Everything here is my opinion. I do not speak for your employer.

← *March 2020* *November 2020* →

2020-07-08 »

IPv4, IPv6, and a sudden change in attitude

A few years ago I wrote *The World in Which IPv6 was a Good Design*. I'm still proud of that article, but I thought I should update it a bit.

No, I'm not switching sides. IPv6 is just as far away from universal adoption, or being a "good design" for our world, as it was three years ago. But since then I co-founded a company that turned out to be accidentally based on the principles I outlined in that article. Or rather, from turning those principles upside-down.

In that article, I explored the overall history of networking and the considerations that led to IPv6. I'm not going to cover that ground again. Instead, I want to talk about attitude.

Internets, Interoperability, and Postel's Law

Did you ever wonder why "Internet" is capitalized?

When I first joined the Internet in the 1990s, I found some now-long-lost introductory tutorial. It talked about the difference between an internet (lowercase i) and the Internet (capital I). An internet is "any network that connects smaller networks together." The Internet is... well... it turns out that you don't need more than one internet. If you have two internets, it is nearly unavoidable that someone will soon figure out how to connect them together. All you need is one person to build that one link, and your two internets become one. By induction then, the Internet is the end result when you make it easy enough for a single motivated individual to join one internet to another, however badly.

Internets are fundamentally sloppy. No matter how many committees you might form, ultimately connections are made by individuals plugging things together. Those things might follow the specs, or not. They might follow those specs well, or badly. They might violate the specs because everybody else is also violating the specs and that's the only way to make anything work. The connections themselves might be fast or slow, or flakey, or only functional for a few minutes each day, or subject to amateur radio regulations, or worse. The endpoints might be high-powered servers, vending machines, toasters, or satellites, running any imaginable operating system. Only one thing's for sure: they all have bugs.

Which brings us to Postel's Law, which I always bring up when I write about networks. When I do, invariably there's a slew of responses trying to debate whether Postel's Law is "right," or "a good idea," as if it were just an idea and not a force of nature.

Postel's Law says simply this: be conservative in what you send, and liberal in what you accept. Try your best to correctly handle the bugs produced by the other end. The most successful network node is one that plans for every "impossible" corruption there might be in the input and does something sensible when it happens. (Sometimes, yes, "something sensible" is to throw an error.)

[Side note: Postel's Law doesn't apply in every situation. You probably don't want your compiler to auto-fix your syntax errors, unless your compiler is javascript or HTML, which, kidding aside, actually were designed to do this sort of auto-correction for Postel's Law reasons. But the law does apply in virtually every complex situation where you need to communicate effectively, including human conversations. The way I like to say it is, "It takes two to miscommunicate." A great listener, *or* a skilled speaker, can resolve a lot of conflicts.]

Postel's Law is the principle the Internet is based on. Not because Jon Postel was such a great salesperson and talked everyone into it, but because that is the only winning evolutionary strategy when internets are competing. Nature doesn't care what you think about Postel's Law, because the only Internet that happens will be the one that follows Postel's Law. Every other internet will, without exception, eventually be joined to The Internet by some goofball who does it wrong, but just well enough that it adds value, so that eventually nobody will be willing to break the connection. And then to maintain that connection will require further application of Postel's Law.

IPv6: a different attitude

If you've followed my writing, you might have seen me refer to IPv6 as "a second internet that not everyone is connected to." There's a lot wrapped up in that claim. Let's back up a bit.

In *The World in Which IPv6 was a Good Design* I talked about the lofty design goals leading to IPv6: eliminate bus networks, get rid of MAC addresses, no more switches and hubs, no NATs, and so on. What I didn't realize at the time, which I now think is essential, is that these goals were a *fundamental attitude shift* compared to what went into IPv4 (and the earlier protocols that led to v4).

IPv4 evolved as a pragmatic way to build *an internet* out of a bunch of networks and machines that existed already. Postel's Law says you'd best deal with reality as it is, not as you wish it were, and so they did. When something didn't connect, someone hacked on it until it worked. Sloppy. Fits and starts, twine and duct tape. But most importantly, nobody really thought this whole mess would work as well as it turned out to work, or last as long as it turned out to last. Nobody knew, at the time, that whenever you start building internets, they always lead inexorably to The Internet.

These (mostly) same people, when they started to realize the monster they had created, got worried. They realized that 32-bit addresses, which they had originally thought would easily last for the lifetime of their little internet, were not even enough for one address per person in the world. They found out, not really to anyone's surprise, that Postel's Law, unyielding as it may be, is absolutely a maintenance nightmare. They thought they'd better hurry up and fix it all, before this very popular Internet they had created, which had become a valuable, global, essential service, suddenly came crashing down and it would all be their fault.

[Spoiler: it never did come crashing down. Well, not permanently. There were and are still short-lived flare-ups every now and then, but a few dedicated souls hack it back together, and so it goes.]

IPv6 was created in a new environment of fear, scalability concerns, and Second System Effect. As we covered last time, its goal was to replace The Internet with a New Internet — one that wouldn't make all the same mistakes. It would have fewer hacks. And we'd upgrade to it incrementally over a few years, just as we did when upgrading to newer versions of IP and TCP back in the old days.

We can hardly blame people for believing this would work. Even the term "Second System Effect" was only about 20 years old at the time, and not universally known. Every previous Internet upgrade had gone fine. Nobody had built such a big internet before, with so much Postel's Law, with such a variety of users, vendors, and systems, so nobody knew it would be different.

Well, here we are 25 years later, and not much has changed. If we were feeling snarky, we could perhaps describe IPv6 as "the String Theory of networking": a decades-long boondoggle that attracts True Believers, gets you flamed intensely if you question the doctrine, and which is notable mainly for how much progress it has held back.

Luckily we are not feeling snarky.

Two Internets?

There are, of course, still no exceptions to the rule that if you build any internet, it will inevitably (and usually quickly) become connected to The Internet.

I wasn't sitting there when it happened, but it's likely the very first IPv6 node ran on a machine that was also connected to IPv4, if only so someone could telnet to it for debugging. Today, even "pure IPv6" nodes are almost certainly connected to a network that, if configured correctly, can find a way to any IPv4 node, and vice versa. It might not be pretty, it might involve a lot of proxies, NATs, bridges, and firewalls. But it's all connected.

In that sense, there is still just one Internet. It's the big one. Since day 1, The Internet has never spoken just one protocol; it has always been a hairy mess of routers, bridges, and gateways, running many protocols at many layers. IPv6 is one of them.

What makes IPv6 special is that its proponents are not content for it to be *an internet* that connects to The Internet. No! It's the chosen one. Its destiny is to be The Internet. As a result, we don't *only* have bridges and gateways to join the IPv6 internets and the IPv4 internet (although we do).

Instead, IPv6 wants to eventually run directly on every node. End users have been, uh, *rather unwilling* to give up IPv4, so for now, every node has that too. As a result, machines are often joined directly to what I call "two competing internets" --- the IPv4 one and the IPv6 one.

Okay, at this point our terminology has become very confusing. Sorry. But all this leads to the question I know you want me to answer: Which internet is better!?

Combinatorics

I'll get to that, but first we need to revisit what I bravely called Avery's Laws of Wifi Reliability, which are not laws, were surely invented by someone else (since they're mostly a paraphrasing of a trivial subset of CAP theorem), and as it turns out, apply to more than just wifi. Oops. I guess the name is wrong in almost every possible way. Still, they're pretty good guidelines.

Let's refresh:

- Rule #1: if you have two wifi router brands that work with 90% of client devices, and your device has a problem with one of them, replacing the wifi router brand will fix the problem 90% of the time. Thus, an ISP offering both wifi routers has a [1 - (10% x 10%)] = 99% chance of eventual success.
- Rule #2: if you're running two wifi routers at once (say, a primary router and an extender), and both of them work "correctly" for about 90% of the time each day, the chance that your network has no problems all day is 81%.

In Rule #1, which I call "a OR b", success compounds and failure rates drop.

In Rule #2, which I call "a AND b", failure compounds and success drops.

But wait, didn't we add redundancy in both cases?

Depending how many distributed systems you've had to build, this is either really obvious or really mind blowing. Why did the success rate jump to 99% in the first scenario but drop to 81% in the second? What's the difference? And... which one of those cases is like IPv6?

Failover

Or we can ask that question another way. Why are there so many web pages that advise you to solve your connectivity problem by disabling IPv6?

Because *automatic failover* is a very hard problem.

Let's keep things simple. IPv4 is one way to connect client A to server X, and IPv6 is a second way. It's similar to buying redundant home IPv4 connections from, say, a cable and a DSL provider and plugging them into the same computer. Either way, you have two independent connections to The Internet.

When you have two connections, you must choose between them. Here are some factors you can consider:

- Which one even offers a path from A to X? (If X doesn't have an IPv6 address, for example, then IPv6 won't be an option.)
- Which one gives the shortest paths from A to X and from X to A? (You could evaluate this using hopcount or latency, for example, like in my old netselect program.)
- Which path has the most bandwidth?
- Which path is most expensive?
- Which path is most congested right now?
- Which path drops out least often? (A rebooted NAT will drop a TCP connection on IPv4. But IPv6 routes change more frequently.)
- Which one has buggy firewalls or NATs in the way? Do they completely block it (easy) or just act strangely (hard)?
- Which one blocks certain UDP or TCP ports, intentionally or unintentionally?
- Which one is misconfigured to block certain ICMP packets so that PMTU discovery (always or sometimes) doesn't work with some or all hosts?
- Which one blocks certain kinds of packet fragmentation?

A common heuristic called "Happy Eyeballs" is one way to choose between routes, but it covers only a few of those criteria.

The truth is, it's extremely hard to answer all those questions, and even if you can, the answers are different for every combination of A and X, and they change over time. Operating systems, web browsers, and apps, even if they implement Happy Eyeballs or something equivalent, tend to be pretty bad at detecting all these edge cases. And every app has to do it separately!

My claim is that the "choose between two internets" problem is the same as the "choose between two flakey wifi routers on the same SSID" problem (Rule #2). All is well as long as both internets (or both wifi routers) are working perfectly. As soon as one is acting weird, your overall results are going to be weird.

...and the Internet *always* acts weird, because of the tyranny of Postel's Law. Debugging the Internet is a full time job.

...and now there are two internets, with a surprisingly low level of overlap, so your ISP has to build and debug both.

...and every OS vendor has to debug both protocol implementations, which is more than twice as much code.

...and every app vendor has to test with both IPv4 and IPv6, which of course they don't.

We should not be surprised that the combined system is less reliable.

The dream

IPv6 proponents know all this, whether rationally or intuitively or at least empirically. The failure rate of two wonky internets joined together is higher than the failure rate of either wonky internet alone.

This leads them to the same conclusion you've heard so many times: we should just kill one of the internets, so we can spend our time making the one remaining internet less wonky, instead of dividing our effort between the two. Oh, and, obviously the one we kill will be IPv4, thanks.

They're not wrong! It *would* be a lot easier to debug with just one internet, and you know, if we all had to agree on one, IPv6 is probably the better choice.

But... we don't all have to agree on one, because of the awesome unstoppable terribleness that is Postel's Law. Nobody can declare one internet or the other to be officially dead, because the only thing we know for sure about internets is that they always combine to make The Internet. Someone might try to unplug IPv4 or IPv6, but some other jerk will plug it right back in.

Purity cannot ever be achieved at this kind of scale. If you need purity for your network to be reliable, then you have an unsolvable problem.

The workaround

One thing we can do, though, is build better heuristics.

Ok, actually we have to do better than that, because it turns out that correctly choosing between the two internets for each connection, at the start of that connection, is not possible or good enough. Problems like PMTU, fragmentation, NAT resets, and routing changes can interrupt a connection partway through and cause poor performance or dropouts.

I want to go back to a side note I left near the end of *The World in Which IPv6 was a Good Design*: mobile IP. That is, the ability for your connections to keep going even if you hop between IP addresses. If you had IP mobility, then you could migrate connections between your two internets in real time, based on live quality feedback. You could send the same packets over both links and see which ones work better. If you picked one link and it suddenly stopped, you could retransmit packets on the other link and pick up where you left off. Your precise heuristic wouldn't even matter that much, as long as it tries both ways eventually.

If you had IP mobility, then you could convert the "a AND b" scenario (failure compounds) into the "a OR b" scenario (success compounds).

And you know what, forget about IPv4 and IPv6. The same tricks would work with that redundant cable + DSL setup we mentioned above. Or a phone with both wifi and LTE. Or, given a fancy enough wifi client chipset, smoothly switching between multiple unrelated wifi routers.

IP mobility is what we do, in a small way, with Tailscale's WireGuard connections. We try all your Internet links, IPv4 and IPv6, UDP and TCP, relayed and peer-to-peer. We made mobile IP a real thing, if only on your private network for now. And what do you know, the math works. Tailscale's use of WireGuard with two networks is more reliable than with one network.

Now, can it work for the whole Internet?

This article was originally posted to the Tailscale blog

Related
Systems design explains the world: volume 1 (2020)

The world in which IPv6 was a good design (2017)

Unrelated
Git's DAG is just a bunch of branches (2009)

Try my project! **Tailscale**: a new, magically easy mesh VPN based on WireGuard.

Why would you follow me on twitter? Use RSS.

apenwarr-on-gmail.com