remains useable across some DNS resolver cache lifetime IP addresses for servers need not be stable for very long. The reason here is because it's the DNS namespace we use as the persistent service identity schema for the Internet. Our mechanisms for service authenticity and channel security are based on persistent service names, not on their IP addresses.

In the light of this, I can't help but ask: Do IP addresses matter? I have to observe that, yes, IP addresses do matter. But I must qualify that to add that they matter at the moment, but this is not necessarily an invariant property going forward. IP addresses are not persistent identity tokens, nor are they useful location tokens. They are useful tokens to pass into the routing system to assist packets to be passed to their intended destination, but even this is not quite what it seems. For example, the MPLS approach places a wrapper around incoming packet which identifies to the network the desired network egress point and a path to reach that egress point. On its transit across that MPLS network the inner IP protocol destination address is unused for forwarding the packet. So more accurately, addresses are used by individual networks to determine where to evict each packet from the network!

An extreme view is that IP addresses are ephemeral tokens to map from the service layer of named services to the underlaying layer of packet forwarding, and that's about it. Yes, we've attached a whole set of ancillary roles about reputation, accountability, location and reporting to IP address,

but perhaps these additional uses of addresses were making assumptions about the role and persistence of addresses which are unwarranted in the longer term.

We are not done. The silicon chip industry is still operating at full throttle, and there is the expectation that all of these processors will end up in Internet-connected devices in one form or another. We need to scale the network further. We have to scale the network further. It's service level scaling that's the challenge, right up at that top level of trying to orchestrate the behaviour of individual network components to generate coherent outcomes.

Do we know how we will achieve this? Not really. But don't forget that this is not a centrally planned global enterprise. Noone is in charge. No single country is in charge, despite some persistent suspicions about the role of the United States in the background. But such suspicions are more in the realm of conspiracy theory fodder than a clearly established national role. If there is no single country, then is it a group of countries? There is no specific international treaty to provide oversight for the Internet, no international convention or international protocol. There is no imposition of control over the Internet, or even a clear exposition of direction. The Internet is a diverse collection of markets, large and small. Markets are a means of orchestrating the behaviour of actors in producing coherent outcomes in a manner that also attempts, often imperfectly, to maximise efficiencies along the way. However, markets do not necessarily behave rationally or even predicably. When we gaze into the Internet's crystal ball and attempt to make such prognostications about its future, the vision is made as if through a glass darkly.

The presentation on which this article is based can be found at
https://www.potaroo.net/presentations/2025-10-30-evolution.pdf

A recording of the presentation can be found at:
https://www.youtube.com/live/9gvjMYs6XT0?t=27084s

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston AM, M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

www.potaroo.net