Network Protocol Folklore

Perlman, Radia
Dell Technologies
radia.perlman@dell.com

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT

There's no way to understand today's networks without knowing the history. Too often, network protocols are taught as 'memorize the details of what is currently deployed', which creates a lot of confusion, and certainly does not encourage critical thinking. Some decisions have made today's networks unnecessarily complex and less functional. But surprisingly, mechanisms that were created out of necessity, to compensate for previous decisions, sometimes turn out to be useful for purposes other than the original reason they were invented. If the world had adopted a different approach originally, some very interesting technology may not have been invented. Given limited space in this article, we will not worry about exact details, but rather convey the main conceptual points, and only cover a few examples.

CCS CONCEPTS

• **Networks → Network protocol design**; **Network layer protocols**; **Routers**; **Bridges and switches**; **Local area networks**;

KEYWORDS

Network Layer, Routing Protocols, Ethernet, CLNP, IP

1 NETWORK LAYERS

It is common to introduce network protocols using the OSI 7-layer model. This article will focus on layers 2 and 3. The distinction between layers 2 and 3 has become quite blurred, but I claim that layer 2 is a way to send a packet to a neighbor node that shares the same link as the transmitter, and layer 3 forwards between links.

2 SPANNING TREE ETHERNET

2.1 Original Ethernet Concepts

Ethernet as originally invented was a way to inexpensively hook together few hundred nodes, within a limited distance. Conceptually, the original Ethernet design is a single wire. If one node transmits, all other nodes hear the transmission. If more than one node transmits at the same time, the messages get garbled (this is known as a *collision*). The mechanism for keeping all the nodes equal (no master node that gives other nodes permission to speak), minimizing bandwidth loss due to collisions and management traffic overhead, and keeping the design simple was known as CSMA/CD. CS is *carrier sense*, meaning listen before you talk and don't interrupt if someone else is transmitting. MA is *multiple access*, meaning lots of nodes are all on the same medium. CD is *collision detect*, meaning that even while you're transmitting, you should listen to see if someone else has started talking, in which case you should stop, wait a random amount of time, and try again when nobody else is talking.

An interesting aspect of Ethernet (aside from CSMA/CD) was the chosen size of the address. Ethernets were never meant to have more than a few hundred nodes attached, and yet the address chosen was 6 bytes long. In contrast, IPv4, intended to be such that every node in the Internet could have its own address, had only 4 bytes. The rationale for choosing 6 bytes for Ethernet was that with such a large address, Ethernets would not require any address configuration. Every device with an Ethernet interface would be installed, at manufacture time, with a unique 6-byte Ethernet address, and use that address wherever it attached to other Ethernet nodes. The way this was accomplished was to have an organization (IEEE) sell blocks of addresses to manufacturers. A block consists of a 24-bit prefix, allowing the manufacturer who purchases the block to have 2^{24} unique addresses. In contrast, the IP address is location-dependent.

2.2 Original Ethernet vs Layer 3

In the early 1980's, I was designing layer 3 of DECnet, which meant designing a routing protocol…a distributed algorithm run by routers that enables them to create forwarding tables. The particular protocol that I designed was named IS-IS (where IS, short for *intermediate system*, was a fancy name for a router. IS-IS was so named because it was a protocol spoken between IS's). Although IS-IS was originally designed for forwarding DECnet (Phase V) data, it can be used with any layer 3 protocol, including IP (and it is widely used by Internet Service Providers).

DECnet originally had a 2-byte address. Realizing that that was too small, rather than inventing a new data packet format, we adopted (for DECnet Phase V) ISO's CLNP (ConnectionLess Network Protocol). The CLNP format has 20-byte addresses. In contrast, IPv4 has 4-byte addresses, and IPv6 has 16-byte addresses. CLNP, IS-IS, spanning tree, and other concepts are described in detail in [2].

When Ethernet arrived, it was a new type of network link and routing protocols needed to be slightly modified to accommodate it. One example where treating Ethernet as just another link would have led to suboptimality is that IS-IS requires each router to report a list of its neighbors, and each router to store all the other routers' reports. If an Ethernet with, say, n nodes were treated as full connectivity between all the nodes, where each of the nodes reported each of the other nodes as a neighbor, this would have resulted in n^2 link reports. So, I introduced the concept of modeling an Ethernet within the routing protocol as a pseudonode, where each router only reports connectivity to the pseudonode (rather

Figure 1: Pseudonodes

than reporting connectivity to all the other nodes on the link). That results in n+1 nodes and n links.

But other than requiring a bit of minor redesign to the routing protocol, Ethernet was just another link in the network.

Unfortunately, at the time, many people thought that Ethernet was a replacement to layer 3. For instance, the Digital proprietary protocol LAT (Local Area Transport) was implemented directly over Ethernet, i.e., did not have layer 3 in the network stack in the LAT nodes. That meant that the protocol could only work between nodes on a single Ethernet (it was *unroutable*).

If only the inventors of Ethernet had called it 'EtherLink' instead of 'EtherNet'. Applications like LAT were useful and worked well, but would have been just as good if they'd been implemented on top of layer 3. Better even, because it would have allowed the applications to work among nodes on different Ethernets.

The Ethernet header looks very similar to a layer 3 header. There is a source and destination address. However, layer 3 headers should contain an additional field known as a *hop count*. The reason for this is that when the topology changes, the forwarding tables in all the routers cannot be simultaneously switched to the new topology. For some time after topology changes, forwarding tables can be inconsistent, and packets may wander around in circles. The hop count is a field that is decremented by each router as it forwards the packet, so that when the hop count reaches zero, routers delete the confused packet. Ethernet has no such field because the Ethernet designers were intending the protocol to be a layer 2 protocol. A layer 2 protocol should be confined to a single link, and not forwarded, so there would be no need for a hop count in a layer 2 protocol.

2.3 Spanning Tree Bridging

Then (somewhere around 1985), it became apparent that indeed, customers wanted their applications to work between nodes on different Ethernets. Although DECnet routers certainly were intended to serve that function, routers depended on endnodes to implement layer 3. Routers do not forward based on the Ethernet header. Some people (including me) argued that the right solution was to fix the endnodes to put layer 3 into the network stack of the attached nodes rather than try to invent some magic box that could glue Ethernets together despite endnodes not implementing layer 3.

However, the assumption was that deploying a magic box (to be known as a *bridge*) could be done sooner than fixing the endnodes to be layer 3-aware. Inventing and deploying a bridge would be a short-term fix until all the endnodes were upgraded (at least that's what I assumed at the time).

This is where spanning tree bridges came in. The basic concept is that a bridge listens to all packets transmitted on each port, stores each packet in a queue to be transmitted on the other ports, and when the Ether is free (if the outgoing port is an Ethernet), or it gets a token (if the next link is a token ring), the bridge transmits the packet onto the next link. As an optimization, the bridge looks at the source address, S, on a packet received on port x, and remembers that S is located on port x. Learning the location of the source enables the bridge to cut down on flooding. If a bridge has learned that S resides on port x, and then receives a packet for destination=S, it need only forward it onto port x. If a packet received on port x is destined for S, the bridge need not forward it at all.

This concept works unless there are loops in the topology. Especially without hop counts, if there are loops, nothing will prevent packets from circulating forever, and even multiplying exponentially. One strategy would be to tell customers "make sure you don't plug your network into loops". But loops are not merely misconfiguration. They allow alternate paths in case things break.

So, I was asked to figure out a distributed algorithm for bridges to break loops. The constraints were:

- No changes required of endnodes
- No spare fields in the Ethernet header
- An absolute maximum size of an Ethernet packet. (Since endnodes assumed they could transmit maximum sized packets, there was no room to add additional headers.)
- No configuration required of either endnodes or bridges

The result was the spanning tree algorithm [1], where bridges collaborate to find a loop-free subset of the topology for forwarding data packets. (For the record, I spent 5 days on creating the spanning tree algorithm -- two days to invent it, prove it worked, and write the specification, and 3 days to write the poem that was the abstract of the paper in which I published the algorithm. The poem is called "Algorhyme" and is easily found on the Internet.)

Despite having invented the spanning tree algorithm (and written the poem), I still thought it was a kludge, and the right solution was fixing the endnodes to include layer 3 in the network stack. It seems plausible that spanning tree bridging would never have been invented if people hadn't deployed unroutable protocols.

Comparing Ethernet-as-a-link in a true layer 3 protocol vs bridged Ethernets, the layer 3 solution is better. The disadvantages of spanning tree;

- It forwards data on a loop-free subset of the topology, so it does not always use optimal paths
- Links that are not selected for the tree are unused
- Bridges need to flood to unknown destinations
- Spanning tree cannot use multiple paths to reach the destination.
- Forwarding based on the Ethernet header is risky because of the lack of a hop count.

But since (I assumed) spanning tree Ethernet was only going to last for a few months until people fixed the endnodes to speak layer 3, it seemed OK to deploy spanning tree bridging. (Note: the 'few

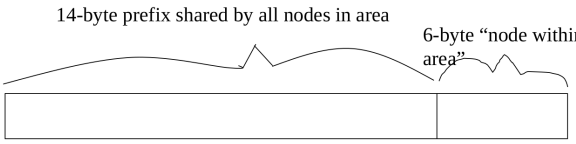

Figure 2: CLNP Addressing

months' of spanning tree bridging started in 1985 and as of today, remains virtually unchanged and widely deployed).

Within a few years after spanning tree bridging was created, shared link (CSMA/CD) Ethernet was completely replaced by a network connected with point-to-point links and bridges. When there are only two nodes on a link, there is no need for CSMA/CD. Variants of CSMA/CD still exist for wireless links, but when someone refers to an 'Ethernet' today, it does not include CSMA/CD. It is just the original Ethernet header and spanning tree bridges.

Although bridging was specifically designed because of applications that were built without layer 3, it had some unexpected nice side effects.

- At that point in time, there were many deployed layer 3 protocols (e.g., IPv4, CLNP, older versions of DECnet, Appletalk, IPX). Some companies provided multiprotocol routers that handled some subset of these protocols, but they were expensive and low performance. In contrast, bridges forward everything and were cheap and fast.
- Bridged Ethernets create fairly large networks that look to IP like a single link. This enables IP to be much more usable than if the world depended on everything being connected with IP routers (see section on CLNP).
- CSMA/CD Ethernet does not work with higher speeds. because CSMA/CD requires noticing a collision. Given speed of propagation of the signal, to increase the speed by a factor of x, the maximum length of the wire would have to be shortened by a factor of x.

2.4 CLNP Concepts

In the 1980's, there were many competing layer 3 protocols. However, in the interest of space, let's focus on IPv4 (at the time) and CLNP. IPv4 has 4-byte addresses. CLNP has 20-byte addresses. (We're simplifying a bit…CLNP addresses were actually variable length, though the common usage was 20-bytes.) Note that IPv6 has 16-byte addresses.

In addition to having large addresses, CLNP has some other important properties. The most important property, and the one that most distinguishes CLNP from IP, is *area routing*, the 3rd bullet point below:

- Autoconfiguration of endnode addresses: CLNP's 20-byte address is partitioned into two parts. The top 14 bytes of the CLNP address is a prefix shared by all nodes within a region known as an *area*. An endnode is informed of the 14-byte prefix by the router on the link, and then the endnode can complete its 20-byte CLNP address by inserting its 6-byte Ethernet address into the bottom 6 bytes.
- Inter-area routing: The top 14 bytes of the CLNP address is hierarchically assigned, so that regions of the network can be summarized with a prefix. This technique of summarizing portions of the network with a prefix, and forwarding based on longest prefix match, is a common technique in network protocols for minimizing the size of routing tables. IPv4 also uses this technique, although with CLNP's 14-byte hierarchy, there is much better flexibility and scalability than with IP's 4 byte address.
- Area routing: Within an area, all CLNP nodes share the same 14-byte prefix. An area can have hundreds of thousands of nodes, and have complex topologies (lots of links, no restrictions such as requiring the links to be structured into a tree). Routing within an area uses normal layer 3 routing (IS-IS), where the destinations in the router's forwarding table are specific endnodes within the area. Routers find out where the endnodes are because endnodes announce themselves to their nearest router, and IS-IS spreads endnode location information to the rest of the routers in that area. An endnode can move within the area without changing its CLNP address. An endnode can have multiple links, and the path chosen by routers to reach that endnode will be via whichever of the endnode's links yields the best path.

Let's use the term 'area' to refer to a portion of the network with a flat address. One might be concerned about the size of the forwarding tables required of CLNP area routers, because the forwarding table inside an area cannot be summarized with prefixes; it has to include every endnode within the area. However, a forwarding table of a few hundred thousand nodes is not that difficult to implement. Note that IP does not have this concept of having an area with a flat address space, with destinations inside the area being endnodes that can move freely within the area without changing their layer 3 address. IP depends on Ethernet to create the illusion (to IP) of large links where nodes can move and keep their IP address. CLNP area routing can certainly support bigger networks than bridged Ethernet.

2.5 Contrasting CLNP with IP

CLNP's provision of area routing is a subtle and important difference between CLNP and IP. In IP, every link must have a unique block of addresses. If an IP node moves from one side of an IP router to another, its IP address must change. In contrast, with CLNP, a node can move, as long as it is within its area, and keep its CLNP address. This ability to have a large portion of the network with a flat address space is very useful for building things like data centers with virtualization, where VMs can be easily moved.

Another nice feature of CLNP's areas is that routers within an area need not be configured with addresses. Some router within the area needs to be configured with the 14-byte prefix for the area, and it then informs the other routers in the area. Also, routers that have ports connecting to nodes in other areas need to be configured to know which ports are inter-area ports, so that areas do not unintentionally merge. In contrast, with IP, every router has to be configured with the unique address prefix on each of its ports.

Note that CLNP keeping track of all the individual nodes inside an area requires some traffic overhead, because each endnode needs

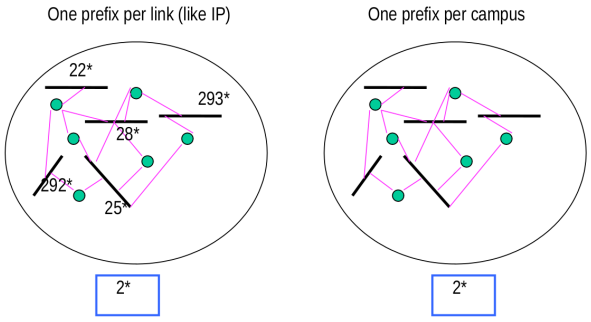

Figure 3: CLNP area routing versus IP

to announce its location. However, the alternative (IP relying on Ethernet to provide the functionality of a region with a flat address space) does not avoid that overhead. IP plus Ethernet requires ARP (address resolution protocol) to obtain the endnode's address on the final 'link', and Ethernet requires bridges to keep track of the learned location of endnodes, and to flood traffic to endnodes whose location is not yet learned.

2.6 CLNP or a to-be-designed protocol to be named IPv6?

Around 1992, people were concerned that IPv4 addresses were too small to support the explosive size predicted for the Internet. Some people suggested that IP should be replaced by CLNP. Unfortunately, because a few vocal people objected (there is unfortunate rivalry between standards organizations) the decision was to invent something new, to be called 'IPv6'. If instead, the decision would have been to adopt CLNP as the layer 3 packet format [3], the Internet would have been working with 20-byte addresses by 1993.

- At that point, CLNP was implemented by most of the major router vendors, and they all supported the decision to adopt the CLNP packet format for layer 3.
- With just a few weeks of work, someone created an implementation of TCP that worked over CLNP, so all Internet applications either immediately worked over CLNP, or could be made to work with very little effort. (Despite the wonderful theory of network layers, the API that TCP exposed to the layer above included the layer 3 address. Some applications used this information, and assumed it would be exactly 4 bytes long, so these applications would have to be slightly modified.)
- It would have been a very painless and quick conversion at that point in time. The Internet was still fairly small, and not the ubiquitous mission-critical infrastructure that society depends on today.
- DHCP had not yet been invented, so converting to CLNP enabled autoconfiguration of endnode addresses.
- As described in the previous section, CLNP's area routing concept allows movement of endnodes within an area, and autoconfiguration of routers within the area.

I have never heard any plausible technical reasons for not having adopted CLNP in 1992. I've heard things like:

- Replacing IP with CLNP would be "ripping the heart out of the Internet and putting in a foreign substance, whereas IPv6 will just be a gentle upgrade to a new version of IP". (Note: IPv6 is no more compatible with IPv4 than CLNP.)
- "We don't like ISO's session layer" (or management layer, or ...). Note that the proposal was replacing layer 3 alone, not changing all the IP application and management protocols. TCP and UDP (slightly modified as necessary because the address size changed) would run over CLNP.
- "We're not running out of IPv4 addresses yet, so let's wait and I'm sure we can invent something way more awesome than CLNP." I claim that CLNP has important technical advantages over IPv6, such as area routing, whereas IPv6, like IP4v, requires every link to have a unique block of addresses.

3 SURPRISING OUTCOMES

The rest of this article will discuss surprising results of some of the technologies and decisions covered in the first part of this article.

3.1 Why do we need both Ethernet and IP?

Originally bridging was needed to support unroutable protocols, and inexpensively support the plethora of layer 3 protocols (e.g., IPX, Appletalk, IPv4, etc.) deployed at the time. But today, neither of these reasons apply. The world has converged upon IPv4 (and sometimes IPv6), and all products have IP in their network stacks. Ethernet is no longer a multiaccess link. Ethernet today is just point-to-point links. So why do we need bridges and Ethernet? Why not interconnect everything with IP routers?

Indeed, if the world had adopted CLNP, there would be no reason for the Ethernet header, or bridging. Anything provided by bridged Ethernet was provided (and in a technically superior way) by CLNP area routing. Because IP has no concept like CLNP's area routing, IP depends on Ethernet to provide the equivalent of CLNP's area routing.

Another advantage of CLNP is that when IP gets to the final 'link', the final IP router needs to do an expensive protocol such as ARP (Address Resolution Protocol) in order to get the address of the destination on the final link. In contrast, in CLNP, once the final area is reached, the node's address within the area is part of the layer 3 address.

So, a surprising outcome of having deployed spanning tree bridging is that it enabled IP to provide (although less well than CLNP) the region-with-a-flat-address feature that CLNP area routing provides.

Note that IPv6 acts just like IPv4, where every link needs its own block of addresses, so IPv6 will also depend on some other technology to simulate CLNP area routing.

Recently, technologies have emerged (TRILL, VXLAN, NVGRE, GENEVE) to eliminate bridged Ethernet's suboptimal routing, and danger of forwarding without a hop count. Again, if the world were using CLNP, it is unlikely these technologies would have been invented.

Within a region similar to a CLNP area, these technologies use addresses internal to what we are referring to as an 'area' (i.e., what

IP perceives as a single link), and have the router injecting traffic into the area add an extra header, meaningful only within the area, and the data inside the area is routed according to that outer header. A disadvantage of this approach compared with CLNP area routing is that these encapsulation techniques require the router injecting traffic into the area to have some mechanism to find the internal address of the destination node.

However, an advantage of the extra encapsulation is that it allows other features, such as the ability to provide the equivalent of VLANs to make it easy to configure which nodes are allowed to talk to each other. Also, today most data centers have a central node that knows where all the VMs are, and can update the tables of the routers, which is more efficient than the CLNP approach of having endnodes announce themselves. It's likely that CLNP would have also adopted a concept of a central node configuring endnode location data into the routers for endnodes that were explicitly positioned by a central node (and perhaps kept the feature of endnodes advertising themselves, for endnodes such as laptops that were physically plugged into some place by a user).

3.2 DHCP

DHCP (Dynamic Host Configuration Protocol) is a very different mechanism for autoconfiguring addresses than CLNP's mechanism of 'insert your permanently configured 6-byte Ethernet address into your CLNP address'.

With DHCP, a server on the bridged Ethernet (or other region of the network perceived by IP as a single "link") has a block of IP addresses within the prefix for that link. An endnode with Ethernet ID 'X' makes a request "I am X. I need a layer 3 address" and the DHCP server gives X an IP address to use for some amount of time.

DHCP would likely never have been invented had the world adopted CLNP, because CLNP already had the ability to do autoconfiguration of endnodes. However, the DHCP approach has some attractive properties over the CLNP approach.

- Addresses can be much smaller, since they can be handed out densely and be specific to a single 'link' (unlike Ethernet addresses that need to be globally unique). Since there will likely never be more than a million or so endnodes within a 'link', 3 bytes for 'address within the link' would be more than sufficient.
- People have gotten nervous about the privacy issue of having a computer's permanently configured Ethernet address embedded into a layer 3 address. With DHCP, wherever your computer plugs into the Internet, it is given an address that allows your computer to be somewhat anonymous. The DHCP server on your link might know who you are, but a remote node that sees your traffic will not be able to identify you from your layer 3 address.

Something like DHCP could have been used with CLNP, or given that the 'node address within area' was 6 bytes long, CLNP nodes could have chosen a node address at random and there would be a very low probability that a node would be unlucky enough to choose the same address as another node in the area. But would people have bothered to invent something like DHCP if they already had autoconfiguration?

3.3 NATs

NATs (Network Address Translators) [4] are a somewhat frightening concept, and people love to hate them. They were invented because there weren't enough IPv4 addresses for every Internet-attached node to have an IPv4 address. A NAT has a pool of globally routable IP addresses and sits between an 'internal network' and the global Internet. The internal network has a block of 'private IP addresses' that can be used for communication within the internal network. But that same block of private IP addresses will be used in other internal networks, so a node's private IP address cannot be used outside its internal network.

That is where the NAT device comes to the rescue. If a node, say A, with an internal-only IP address wishes to communicate with a node B on the Internet (where B has a globally routable IP address), A's packet, with source=A and destination B, will go through the NAT. The NAT assigns to node A temporary use of a globally routable IP address G, replaces 'A' in the IP header with G, and remembers that when the NAT sees traffic between B and G, the NAT needs to replace G with A. To further preserve IPv4 addresses, NATs share globally routable addresses among many internal nodes, and distinguish the actual nodes based on TCP/UDP ports.

If the world had converted to CLNP, NATs would probably not have been invented (because with CLNP's 20-byte address, nobody would have been concerned with running out of addresses). However, now that NATs are deployed, people have come to like an important security feature that they provide; namely, that an internal node A with an internal-only IP address cannot be communicated with by any node outside the internal network unless A initiates the communication.

4 THE BIGGER PICTURE

Although it's interesting to examine some of the subtle features of specific protocols, the Internet is more than its protocols. It has changed society in many ways, some good and some bad. Some good things:

- It is extremely easy to stay in touch with friends and family, and meet new people.
- A new business, with minimal marketing and no storefront, can reach a global customer base. Likewise, customers can purchase products from all over the world.
- Information is easily and freely available. Wikipedia is a true marvel. There are free college-level courses available.
- Everyone with a cellphone camera becomes a reporter. It is not feasible to expect a major network's camera crew and reporters to be onsite when an unexpected newsworthy event occurs.

However, there are ways in which the Internet had changed society for the worse.

- It is trivial to create fake news, and instantly disseminate it throughout the world. If the only source of news was from credentialed major networks, cryptographic signatures on the data could verify its authenticity. However, since everyone is now a potential reporter, there is no way to verify the identity of the source.

- Society becomes more polarized. If the only source of news was a few major news organizations, people would be exposed to balanced coverage (assuming a reasonably free press and not state-run media). But with the ability to only focus on stories that you want to see, and with machine learning programs having a goal of keeping the viewer engaged for as long as possible, the stories someone sees will be more and more extreme, in line with what the machine learning algorithm has learned that the user wants to see.
- Scam artists have a global audience, and can reach them with very little fear of consequences, and very little effort. This is worth a whole section in this article.

4.1 Scams

The theory of how things should work on the Internet is so wonderful and simple.

- A service obtains a DNS name.
- The service creates a public key pair.
- The service obtains a certificate signed by a trusted authority that vouches for the fact that that site's public key owns that DNS name.
- When a client communicates with the service, it uses a protocol (e.g., TLS) that verifies that the service knows the private key associated with the public key in the certificate, and cryptographically protects the client-service conversation.

Unfortunately, the DNS name of a service is not obvious to a human. Humans find services through Internet searches, which yield obscure huge URL strings with DNS names embedded somewhere inside. But what is the right DNS name for the service?

I was recently taken in by a scam. I needed to renew my driver's license. I typed into the Internet search engine "renew Washington state driver's license". The top result had a URL with a DNS name that looked reasonable to me, though I was in a hurry and didn't look very carefully, since the top result is always the right one, right? (The DNS name was washington-licensing.org). The web site was well organized, and I typed in all the information I expected it to ask for (my name, address, license number, and credit card). Then it said "Here are a bunch of offers you are now qualified for", which made me realize I might not be at the real site. Examining the site again, I realized it wasn't promising me a driver's license, just "information for getting a license". Perhaps, if I bothered downloading the "free e-book about getting a license" that it said was one of the offers I was qualified for, it would tell me the real URL I should have gone to. (The scammers made lots of charges on my card, but my bank disallowed them all and changed my credit card number.)

I checked at the time and there were scam sites for all 50 states with DNS names like `Utah-licensing.org`. These sites appeared first in the search order because the Internet search sites allow people to pay to be first. But even if there was not the ability to pay to be first in the search order, scammers can figure out how to manipulate the search algorithm.

This sort of scam is so easy to deploy, and instantly reaches a huge potential target audience. I claim that the answer is not 'better user training'. There is no reason why a human should know that `Washington-licensing.org` is a scam site and `dol.wa.gov` is the correct DNS name.

5 SUMMARY

Some very interesting technology has been created to compensate for choices that seem suboptimal. If 'unroutable protocols' had not been deployed, spanning tree bridging might never have been created, but bridging, in addition to solving the ability to forward unroutable protocols, enabled faster Ethernets, cost-effective multiprotocol forwarding, and more usable IPv4.

If CLNP had been adopted in 1992, NATs might never have been invented. Although originally invented to solve the problem of IPv4 address scarcity, they also create a level of security because nodes with internal-only addresses cannot be addressed by outside nodes.

CLNP's autoconfiguration based on 'insert layer 2 address into layer 3 address' might have prevented deployment of server-assigned addresses with protocols such as DHCP. DHCP temporary assignment of addresses enhances privacy, and could allow smaller layer 3 addresses.

Although these technologies can be used as easily with CLNP as with IP, they might not have been deployed, because there would not have been as pressing a need for them with CLNP.

The final thought in the article is that we need to be aware of just how powerful a force for both good and bad the Internet is, and that instead of blaming users for things like not recognizing the difference between the correct DNS name and a scam DNS name, we need to think of better solutions.

REFERENCES

[1] Radia Perlman. 1985. An algorithm for distributed computation of a spanningtree in an extended lan. *ACM SIGCOMM Computer Communication Review* 15, 4 (1985), 44–53.

[2] Radia Perlman. 1999. *Interconnections: Bridges, Routers, Switches and Internetworking Protocols, 2/e.* Addison Wesley.

[3] D Piscitello. 1993. Use of ISO CLNP in TUBA environments. IETF RFC1561. (1993).

[4] Paul F Tsuchiya and Tony Eng. 1993. Extending the IP internet through address reuse. *ACM SIGCOMM Computer Communication Review* 23, 1 (1993), 16–33.