Trading on the Benefits of Merchant Silicon

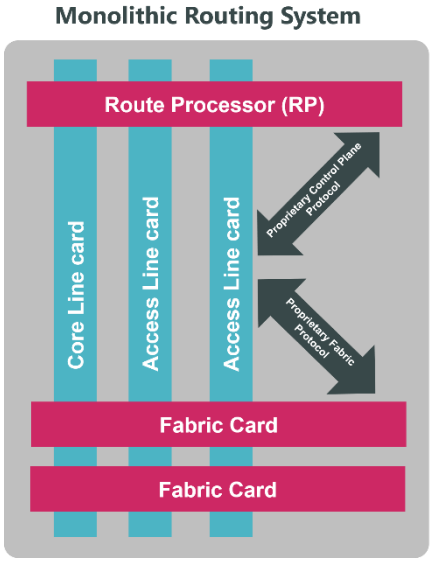
Why carriers are disaggregating their networks

It is the way it is

Network systems have remained largely unchanged since carriers first started delivering IP services. Once it became clear that router architectures constrained to discrete 'boxes' weren't powerful enough to build telco networks, every networking vendor's architecture evolved with the same basic characteristics.

This was based around a chassis that housed fabric cards and line cards. Core line cards faced into the network and were relatively dumb but fast, and access line cards faced towards subscribers and had smarter chips that could perform the service processing. Fabric cards provided interconnectivity between all the other components in the chassis.

Of course, you couldn't mix and match cards from different vendors inside the chassis and the control plane was also proprietary – but that's necessary to give the illusion that it is one single system and not just a collection of discrete components sharing a power supply.



Traditional chassis system

What's so bad about that?

Well, nothing – especially if you're an equipment vendor!

But for carriers, it's resulted in several harsh realities:

- They find themselves in a perpetual hardware replacement cycle, as each element of the system is upgraded in turn, without the ability to break-out of the system investment that they are locked into
- They can't mix and match the best hardware with best software – equipment selection is always a compromise between the two
- Multiservice systems have to provide ALL the service features that might be required by ANY service, whether they are being used or not. This is fundamentally bad economics as well as being a test nightmare

In short, monolithic chassis-based systems have become expensive, complex and lock the carrier into a vendor.

So, what's changed?

In a word (well, two words) - *merchant silicon*.

For many years the only way to scale was to use custom silicon and optimise the software around it. But silicon vendors now have the equivalent capabilities on their high-volume, low-cost networking chips as the traditional system vendors use on the line cards in their systems. Even the system vendors are starting to use the same merchant silicon chipsets, recognising that they simply can't compete on economies of scale.

And most importantly this merchant silicon is being used to build a new category of powerful low-cost 'bare-metal' switches, often constructed on the same outsourced assembly lines that manufacture the traditional router systems.

That's all very well for the hardware, but what about the software?

Well that's where companies like RtBrick come in. RtBrick produces carrier-grade routing software that turns these bare-metal switches into fully functional IP/MPLS carrier routers. The software runs in a container on a Linux operating system on the switch. And a 'cloud-native' approach to management systems allows these switches to be deployed with zero touch provisioning and run at the same low cost-points as cloud infrastructure.

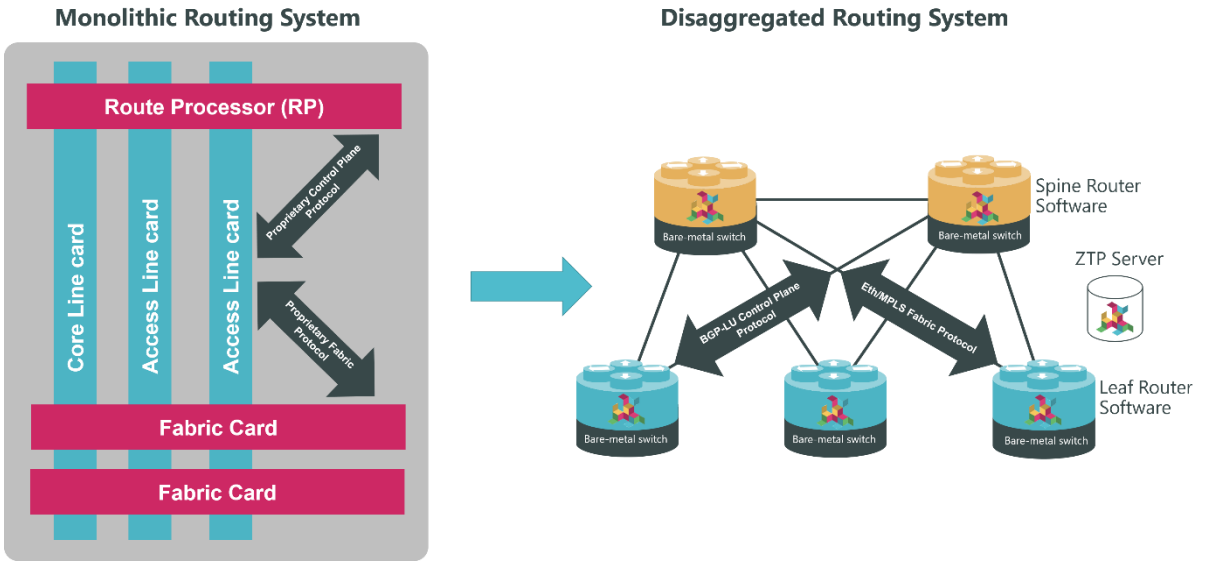
Separating out the hardware and software like this is referred to as *disaggregating* the network.

But how does it scale?

If the chassis-based router first evolved to deliver carrier-scale beyond single box systems, surely we have the same problem with disaggregated bare-metal switches?

Not at all. That's where modern high-performance silicon comes in. In effect we are replicating the chassis-based architecture, but in an open and flexible way, rather than a proprietary one. We replace the hardware cards with single RU systems which are optimized for each function.

The service-related functions (previously the access line cards) have their own 'leaf' routers, performing subscriber termination, queuing, metering, lawful-intercept and so on. And the fabric cards are replaced with a mesh of 'spine' routers, optimized for 400G connections and the support of full Internet routing tables.



Moving from a monolithic to disaggregated architecture

Unlike the closed chassis-based systems, the proprietary protocols that run between data and control panes are replaced by protocols such as Ethernet with IP/MPLS data encapsulation and BGP. These are both 'battle-hardened' and open, which means any element of the new system can be provided by any vendor. So, now we can mix and match both hardware and software vendors and upgrade the capacity of any dimension of the system without throwing away the existing infrastructure.

Hang on, can't we do this in the cloud or on x86 servers instead?

X86 servers and cloud infrastructure also come with network cards based on this same merchant silicon. And if we're using x86 servers to build out capacity for NFV, for example, why can't we just use the same infrastructure for our network services?

Well, there are network topology and latency reasons why we might not want to send massive amounts of traffic in and out of centralised data centers, but that aside there is no reason we can't run the routing software on x86 servers. In fact, at RtBrick, we test all of our routing software features in an x86 environment.

It's actually a very good approach for small numbers of subscribers or for niche services with complex forwarding chains such as firewall functions (where you might run out of chip resources on a bare metal switch). But it doesn't scale as well for high volumes of standardized services, terminating millions of subscribers. In those environments, x86 servers can cost significantly more per port, especially for high speed ports at 100G and above.

	BNG Chassis	x86 Server with NICs	Bare-metal switch
			
Cost* per 100G port	~$15,000	~$2,500	~$350
Annual energy per Gbps	~4.4kWh	~65.7kWh	~3.1kWh

*Estimated 'street price' for large deals
Numbers include share of chassis/spines

Alternative hardware approaches

But more problematic is the amount of power consumed. Every processing action has to be translated into x86 instructions – which is not an energy efficient way of forwarding packets. In fact, it uses about twenty times more power than a bare-metal switch. For a 48x10G switch this adds up to ~$35,000 of energy costs over a five-year period (compared to about $1,500 for a bare metal switch).

So, what do we win by disaggregating our network?

One thing is clear. The days of the monolithic chassis-based systems are numbered.

Taking advantage of merchant silicon in bare-metal switches, and disaggregating the hardware from the software, brings some huge benefits to carriers:

- Lower cost hardware – per port costs are a fraction of traditional vendors' systems
- We can swap and change software without throwing away or even upgrading the hardware
- We break the hardwiring between platforms and services, enabling easier re-use of common hardware across any service (a disaggregated architecture can also act as a software-based service cross-connect to route customer connections to the appropriate infrastructure)
- A bigger choice of software will make it faster to deliver new features and services
- Open standards and Web2.0 tools make the whole system easier to automate and to operate

We can use the disaggregated network systems to replace many functions within the network, from PE routers to Broadband Network Gateways.

And along with a cloud-IT toolset to operate the networks, disaggregated systems are a fundamental part of the way telecoms operators are set to transform their operations to match the agility, simplicity and cost-levels of cloud-IT infrastructure.

If you want to learn more about how some of the world's largest telecoms operators are evolving to software-based networks running on merchant silicon, why don't you get in touch with RtBrick?

rtbrick