February 2021
Geoff Huston

Notes from NANOG 81

As the pandemic continues, the network operational community continues to meet online. NANOG held its 81st meeting on February 8 and 9, and these are my notes from some of the presentations at that meeting.

A Brief History of Router Architecture

Ethernet, developed in 1973 at Xerox PARC, was a revolutionary step in terms of network architectures in many ways. The common bus architecture imposed a number of constraints on the network that have echoed through the ensuing four decades in all kinds of ways. Ethernet used a packet-based technology, not a virtual circuit technology, nor a fixed-sized cell technology. The original CSMA/CD model did not use time division switches, so Ethernet switches did not require highly stable (and expensive) high speed oscillators. Ethernet was not a reliable medium, in that a sender would not be told by the Ethernet medium whether the packet was received by the intended recipient or not. There was no assured delivery time for packets. Instead of fixed-sized packets (or cells) Ethernet payloads were defined to be packets between 46 to 1,500 octets in size.

One of the implications of Ethernet's design parameters is the maximum packet rates of Ethernet networks. Using a 10Mbps Ethernet cable network, the total packet throughput of Ethernet networks was between 14,880 packets per second for minimal-sized packets and 810 packets per second for large packets. That's 67µs per packet for small packets and 1.2ms per packet for large packets. By the 1980's, when Ethernet gained industry attention, the processing capabilities of switching equipment was able to meet these timing demands, and Ethernet switching equipment was available that performed two port switching at the maximal packet rate of 14,880 packets per second.

Dedicated routing equipment in the 80's followed a conventional computer architecture. A router was a set of network interface cards (NICs) whose task was to assemble an incoming packet and pass the packet on a common bus to the CPU. Software resident in the CPU would perform a lookup and push the packet back on the bus to an outbound NIC card (Figure 1). As long as the capacity of the bus was double the aggregate capacity of the NIC cards, and CPU was also able to process the sum of the packet rate of the NIC cards, then the unit would be able to keep pace with the switching load.

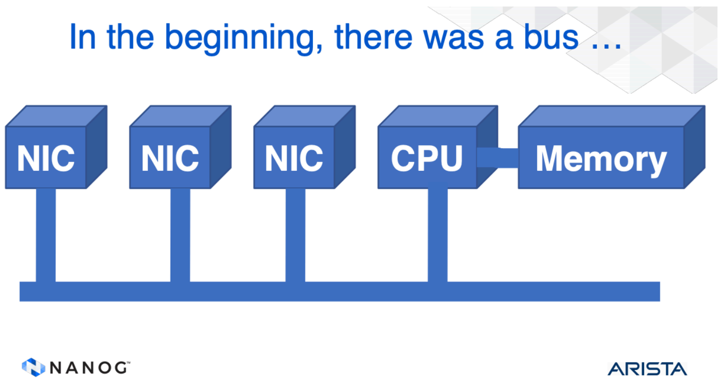


Figure 1 – Early Router Architecture
From: Tony Li, Arista, A History of Router Architecture: https://bit.ly/2Nqt0ne

Scaling the network meant that routers needed more NIC cards, which, in turn, called for faster buses and faster processors and memory. Then we also started looking at serial line transmission systems where the line transmission speeds grew to 100's of Mbps then Gbps. Through all of these changes in line speeds we used the same packet size limitations. A 2.5Gbps circuit is running 250 million times faster than the original 10Mbps speed but has the same packet sizes. That implies a 2.5G single circuit can generate 4M small (64 octet) packets per second and some 200,000 large (1,500 octet) packets per second. If a NIC care has 8 or 16 such interfaces, then the packet numbers that the router needs to handle scales accordingly.

The challenge for switch designers over this same period has been to improve packet processing capability without being able to simply increase the clock speed on the bus, processor and memory. Basic memory cycle times have not changed for more than 10 years, and processor speeds have similarly plateaued. In such cases, where it's not possible to just increase the clock speed, the typical response from hardware designers is to use parallel processing paths and provide increased performance by processing packets in parallel. One way to do this is to use multiple buses within the unit and use a processor between the buses.

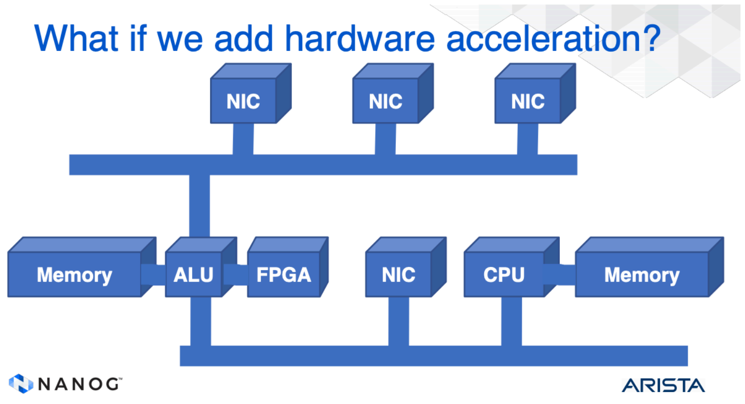


Figure 2 – Multi-Bus Router Architecture
From: Tony Li, Arista, A History of Router Architecture: https://bit.ly/2Nqt0ne

But the approach does not really scale. If we want to remove the common bottleneck of the bus and the processor then the next step was to use a crossbar switch as the internal switching fabric. An incoming packet in a NIC was scheduled to be switched to an output NIC and was passed thorough the switch lanes of the unit. This is a higher capacity architecture than a collection of common buses, but the issue of head-of-line blocking still remains, and there is a need for both input and output queues. We can replace the crossbar switch with a fully connected mesh, or a partial mesh between NIC cards, where each NIC can pass a packet directly to a set of NICS or any other NIC. In some ways this is equivalent to using a bundle of dedicated internal buses, each bus interconnecting just 2 NICs.

In searching for higher switching capacity router designers rediscovered earlier research from Bell by Edson Erwin in 1938 and Charles Clos in 1953 in the context of circuit switches. In his 1953 paper Clos described how telephone calls could be switched with equipment that used multiple stages of interconnection to allow the calls to be completed. The switching points in the topology are called crossbar switches. Clos Switches were designed to be a three-stage architecture, an ingress stage, a middle stage, and an egress stage. The concept is that there are multiple paths for the call to be switched through the network so that calls will always be connected and not "blocked" by another call.

This results in a Clos Router architecture, where the NICs split input packets into fixed length cells and pass them into a three stage Clos switch. The number of switches can be reduced by using a "folded" Close architecture, but the scaling pressure on switching capacity still exists. Each input needs to use a queue per output and call addressing is finite so the number of interfaces to the switch fabric is limited (Figure 3).

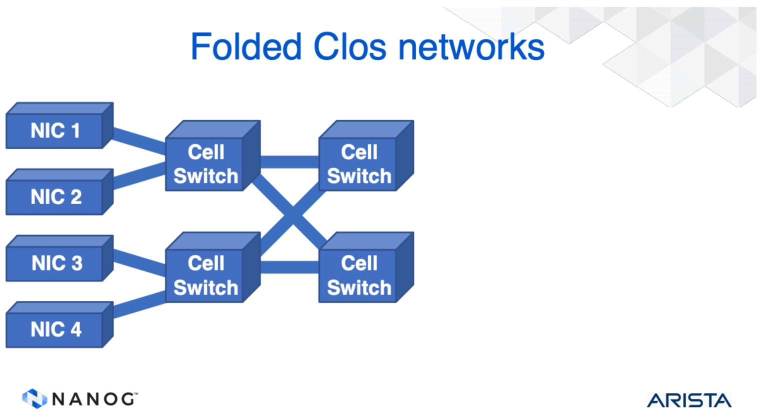
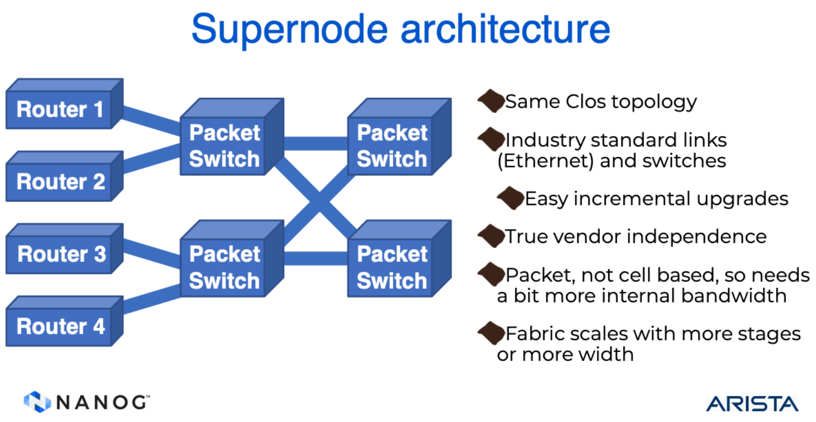


Figure 3 – Folded Clos Router Architecture
From: Tony Li, Arista, A History of Router Architecture: https://bit.ly/2Nqt0ne

Why use cells for switching? We can use the switching frameworks used in data centres and design a "super node" by replacing the cell switches with packet switches and the NICs with Routers. With an appropriate internal routing architecture, such as hierarchical IS-IS or RIFT, for example. the entire switching fabric can be used (Figure 4).



- Same Clos topology
- Industry standard links (Ethernet) and switches
- Easy incremental upgrades
- True vendor independence
- Packet, not cell based, so needs a bit more internal bandwidth
- Fabric scales with more stages or more width

Figure 4 – Super Nodes
From: Tony Li, Arista, A History of Router Architecture: https://bit.ly/2Nqt0ne

This form of POP design can be seen in Comcast's service platform, where "Core Clusters" use sets of leaf routers (Regional Aggregation Routers) which are connected to a set of Core Spine Routers, which are collected to a set of Core Routers. With extensive use of ECMP paths, the objective is to evenly load the traffic within the POP.

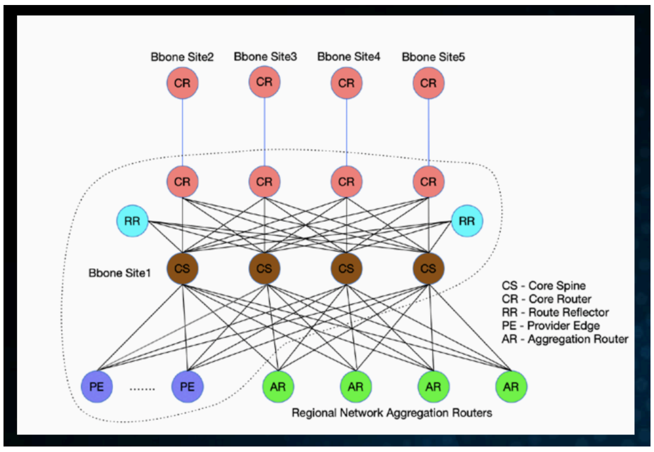


Figure 5 – Core POP Design
From: Mannan Venkatesan, Comcast, Next Gen Core Networks. https://bit.ly/3pHYnqI

There are two parameters of growth in their network. The first is the number of ports per node, and secondly the capacity of each port. Both numbers are rising, with the port capacity growing from 100G to 400G in recent years. Tracking this demand curve is challenging, particularly when the lead time for new hardware certification and deployment operates in 2-3 year cycles, and the hardware refresh in the field takes 1 year as well.

The overall node design also has to factor in what was termed a "failure blast radius". Any form of service withdrawal of a critical core router could impact not only the locally attached services but also have impact on the backbone network, peer links and content delivery. The objective hear is to minimize the impact of failure of any single unit, so that isolated failures do not generate a cascading failure scenario across the larger network.

There is a tension between packet handling features and packet handling capacity. ASIC-based switching bandwidth is inversely proportionate to features supported on the switch. Stripping packet handling features from the switch results in higher capacity ports at a constant cost.

The network operates at levels close to capacity, which implies that there are few windows available for operational maintenance and incremental upgrades, and over time these windows are getting shorter.

The highly meshed design that distributes load over a collection of paths places stress on the interior routing systems, and the older "best path" routing algorithms are of little use in such networks. The requirement these days is to maintain a set of paths between network ingress and egress, and distribute traffic across such paths in a balanced fashion that still respects the packet order within each micro-flow.

The network platform operator wants to automate device, cable, device and code deployment and manage upgrades. The network element and system management tools are not keeping up with the requirements of these large-scale networks. When they opened up the single routers and exposed a CLOS-connected spine and leaf fabric they then had a need to monitor the internals of the exposed CLOS fabric. These days the network platform operator needs to monitor CLOS performance, fabric links, ISIS, PIM, BGP, plus stability, churn, policies and route redistribution.

There are a number of tensions in this space. We are still in the space where carriage capacity is being constructed that is in excess of silicon switching capacity, so we are continually working on silicon designs that break out the traffic flows to stay within silicon capacity. There is also the desire to create a richer set of packet handling capabilities deeper within the network, which further exacerbates the problem.

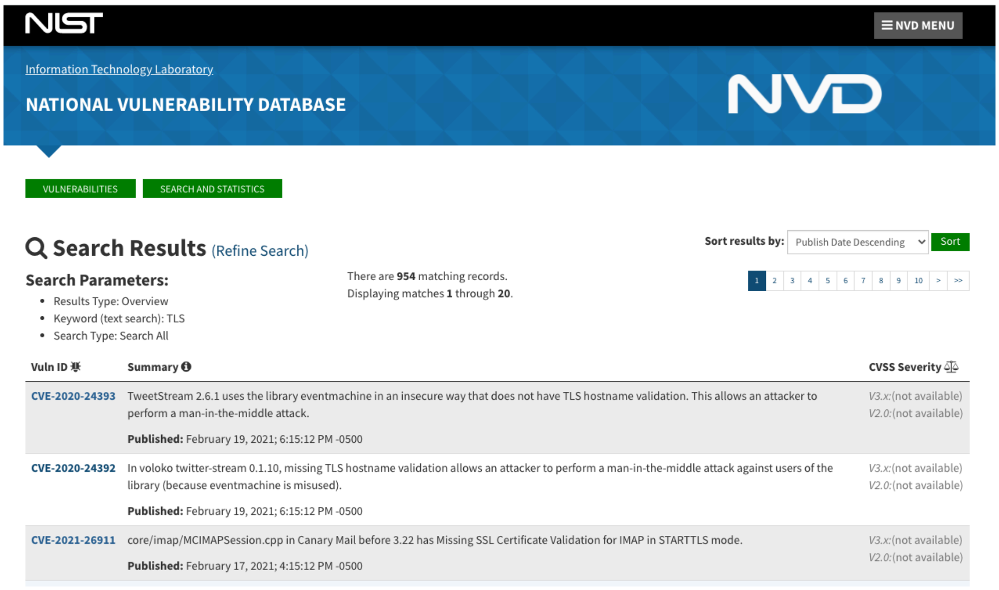
This is not a new problem by any means. It appears that the common theme of the Internet's growth over the past thirty years has been one where the capabilities of the infrastructure is the limiting factor, while the underlying dynamics of demand continue to completely outpace the delivery capacity of these platforms. There is no reason to suspect that this will change anytime soon.

Securing Internet Applications from Routing Attacks

It's a common observation that if routing attacks are so easy to mount, then why don't we take the topic of routing security more seriously? Why do we feel the need to actively market various routing security-related practices? If we all were so concerned about the risks of being impacted by routing hijacks, surely we would all be enthusiastic practitioners of "safe" routing practices?

The response I often get to this question is based around HTTPS. The response points out that a) the entire Internet uses HTTPS, HTTPS uses TLS, and b) TLS is awesome! A routing attack might be able to misdirect a user to the 'wrong' destination, but if HTTPS is being used then the attacker cannot complete the connection as they do not have knowledge of the "genuine" private key and therefore cannot present a valid certificate that will kick off the TLS session establishment process. TLS protects the end user from being duped by such an attack.

The problem with this response is that both parts of this response are untrue. There is a whole world of applications and protocols that exist outside the world of browsers and HTTPS connections, and TLS itself is not exactly bullet-proof. It is a sobering exercise to search for TLS vulnerabilities in the NIST National Vulnerability Database (Figure 6). Yes, that number from the screenshot really is 954 reported vulnerabilities!


NIST
NVD MENU
Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
NVD
VULNERABILITIES
SEARCH AND STATISTICS

Search Results (Refine Search)

Sort results by: Publish Date Descending Sort

Search Parameters:

- Results Type: Overview
- Keyword (text search): TLS
- Search Type: Search All

There are **954** matching records.
Displaying matches **1** through **20**.

1 2 3 4 5 6 7 8 9 10 > >>

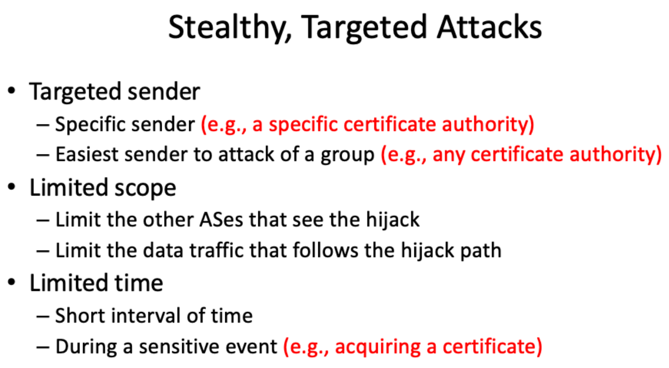
Vuln ID	Summary	CVSS Severity
CVE-2020-24393	TweetStream 2.6.1 uses the library eventmachine in an insecure way that does not have TLS hostname validation. This allows an attacker to perform a man-in-the-middle attack. **Published:** February 19, 2021; 6:15:12 PM -0500	*V3.x:*(not available) *V2.0:*(not available)
CVE-2020-24392	In voloko twitter-stream 0.1.10, missing TLS hostname validation allows an attacker to perform a man-in-the-middle attack against users of the library (because eventmachine is misused). **Published:** February 19, 2021; 6:15:12 PM -0500	*V3.x:*(not available) *V2.0:*(not available)
CVE-2021-26911	core/imap/MCIMAPSession.cpp in Canary Mail before 3.22 has Missing SSL Certificate Validation for IMAP in STARTTLS mode. **Published:** February 17, 2021; 4:15:12 PM -0500	*V3.x:*(not available) *V2.0:*(not available)

Figure 6 – NIST National Vulnerability Database records of TLS vulnerabilities (23 Feb 2021)

So, if the answer as to why routing security is not a serious problem is that we all use TLS, and TLS is simply fantastic, then we are missing the point!

By its very design, securing the eBGP inter-domain routing environment is extraordinary challenging, as the last thirty years of effort in this space illustrates. While it would be nice to think that we will deploy a "solution" to the routing security question that will wrap the operational routing system in some form of impregnable cloak that will resist all attempts at unauthorised subversion, this is much closer to an example of wishful thinking than to a confident prediction!

This line of thought was behind Jennifer Rexford's presentation to NANOG on "Securing Internet Applications from Routing Attacks." If we can't secure the routing system how can we design applications that will function correctly in the face of a routing attack. The attack scenario looked at in her presentation is the "targeted attack" (Figure 7)



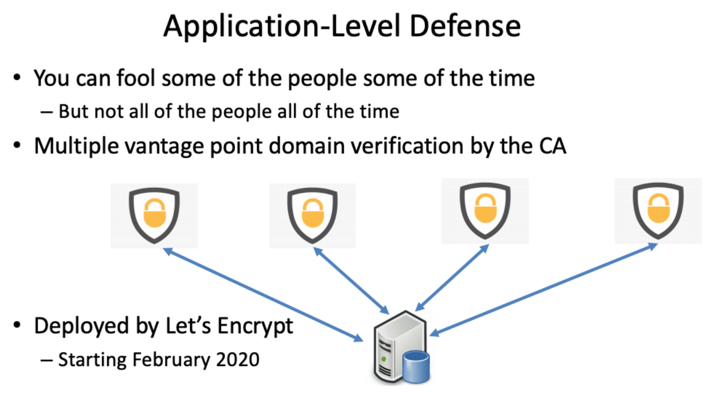
Stealthy, Targeted Attacks

- Targeted sender
  - – Specific sender (e.g., a specific certificate authority)
  - – Easiest sender to attack of a group (e.g., any certificate authority)
- Limited scope
  - – Limit the other ASes that see the hijack
  - – Limit the data traffic that follows the hijack path
- Limited time
  - – Short interval of time
  - – During a sensitive event (e.g., acquiring a certificate)

Figure 7 – Targeted Routing Attacks
Jennifer Rexford, Princeton University, https://bit.ly/2NzdqFX

The application examined here is that of the Certification Authority. If TLS really is the bulwark of authenticity on the Internet, then duping a CA to issue a certificate for the target site using the attacker's private key is the objective, as we've seen numerous times already. These days certification is based on automated checks, where the applicant needs to demonstrate to the CA that they have control of the domain. This can be demonstrated to the CA by adding a record to the domain's DNS zone, or by adding a field to the domain's web site.

How can an application defend itself from targeted attacks? One method is that of multiple vantage points, where the test is not performed by a single server located in a single vantage point, but by multiple servers at diverse locations (Figure 8). The results need to agree within the bounds of a quorum vote from the testing servers. This approach was adopted by Let's Encrypt as of February 2020, and has had positive operational experience so far.



Application-Level Defense

- You can fool some of the people some of the time
  - But not all of the people all of the time
- Multiple vantage point domain verification by the CA
- Deployed by Let's Encrypt
  - Starting February 2020

Figure 8 – Application-Level Defense
Jennifer Rexford, Princeton University, https://bit.ly/2NzdqFX

Not every application can be altered to use a multi-vantage point defence, and not every routing attack can be foiled in this way. However, it does illustrate that cross-layer attacks probably require cross-layer defences, where the application behaviour itself does not relay on a secure and trusted network layer.

A more abstract view of the dilemma in security by design was provided by Russ White in his presentation on security by design. As Bruce Schneier pointed out: "The Internet and all the systems we build today are getting more complex at a rate that is faster than we are capable of matching. Security in reality is actually improving, but the target is constantly shifting. As complexity grows, we are losing ground." [https://www.schneier.com/news/archives/2012/12/complexity_the_worst.html] The consequent question is: "Can we contain the complexity of these systems?" Russ' answer is not all that encouraging: "Reducing complexity locally almost always leads to increasing complexity globally. Optimizing locally almost always leads to decreasing optimisation globally." [https://bit.ly/2NNUd3k] Oh dear!

Disclaimer

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

Author

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

www.potaroo.net