WHITE PAPER · ENTERPRISE NETWORK ARCHITECTURE

Before SD-WAN and SASE Had Names

Rebuilding a 500-site global network on the public Internet, 2007–2009

Between 2007 and 2009 we re-architected Cadbury's 500-site, 60-country enterprise WAN onto ordinary local Internet access, IPsec we owned, and ten carrier-neutral points of presence. The methodology was simple: stop buying a network, and start thinking like an ISP.

BY **JACK NASH** 500 SITES · 60 COUNTRIES ABOUT 23 MIN READ

THE ARGUMENT IN BRIEF

- Our financial data and our incident data pointed at the same layer: roughly **75–80% of WAN spend** and **98% of circuit failures** lived in the local access loop.
- The access circuit was also the **smallest link in the path** — which is where congestion management is worth doing, and where a carrier backbone SLA wasn't measuring.
- So we stopped buying a private WAN service and bought **locally appropriate Internet access** instead, treating the last mile as interchangeable.
- We moved the intelligence to **roughly ten PoPs in carrier-neutral interconnection facilities**, full-meshed over IPsec across a single global ISP backbone.
- The principle underneath all of it: **buy connectivity, own the intelligence, engineer around failure.**

PART I **The Observation**

Introduction

Between approximately 2007 and 2009, while at Cadbury, I was given responsibility for re-architecting a global corporate network supporting approximately 500 sites across 60 countries.

The geographic distribution mattered. Cadbury wasn't simply a large network — it was an unusually global one. Providing connectivity to hundreds of locations across dozens of telecommunications markets meant dealing with different carriers, different access technologies, different economics, different levels of infrastructure maturity, and very different provisioning environments.

The existing environment was typical of a large multinational enterprise of that era: a collection of mostly country-specific, multi-class MPLS networks connected through Cisco routers at hub sites. It worked. But it was expensive, complicated, carrier-dependent, and difficult to scale globally.

The architecture we ultimately developed was radically different.

We used locally appropriate public Internet connectivity as an interchangeable access layer. We secured the transport ourselves with IPsec. We established approximately ten global points of presence in strategically selected carrier-neutral interconnection facilities with access to major IXPs and dense ISP ecosystems. Those facilities became the controlled edge of our global enterprise network.

We interconnected those PoPs using a selected global ISP backbone, deliberately minimizing unnecessary inter-AS handoffs. We engineered failover between access circuits and between global PoPs. We wrapped the network end-to-end with application-aware WAN optimization. And we allowed remote users to connect directly into our geographically distributed edge.

Today, several of those ideas sound familiar. In 2007–2009, they did not. There was no commercial SD-WAN architecture guiding us, and the term SASE wouldn't exist for another decade. We arrived at these functions independently because we were solving the engineering problems in front of us.

And the idea that started the project came from an unlikely place.

The developing countries were teaching us something

Cadbury operated across a wide range of economies. Some of our businesses were in highly developed countries with substantial IT and telecommunications budgets. Others operated in developing countries where considerably less money was available for technology infrastructure.

I began noticing something interesting.

Our wealthier countries could afford conventional enterprise solutions. If something was important, they bought the premium service. If availability was important, they bought redundancy. Sometimes, in practical terms, they simply bought two of everything. It worked. But it was expensive.

Our developing-country operations didn't have that luxury. They still had businesses to run. They still needed reliable communications and access to corporate systems. They simply had much less money with which to accomplish it.

That constraint made them clever. Many relied upon ordinary local Internet service providers and found practical ways to meet business requirements without purchasing the expensive telecommunications

products considered standard in our wealthier markets.

That made me wonder: what would happen if we applied developing-country thinking to first-world infrastructure? What if we stopped asking what enterprise telecommunications product we were supposed to buy, and instead asked what we actually needed to engineer?

That question became the origin of the global network.

Follow the money

The observations from our developing countries gave me the idea. Our own financial data told me where to look.

When I examined the economics of our MPLS environment, our internal analysis showed that approximately 75 to 80 percent of our total WAN cost was associated with local access circuits. The majority of what we were spending wasn't necessarily buying the global backbone. It was repeatedly extending that network from the carrier's infrastructure to approximately 500 individual locations.

Country after country. Site after site. Local loop after local loop.

Then I looked at reliability. Our incident data showed that approximately 75 percent of our network outage tickets were associated with circuits or service providers. And when I examined those circuit outages, approximately 98 percent involved the local access circuit rather than the provider's core network.

The financial data and the incident data independently pointed toward the same architectural layer.

75–80%	~75%	~98%
of WAN spend was associated with local access	of outage tickets involved circuits or providers	of those circuit outages were local access, not the core

Figures are approximate and drawn from internal Cadbury cost and incident analysis of the period.

The most expensive portion of our traditional WAN architecture was also overwhelmingly where our circuit failures were occurring. That was difficult to ignore.

The last mile was changing

There was another reason the economics were beginning to look increasingly irrational.

Our traditional MPLS services commonly relied upon telecommunications access technologies such as T1/E1 and DS3/E3 circuits. These were traditional carrier services with traditional provisioning

processes. They could be expensive. They could take a long time to install. And increasing bandwidth often meant another significant step upward in cost.

Internet access was evolving differently. Local ISPs were increasingly delivering connectivity using DSL and other broadband technologies designed for a much larger market. Those services could often provide considerably greater bandwidth, could frequently be installed much faster, and cost dramatically less than the traditional enterprise access circuits we were purchasing.

So the local access layer had three characteristics. It consumed most of our WAN budget. It generated almost all of our circuit failures. And faster, cheaper, more readily available alternatives were emerging in local markets.

The obvious question became: why preserve it?

PART II **Rethinking the Purchase**

Make the last mile interchangeable

The traditional response would have been to optimize the existing architecture. Negotiate better MPLS pricing. Purchase stronger SLAs. Add redundant carrier circuits. Install redundant routers and firewalls. Perhaps buy two of everything.

But our data suggested something more fundamental. We didn't need a cheaper version of the same architecture. We needed a different architecture.

The developing countries had already demonstrated the raw material: ordinary local Internet connectivity worked. So instead of paying a global carrier to extend a premium private WAN service all the way to every corporate location, we could purchase locally appropriate Internet connectivity. The local ISP already had infrastructure in the market. It already reached businesses. It could frequently provide greater bandwidth for significantly less money. And it could often install that connectivity considerably faster.

The idea wasn't *replace MPLS with cheap Internet*. That would have been cost cutting.

> **The architectural idea was: make the access circuit interchangeable, and move the network intelligence somewhere else.**

Think like an ISP

Once I looked at the problem that way, the design challenge changed. Use ordinary Internet connectivity — then solve security ourselves. Solve performance ourselves. Solve routing ourselves.

Solve redundancy ourselves. Solve application prioritization ourselves. Solve global transport ourselves. And retain ownership of the architecture.

Eventually that led me to the question that defined the project:

> **If I were a global Internet service provider, how would I build this network?**

Question the assumptions behind MPLS

Moving a global enterprise away from traditional MPLS was not an immediately comfortable proposition. During one senior-level discussion, I was asked:

"What Fortune 500 company would ever risk its business to the public Internet?"

My response was essentially: *"You mean other than Microsoft, Facebook, Amazon, Google...?"*

The room went quiet for a moment. There were a few chuckles. But the point was serious. Some of the world's largest technology companies were already dependent upon Internet infrastructure. The Internet clearly could carry business-critical traffic. The real question was whether we could engineer an enterprise architecture that used it appropriately.

There was another important realization. The global telecommunications providers we were dealing with weren't necessarily operating completely separate physical backbones for every service they sold. Their Internet services and premium enterprise services could share substantial portions of the same underlying provider infrastructure. We were paying a premium for how the service was packaged, managed, prioritized, and supported — not necessarily because every packet traversed an entirely different physical network.

That caused me to separate two things enterprise telecommunications traditionally bundled together: connectivity and network architecture.

What if we purchased connectivity from service providers, but took responsibility for the architecture ourselves?

PART III **Where Congestion Actually Matters**

Understand where congestion actually matters

Performance was one of the first objections. The conventional enterprise answer was multi-class MPLS. Voice received one Class of Service. Video received another. Business-critical applications received another. Everything else received best effort.

But understanding what queuing actually accomplished made me question what we were purchasing. Using Cisco's Class-Based Weighted Fair Queuing (CBWFQ) as an example, configured bandwidth guarantees become relevant during congestion. When an egress interface has sufficient capacity to transmit the traffic presented to it, there is no constrained bandwidth for the queuing system to arbitrate among competing classes. When congestion occurs, the router needs rules determining how that constrained capacity should be allocated.

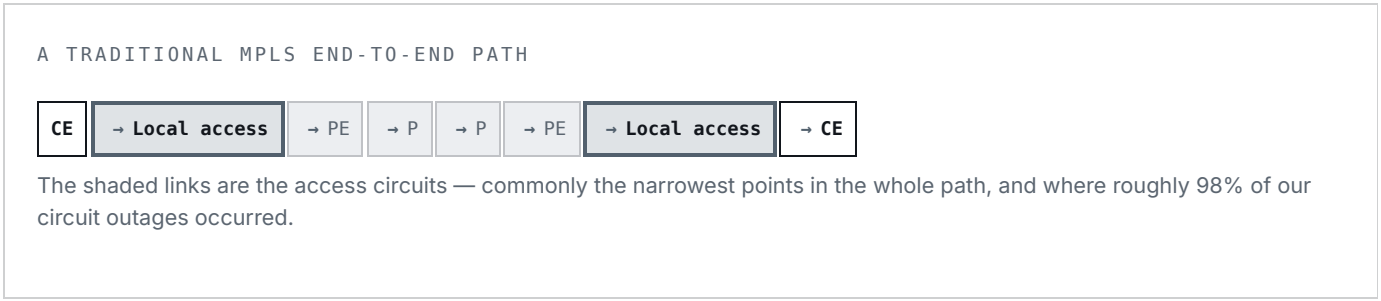
That distinction was critical:

> **QoS doesn't create bandwidth. It determines how constrained bandwidth is allocated.**

And that raised another question. Where was congestion most likely to occur?

The backbone wasn't usually the smallest link

Consider a traditional MPLS path:



The provider backbone generally operated at substantially greater capacity than an individual customer's access circuit. The local CE-to-PE access circuit was commonly one of the smallest links in the end-to-end path.

That wasn't universally true. But in our environment, it was frequently the case. A provider might operate an enormous backbone while one of our sites reached it through a relatively small T1/E1, DS3/E3, or similar circuit.

If congestion management was going to provide significant value somewhere in the path, the constrained access link was an obvious candidate. But congestion management has to occur at the right place.

Manage congestion before the bottleneck

Congestion management is most useful before traffic traverses the constrained resource.

For traffic leaving one of our sites, that was straightforward. If our CE was about to transmit more traffic than the local access circuit could carry, we could classify, queue, and prioritize that traffic before placing it onto the circuit. That made sense.

Inbound traffic presented a different problem. Once the provider had transmitted traffic across the constrained local access circuit toward our CE, the scarce bandwidth had already been consumed. Dropping a low-priority packet after it arrived at our CE didn't give us that bandwidth back. The packet had already occupied the circuit. The congestion-management decision had been made too late.

This distinction became important to how I thought about network performance. The best place to manage congestion is before the constrained link. And when economics make it practical:

> **The best congestion-management strategy is to eliminate the constraint.**

That was another advantage of broadband Internet access. Instead of paying substantial premiums to carefully arbitrate a relatively scarce T1/E1 or DS3/E3 access resource, we could often purchase considerably more local bandwidth for considerably less money.

The CoS paradox

Now consider the provider backbone.

Suppose a global service provider routinely operates its backbone with enough congestion that sophisticated queuing is necessary to protect my applications. Then I have a bigger problem: I'm buying connectivity from the wrong provider.

A properly engineered global backbone should maintain sufficient capacity and headroom. At the time, our experience and understanding of major carrier engineering practices was that backbone capacity was generally groomed well before sustained utilization reached saturation — often around the 70 percent range. I didn't want to purchase an inadequately provisioned backbone and then pay the same provider an additional premium so my packets received preferential treatment when its infrastructure became congested.

Now consider the opposite situation. The provider operates a properly engineered backbone. There is sufficient capacity. Its interfaces aren't congested. There is no constrained resource requiring CBWFQ to choose between my application classes.

That produced what I came to think of as the paradox of multi-class MPLS:

THE PARADOX OF MULTI-CLASS MPLS

If the provider's network is congested, don't buy the network.

If the provider's network isn't congested, don't buy multi-class MPLS.

This was never an argument against QoS. QoS is an excellent engineering tool when a constrained resource genuinely needs to be managed. It was an argument for putting congestion management where the constraint actually existed, rather than paying a global carrier a premium to provide multiple traffic classes throughout a backbone that should not routinely be congested.

PART IV **What the SLA Actually Covered**

Then there was the SLA

Service Level Agreements deserved the same examination.

The theory behind an SLA is reasonable. A provider commits contractually to a particular result. Failure to achieve that result creates a financial consequence. That consequence is intended to influence behavior: fix my problem rather than pay the penalty.

SLAs can have legitimate organizational value. They establish measurable contractual expectations. They create financial remedies. They can also provide internal reassurance that management has contractually protected the organization.

But I began questioning whether purchasing stronger SLAs actually changed the operational result.

Who really pays the SLA?

In the large multinational telecommunications agreements I was dealing with, SLA exposure wasn't simply free financial risk absorbed by the carrier. It was another financial variable in the commercial model. The provider understood its historical performance. It understood the commitments being negotiated. It understood its potential financial exposure. That risk could be modeled and incorporated into the economics of the overall customer agreement.

The customer could therefore end up funding at least part of the SLA risk it believed it had transferred to the provider.

There was another disconnect. The operations personnel restoring our service were generally driven by their internal operational metrics, escalation procedures, and restoration objectives — not by detailed

knowledge of the financial provisions in our master telecommunications agreement. The SLA existed at the commercial level. Restoration happened at the operational level.

That made me question how much additional operational performance we were actually purchasing.

Does the SLA cover the failure domain?

There was another issue. Depending upon the carrier, country, service, and contract, the strongest SLA commitments could apply primarily to the service provider's own network rather than every component of the local access path.

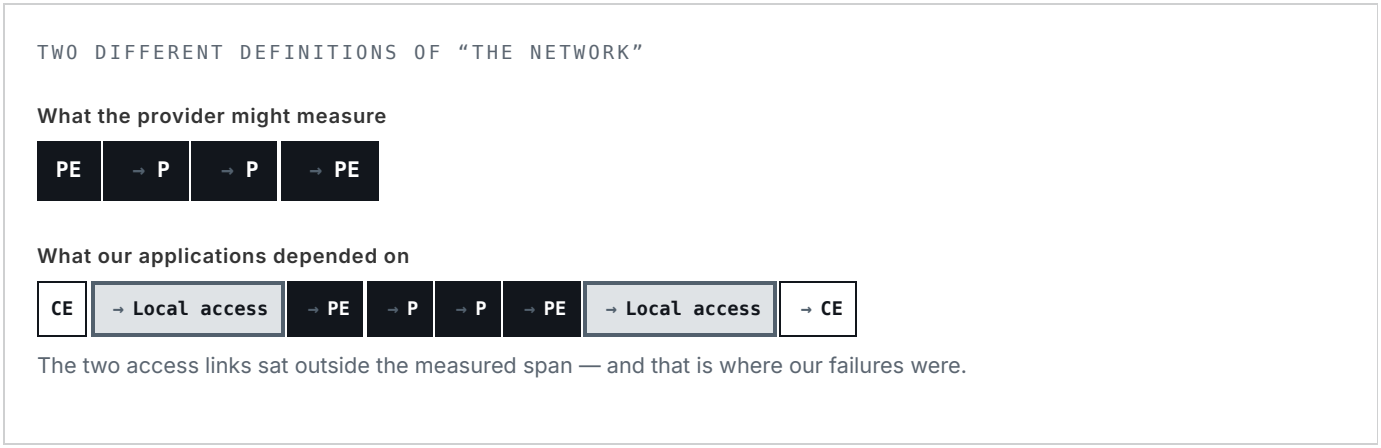
That mattered because our operational data showed something very different about where failures occurred. Approximately 98 percent of our circuit outages involved local access. The provider's core wasn't our primary reliability problem. The edge was.

This created an important question: does the SLA actually protect the portion of the service that is failing?

A contractual guarantee covering a highly reliable provider backbone has limited operational value if the overwhelming majority of the customer's outages occur somewhere else. And even when an access failure qualified for an SLA credit, the credit didn't restore service. It arrived after the failure. The business needed connectivity during the failure.

What exactly is the SLA measuring?

The measurement boundary created another problem. Depending upon the provider and service, SLA performance could be measured between provider-controlled points in the network rather than across the customer's complete end-to-end path.



Those are not equivalent measurements. A carrier could legitimately report excellent latency, packet-loss, or availability performance across its backbone while the customer simultaneously experienced

poor end-to-end service because of an access problem. And our own data told us that access problems dominated our circuit failures.

That meant an excellent provider-core SLA result did not necessarily describe the network experience our applications or users were receiving. The SLA could be measuring the provider's definition of its network. I needed to measure the business's definition of the network.

The question that mattered wasn't *"did the provider meet its SLA?"* It was *"can our applications reliably communicate from one endpoint to the other?"*

Engineer around failure instead

All of this changed how I thought about SLAs. An SLA wasn't inherently useless. But I no longer considered it a substitute for resilient architecture.

Rather than spend additional money primarily on contractual consequences after a provider failed, I preferred to invest in making the provider's failure less important. If a circuit failed, I didn't primarily want a service credit. I wanted the network to stop using the failed circuit. If a path failed, I wanted another path.

> **An SLA is a contractual response to failure. Resiliency is an engineering response to failure.**
>
> **I preferred the engineering.**

PART V **The Architecture**

Build where the Internet meets

Simply buying Internet connections at 500 sites wasn't enough.

The Internet is a network of networks. Traffic can cross multiple autonomous systems between source and destination. Every inter-provider transition introduces another routing policy, capacity boundary, commercial relationship, and potential performance variable. I came to view those inter-provider boundaries as a more significant and less controllable performance risk than the backbone of a properly engineered global ISP.

So I continued thinking like an ISP. Where do the networks themselves meet?

Internet providers exchange traffic at Internet Exchanges and through private interconnection at carrier-neutral facilities. Those facilities are effectively crossroads of the Internet.

Instead of selecting our global network locations primarily because Cadbury happened to have a large office or corporate data center there, I looked at the Internet's geography. We selected regional interconnection locations according to our business footprint. How many countries did we operate within the region? Which carrier-neutral facilities and associated IXPs provided access to the greatest number of local ISPs we needed? Which locations sat at the strongest intersection between Cadbury's geography and the Internet's geography?

Those became our strategic global PoPs.

Take the network to the carriers

At a major carrier-neutral interconnection facility, numerous service providers could be available within the same ecosystem. A new provider could effectively be a cross-connect away.

That changed provisioning. Traditional enterprise access circuits could take months to install; connectivity at an interconnection facility could potentially be established in days.

It also changed the economics. Instead of repeatedly paying to extend carrier infrastructure to locations we controlled, we placed our global infrastructure where the carriers already interconnected.

> **We stopped bringing carriers to our network. We took our network to the carriers.**

Build one global core

We ultimately established approximately ten strategically located global PoPs in these interconnection environments. We selected a global ISP capable of carrying the inter-PoP core within its own ASN, minimizing unnecessary inter-AS handoffs across the international path.

The design principle was straightforward: get international traffic onto the selected global backbone quickly, keep it there for as much of the journey as practical, and get it off close to the destination.

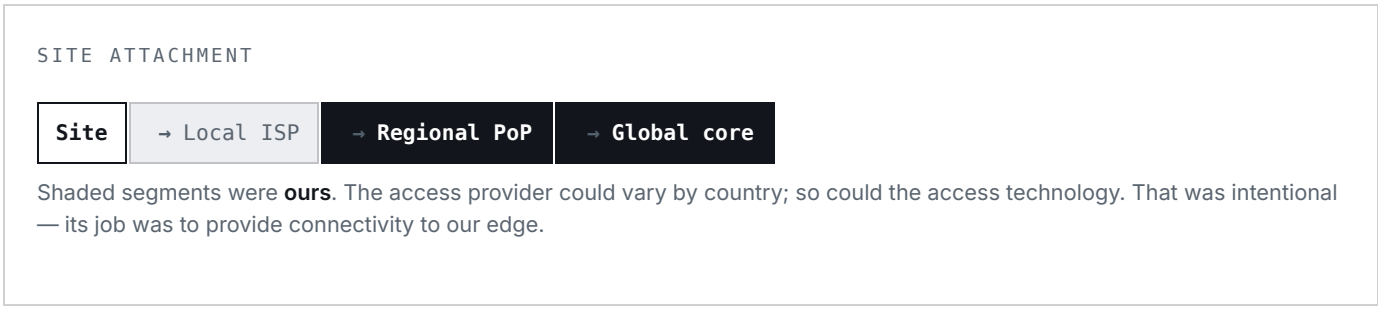
The approximately ten global PoPs formed a full-mesh IPsec topology. At that scale, a full mesh was manageable — ten locations require 45 unique point-to-point relationships. That gave each core location a direct encrypted relationship with every other global PoP, without requiring a worldwide full mesh between hundreds of corporate sites.

The PoP became the enterprise edge

These facilities were more than convenient places to interconnect networks. They became the controlled edge of our global enterprise network.

The local Internet provider wasn't the enterprise network. It was the access mechanism used to reach it. A corporate site could use an appropriate local ISP to reach its regional PoP. A traveling employee

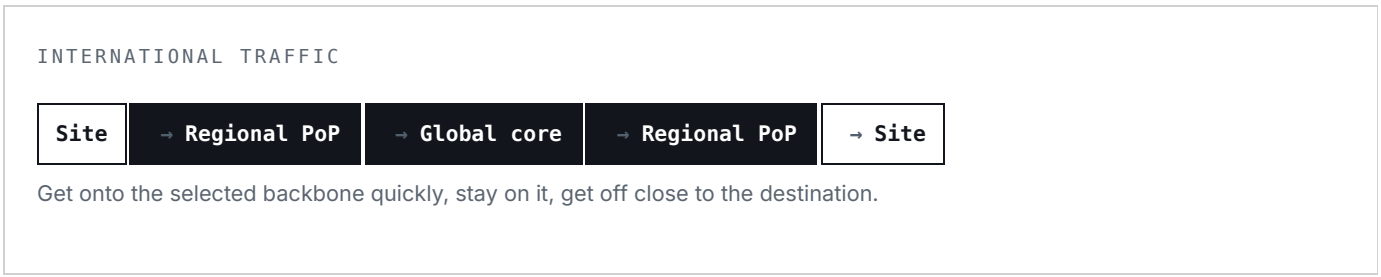
could use whatever Internet connectivity was available to establish a secure connection directly to an appropriate PoP. At those locations, we controlled access and the network edge.



Local traffic local, global traffic global

Below the global core, the architecture was deliberately hierarchical. Where corporate sites shared the same in-country provider, site-to-site traffic could remain within that provider's ASN rather than unnecessarily entering the global core.

International traffic followed a different path. A site communicating internationally moved toward its regional PoP, traversed the selected global backbone to another PoP closer to the destination, and then continued toward the destination site.



The philosophy was simple: local traffic stayed local where practical, and global traffic used the global core.

Contain the IPsec complexity

A worldwide full mesh between approximately 500 corporate locations would have been operationally unreasonable. So we didn't build one. We made the topology hierarchical instead.

The approximately ten global PoPs maintained the full-mesh core. Within an individual country, topology depended upon local requirements. Some countries could use manageable site-to-site meshes. Others primarily connected toward the regional PoP, with selected direct site-to-site tunnels where useful.

A site didn't need a tunnel to every other location in the world. It needed appropriate local connectivity and resilient access to the global core.

> **The objective wasn’t to eliminate complexity. It was to contain complexity where it could be managed.**

PART VI **Engineering for Failure**

Build resiliency at the right layers

Resiliency existed at multiple layers. Where appropriate, sites had multiple Internet circuits; if a preferred circuit failed, traffic could fail over to alternate connectivity.

We also engineered failover between global PoPs. A site's preferred path into the global network could use one PoP while an alternate path provided access through another. That addressed different failure domains: circuit failover protected local access, and PoP failover protected access to the global core.

> **The circuit was not the network. The ISP was not the network. The PoP was not the network.**

Each was simply a component or path into a larger architecture.

Engineer for the failures you actually have

The same philosophy caused me to question another standard enterprise practice: deploying high-availability routers and firewalls at every site. Two routers. Two firewalls. Redundant everything. It sounded safe.

But our incident data told us where the failures actually occurred. Approximately 75 percent of our network outage tickets involved circuits or service providers, and approximately 98 percent of those circuit outages involved local access.

A second firewall didn't restore a failed circuit. A second router didn't repair a carrier outage. We were spending capital protecting against a failure mode that represented a minority of our incidents, while the dominant failure mode existed outside the equipment rack.

So we changed the availability strategy.

Availability is a requirement, not a product

Instead of automatically purchasing redundant edge hardware for every location, we examined the site's actual availability and recovery requirements. How long could the location operate without a particular device? How quickly did we need to restore it? Could strategically positioned spare equipment satisfy that requirement, rather than duplicating production equipment at every site?

In many cases, the answer was yes. So we developed in-country hardware sparing strategies. Spare routers, firewalls, and other appropriate equipment could support multiple locations according to geography and required recovery time.

Where a business requirement genuinely justified immediate hardware failover, we could still deploy local HA. But we stopped treating *two of everything* as the automatic definition of availability. Sometimes immediate failover is required. Sometimes four-hour hardware replacement is sufficient. Sometimes next-business-day recovery is acceptable.

> **Availability is the requirement. Redundancy is one possible solution.**

Build our own application-aware CoS

Eliminating carrier-provided multi-class MPLS didn't eliminate our requirement for application performance. It moved responsibility for it.

Each corporate site had a WAN optimization appliance positioned behind its edge firewall. That effectively wrapped the network with an end-to-end application-aware performance layer.

Instead of purchasing generic traffic classes from the carrier, our WAN optimization infrastructure could recognize and prioritize actual applications. We could make performance decisions at Layer 7 according to our business requirements. This also put the intelligence on our side of the constrained access link, where we could influence traffic before transmission rather than attempting to manage congestion after scarce bandwidth had already been consumed.

In effect, we created our own application-aware traffic-management strategy. And it belonged to us. Changing ISPs didn't remove the policy. Changing circuits didn't remove the optimization. Different countries could use different access providers without requiring us to redesign application performance around each carrier's product portfolio.

Security without trusting the transport

Moving corporate traffic onto Internet infrastructure forced us to address security. But I came to believe the conventional question was wrong. Instead of asking *“can we trust the Internet?”*, I asked: *“why should we trust the transport at all?”*

MPLS provided traffic separation. But separation and security are not synonymous. A private carrier network was still infrastructure operated by another organization, on equipment we didn't control, through facilities we didn't control, and by people we didn't control.

So our architecture began with a simpler assumption: the underlying transport is untrusted.

IPsec allowed us to secure corporate communications independently of the ISP carrying them. The local carrier could change. The access technology could change. Different countries could use completely different providers. Our security model remained ours.

The user didn’t need an office

The distributed PoP architecture also solved remote access.

Traveling employees needed secure access to corporate resources. There was no reason to force those users to VPN back to a distant corporate office merely because an office traditionally represented the

enterprise network perimeter. They could connect directly to an appropriate regional PoP using their laptop VPN client.

Hotel Wi-Fi, home broadband, or another Internet connection didn't need to be trusted. It simply needed to provide IP connectivity to our edge. The user's physical location no longer needed to correspond with a corporate network location.

> **The user could come to the network.**

PART VII **In Retrospect**

What the industry would later call it

It is easy to look backward at this architecture using today's terminology. But this was 2007–2009. There was no commercial SD-WAN product category guiding the design, and the term SASE had not been coined. We arrived at these functions independently because each solved the next engineering problem.

Viewed through today's terminology, the parallels are difficult to miss:

- public Internet replacing carrier-dependent private WAN access
- interchangeable access transports
- multiple WAN paths with failover
- failover between geographically distributed PoPs
- encrypted overlays
- hierarchical site and country connectivity
- a full-mesh global core
- geographically distributed network PoPs
- controlled access and security at those PoPs
- application-aware Layer 7 traffic management
- network intelligence independent of the access carrier
- remote users connecting directly into the distributed edge

These are functions and architectural principles that would later become associated with SD-WAN and SASE. But we weren't implementing somebody else's blueprint. There wasn't one.

I don't claim to have invented the terms SD-WAN or SASE, nor do I claim our architecture was identical to everything those terms encompass today. The claim is narrower: in 2007–2009, we independently developed many of the networking and secure-edge functions that would later appear in SD-WAN and SASE architectures.

We weren't trying to predict what the networking industry would eventually call them. We were trying to solve our problems. The architecture led us there.

Then Cadbury became Kraft

While this work was underway, Kraft Foods acquired Cadbury.

The two companies had very different infrastructure profiles. Cadbury had approximately 500 sites distributed across roughly 60 countries. Kraft had approximately 2,000 sites, predominantly concentrated in the United States and across a comparatively small number of countries. The combined environment therefore wasn't simply larger — it brought together approximately 2,500 locations with dramatically different geographic and telecommunications profiles.

During the integration, I was at Kraft's headquarters in Chicago when Cadbury's CIO asked me to join him in the Kraft CIO's office. He wanted me to explain what we had been doing.

So I walked through the architecture and the reasoning behind it. Why I questioned multi-class MPLS. Why I questioned the value of purchasing stronger carrier SLAs. Why our financial and incident data pointed toward the local access layer. Why SLA measurement boundaries didn't necessarily represent our end-to-end application experience. Why we used local Internet providers as interchangeable access. Why we built global PoPs in major interconnection ecosystems. Why international traffic remained on a selected global backbone. Why we controlled access and the network edge ourselves. Why we secured the transport ourselves. Why we engineered failover between circuits and global PoPs. Why application performance belonged in an end-to-end Layer 7 architecture we controlled. And why our actual outage data caused us to rethink blanket hardware redundancy.

Ultimately, I explained why I believed the enterprise should own the intelligence of its network while treating telecommunications providers primarily as suppliers of connectivity.

When I finished, the Kraft CIO turned to my counterpart, tapped him on the shoulder, and said:

> **“Hey, why don’t we think like that?”**
>
> — Kraft Foods CIO, Chicago

The next day, my counterpart began reporting to me, and I was appointed Global Director of Infrastructure for the combined companies.

I have always remembered that meeting. Not simply because of what happened afterward, but because it validated something larger than a particular network design.

We had stopped asking *“what global network product should we buy?”* and started asking *“if this were truly our network, how would we build it?”*

The larger lesson

The most important lesson from the project wasn't MPLS versus Internet. It wasn't IPsec. It wasn't CBWFQ. It wasn't WAN optimization, BGP, IXPs, or any individual technology. Those were tools.

The larger lesson was understanding those technologies deeply enough to distinguish genuine engineering requirements from the commercial products built around them.

The idea that started the project didn't come from the countries with the largest technology budgets. It came from observing those with the smallest. Constraint had forced them to become resourceful. They demonstrated that ordinary Internet connectivity could be the raw material.

Then our own data told us where to concentrate. The financial data told us where the money was going. The incident data told us where the failures were occurring. The network architecture told us where congestion could actually be managed. The SLA boundaries told us what the carrier was actually measuring.

All of those things pointed toward the same conclusion:

> **Stop optimizing the parts of the network that weren’t causing our problems, and redesign the parts that were.**

So we made the last mile interchangeable. We placed our global infrastructure where networks interconnected. We selected local ISPs according to local requirements. We minimized unnecessary inter-AS boundaries. We secured traffic without trusting the transport. We controlled access at a geographically distributed edge. We engineered failover at both the circuit and global PoP layers. We managed application performance ourselves. We kept local traffic local where practical. We placed international traffic onto a controlled global core. We allowed traveling users to enter that network through geographically distributed points of presence. We spent availability dollars against the failures that were actually occurring. And we retained ownership of the intelligence that made all of those components work together.

The result was an Internet-based private enterprise network supporting approximately 500 sites across 60 countries, designed and implemented in 2007–2009 — years before the industry would package many similar functions into the technologies and terminology familiar today.

It began with a simple observation: sometimes the operation with the smallest technology budget has the most to teach you about architecture.

It became an engineering philosophy:

Buy connectivity. Own the intelligence. Engineer around failure.

We stopped buying the network as a carrier product. We bought connectivity and built the network ourselves.

A note on the account. This is a first-person recollection of work carried out between approximately 2007 and 2009. Site counts, percentages, and cost figures are approximate and drawn from internal analysis of the period. Product and category names are used descriptively and belong to their respective owners.