NGCO Vision

Daniel Bernier

May 2022

Food for thought

While the visionary ten operators who launched NFV back in 2012 didn't envision it in these terms, what they were doing was applying virtualization principles of the time to problems of the time." (http://blog.cimicorp.com/?p=2893)

… Let's try not to do the same mistakes again

Bell

Landscape Transformation

The Processor vs ASIC Dilemma

What is best suited ?

ASICs at the edge

We know exactly what it needs to do

We design/buy a chip that does it
… building or acquiring generally not fast or cheap

We hope we do not have any new bright ideas afterwards

x86 at the edge

We can do whatever we want

We do it with code, not custom circuits

We can develop new ideas quite fast

… but we can't do it **efficiently**

Ref: Ian Wells, Cisco Systems, OPI project kick-off

Bell

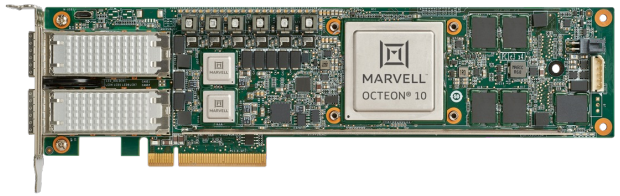
Comparing Apples with Pineapples

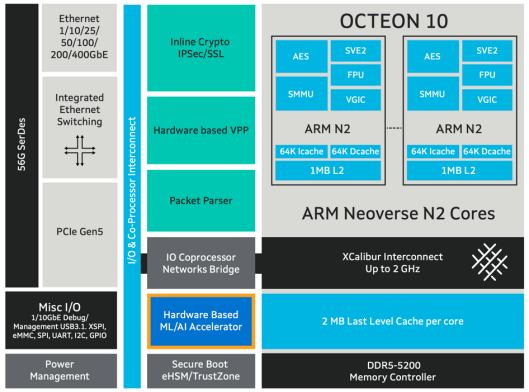
2nd Generation DNX (our current switches)

- Jericho+ 900Gbps Throughput
- 175W
- Closed SDK
- Fixed resource limits
  - Bandwith, SerDes, Max Ports
  - L2/L3/Tunnels/MPLS/ACLs
- Coupled with Celeron processor

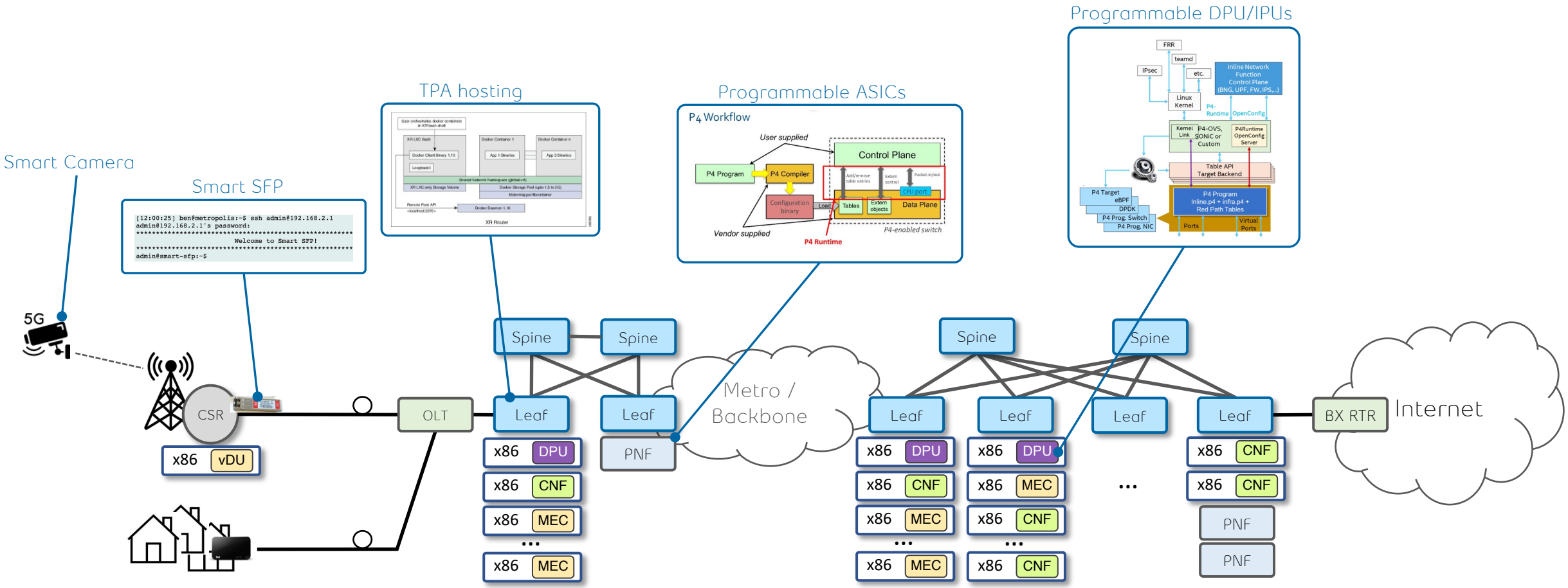
Marvell Octeon 10

- 1Tbps integrate switch Throughput
- 20W
- P4, VPP offload, Velox-SDK
- Integrated AI/ML + Crypto
- ARM Neoverse 2 (64C)





The Changing Landscape of Processing



TPA = Third Party Application Hosting

The Great Hyperscale Migration

AWS Announces general availability of the first AWS Wavelength Zone in Canada

Posted On: Apr 26, 2022

Today, we are announcing the general availability of AWS Wavelength on the Bell 5G network in Toronto. Enterprises, application developers, and Independent Software Vendors (ISVs), can now use the AWS Wavelength Zone in Toronto to build ultra-low latency applications for mobile devices and end-users in Canada.

Google Canada Blog Latest stories Product updates Company news

GOOGLE CLOUD

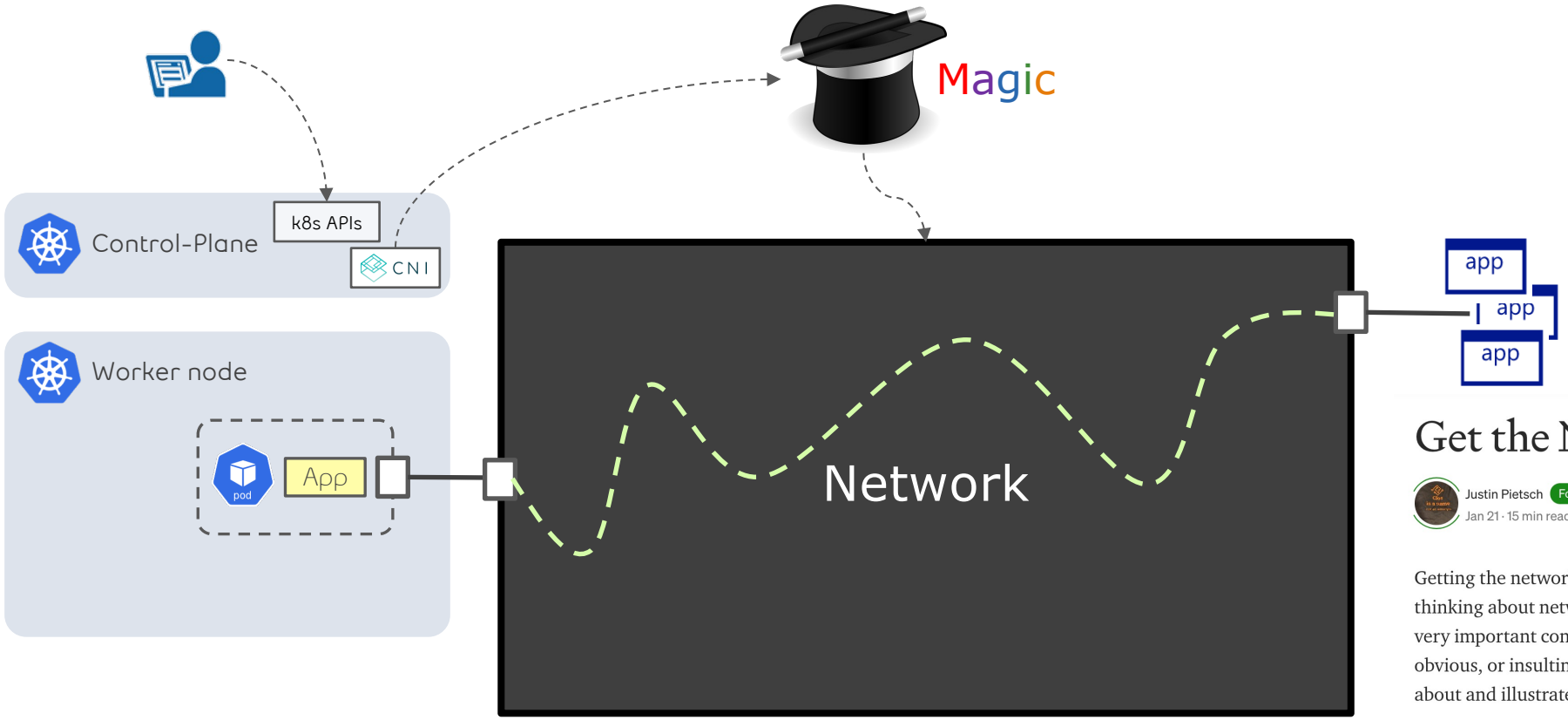
Bell Partners with Google Cloud to Deliver Next-generation Network Experiences for Canadians

Bell

Tenet #1
Protocol Simplification

The road towards Application Aware Infrastructure

Because in the end, **this** is what we want



https://datatracker.ietf.org/wg/apn/about/
https://datatracker.ietf.org/rg/panrg/about/
https://conferences.sigcomm.org/sigcomm/2020/workshop-nai.html

Get the Network Out of the Way

Justin Pietsch Follow
Jan 21 · 15 min read

Getting the network out of the way has been very important for me in my thinking about networks, and is an easy way to help talk about a bunch of very important concepts, decisions, and arguments. This can sound trite, obvious, or insulting. In this post I'll try to describe the concepts I'm talking about and illustrate with some of most important examples in my career.

The more that the network is noticed the worse things are for everyone. Often times, especially when the network is noticed, networking and network engineers are thought of negatively. Instead, if you think of it as a challenge it can help you focus on making a great network. You can think about your goals: how important it is to keep the network working well, to not disrupt the business, and to be able to keep up with any changes that the business needs.

SRv6 Introduction

- **Source Routing Paradigm** where path is encoded at source within an outer IPv6 header or via an extension header (RFC8754)

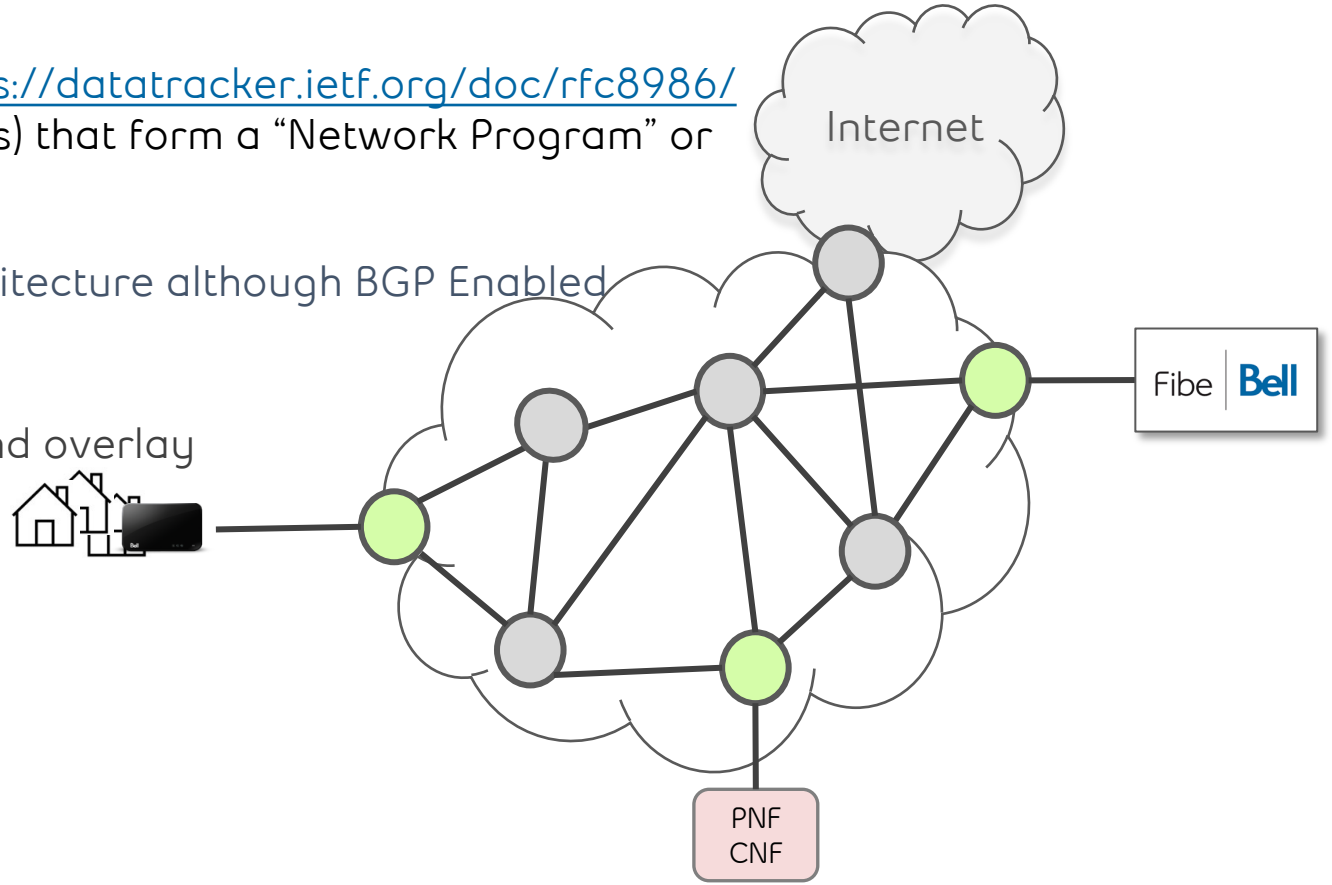
- **Path Segments** are represented within an IPv6 address format
  - in 128-bit BASE format (f128)
  - In 16-bit uSID format (f3216) → https://datatracker.ietf.org/doc/html/draft-filsfils-spring-net-pgm-extension-srv6-usid-12

- **The SRv6 Network Programming Framework** https://datatracker.ietf.org/doc/rfc8986/ allows for the definition of instructions (behaviors) that form a "Network Program" or **Policy**

- **SRv6 is not prescriptive** on the control plane architecture although BGP Enabled Service equivalence is being standardized in IETF

- It provides a **Single encapsulation** for underlay and overlay

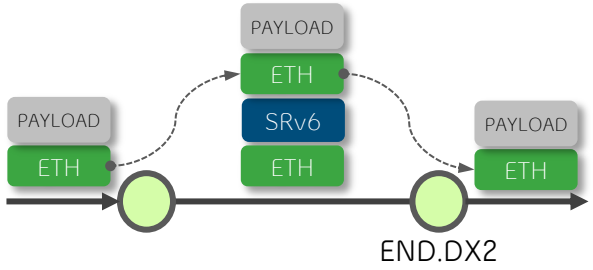
- Leverages **standard IPv6 routing** (no special FIB table, totally stateless)

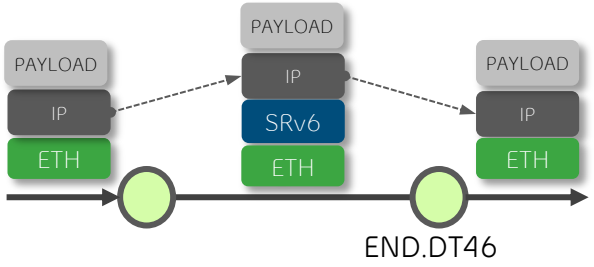


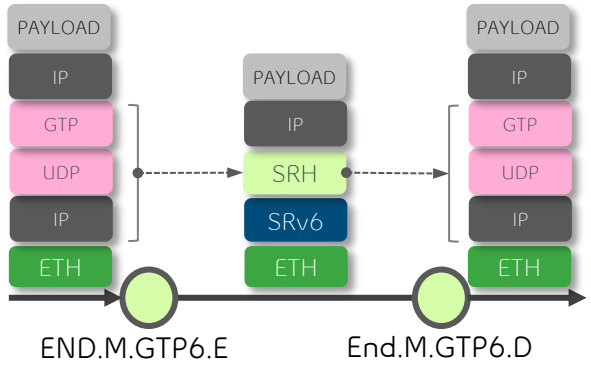
SRv6 Policies

- In SRv6, Layer 2 Cross-Connects (END.DX2), Layer 3 VPNs (END.DT46), Service Chains (END.SC) or GTP DMM Mapping are all **variations of SR Policies**

- Ingress traffic is **steered** into policies based on various criterias
  - Physical/Logical Interface
  - Src/Dst prefix match
  - 5-Tupple or advanced flow mapping
  - GTP Header
  - BGP route-coloring/Flowspec



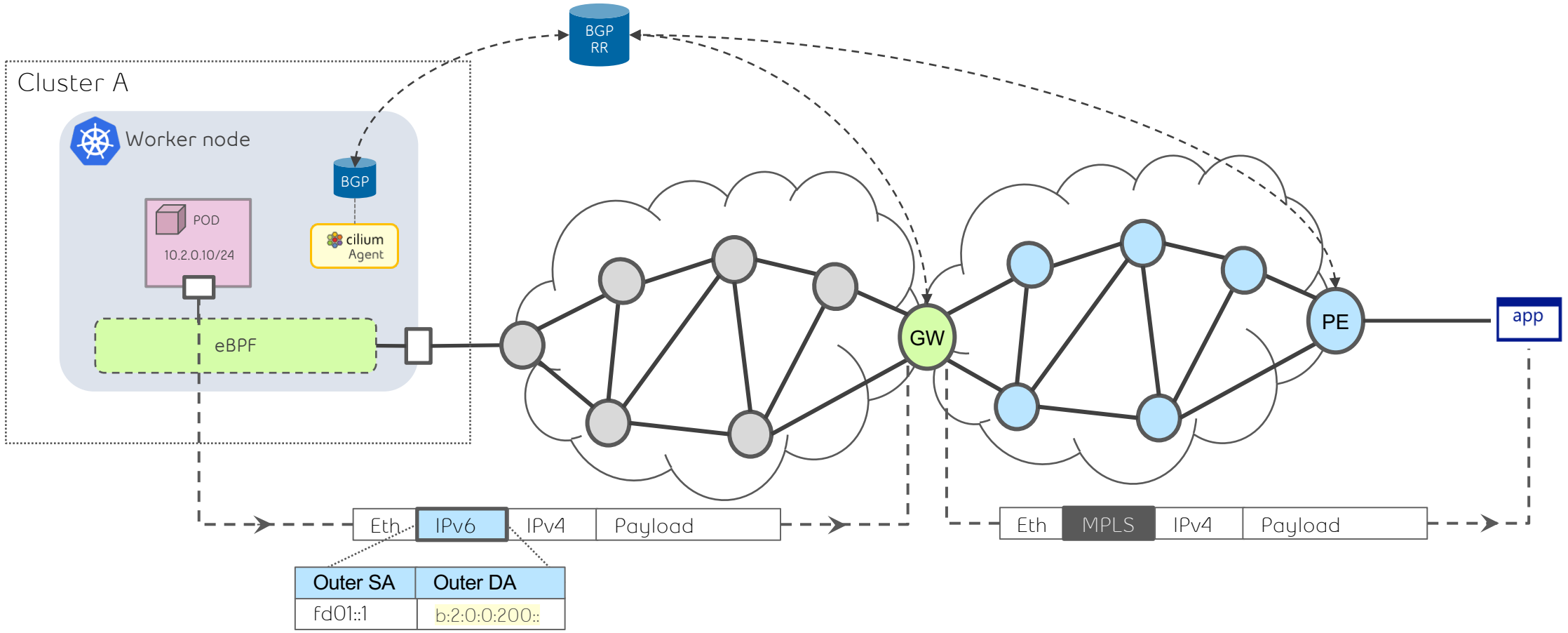




What if we used Cloud Constructs / SDN to apply build these policies ?

SRv6 – Extension of Architecture to Cloud Resources

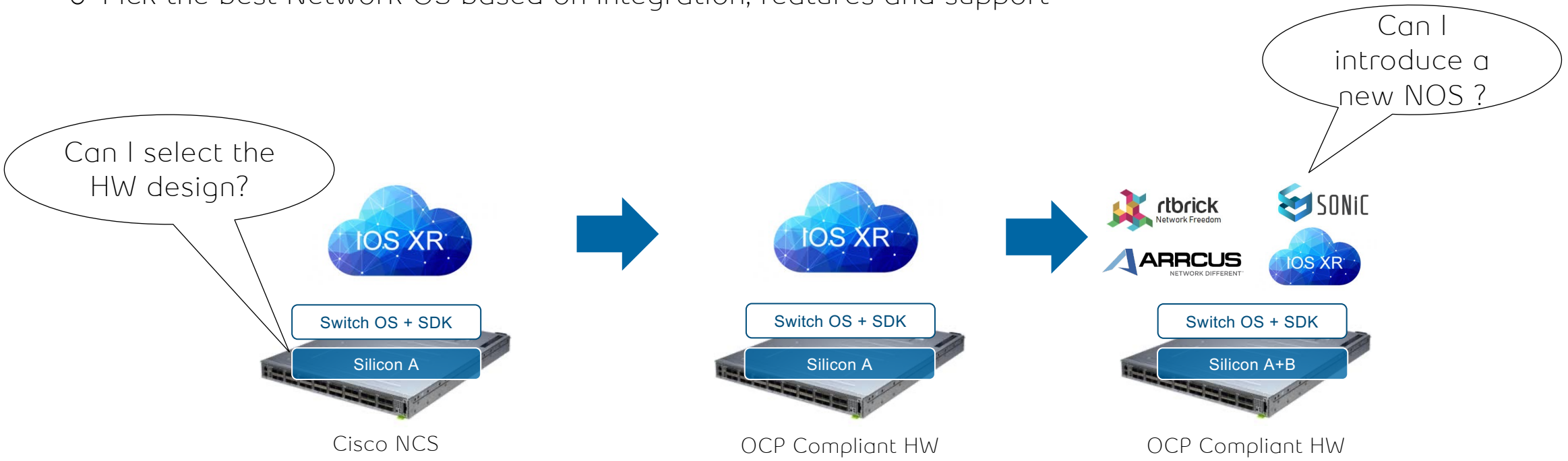
- POD still only has a single interface with only a default route.
- POD is associated with one or more VRFs if required by owner
- CRD defines VRF attributes (RT, name, etc.)
- **SRv6-to-MPLS (or else) Gateway maps SRv6 VPN to MPLS/VPN or equivalent**



Tenet #2
Supply Chain Simplification

North Star → True Platform Independence

- Pick the best HW platform based on need/cost → not because of OEM product choices.
- Pick the best HW platform based on right silicon → not because of OEM product choices.
- Pick the best Network OS based on integration, features and support



Select the right Software on the right Hardware at the right Cost

Opening up the eco-system

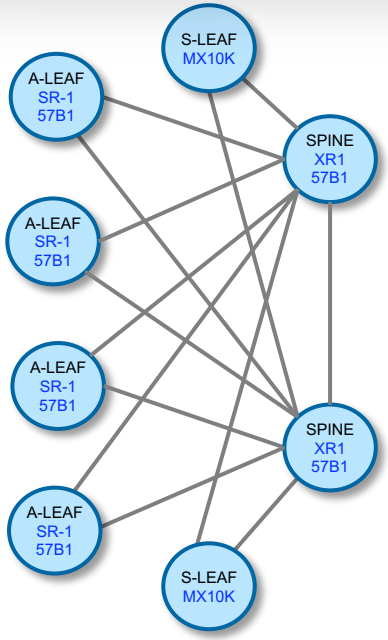
Supply chain issues are a real problem for execution, 2 approaches to address it

Current Approach

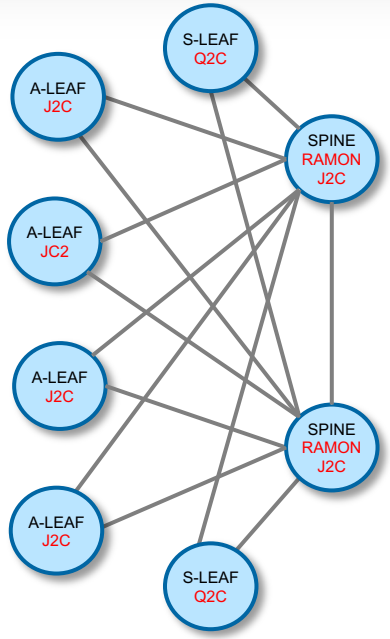
- Pick and chose SKUs per selected vendor – Buy a lot in advance
- Forces you in a vendor lock
- Changing vendor, means changing / introducing HW
- Bound to single vendor supply delays

Disruptive Approach

- Pick HW like COTS servers based on ASIC requirements
- Select SW vendor based on preference (agility, CI/CD, etc.)
- Less willingness from incumbents to participate
- Moves out of comfort zone so intent/automation is key



Journey Towards

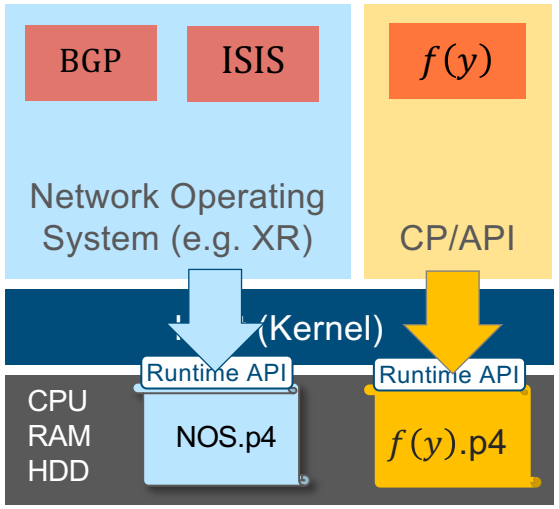


Common HW platforms, multiple vendor options

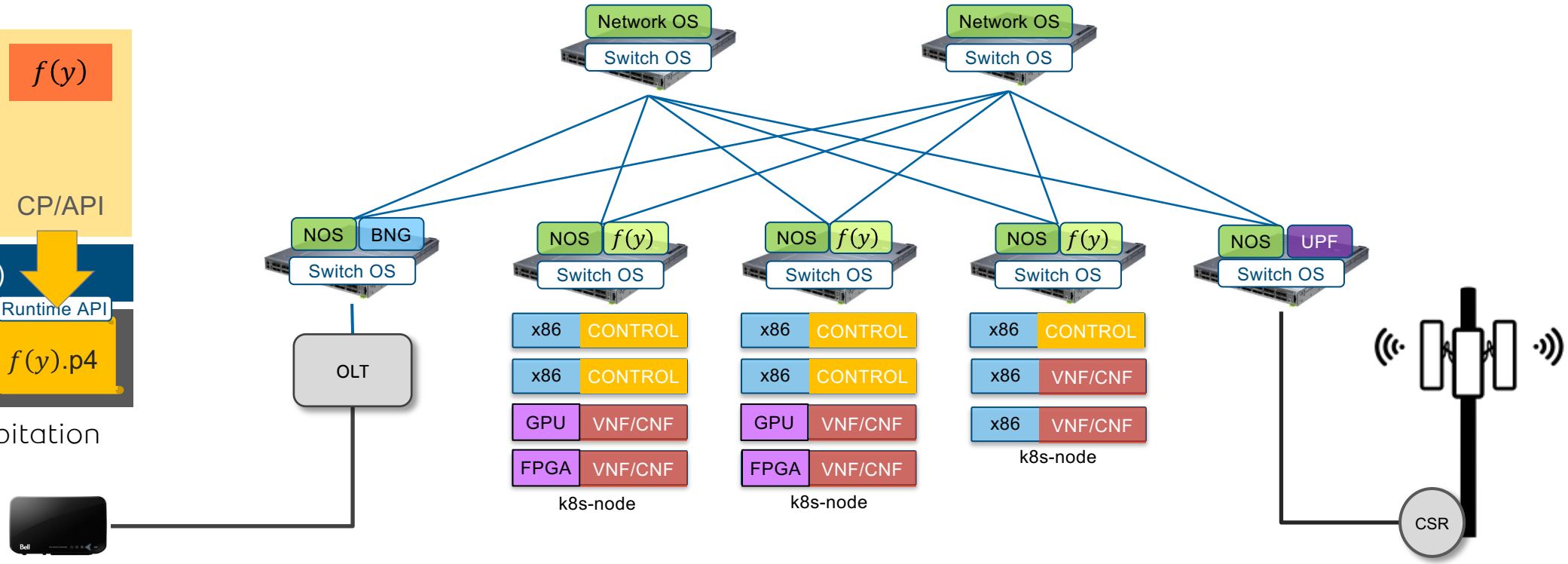
- RTBrick
- Arrcus
- DriveNets
- SONiC
- Cisco IOS-XR*
- CASA/BENU

CO Evolution Use Case – A Disruptive Approach (2019)

- Reduce network fragmentation by merging middle boxes with the network → SR Aware DDoS, BNG, etc
- Optimized Traffic Flow by remove appliance chokepoints → Better performance, experience
- Introduce agility without making the network too complex → Keep the network OS simple



P4 program cohabitation



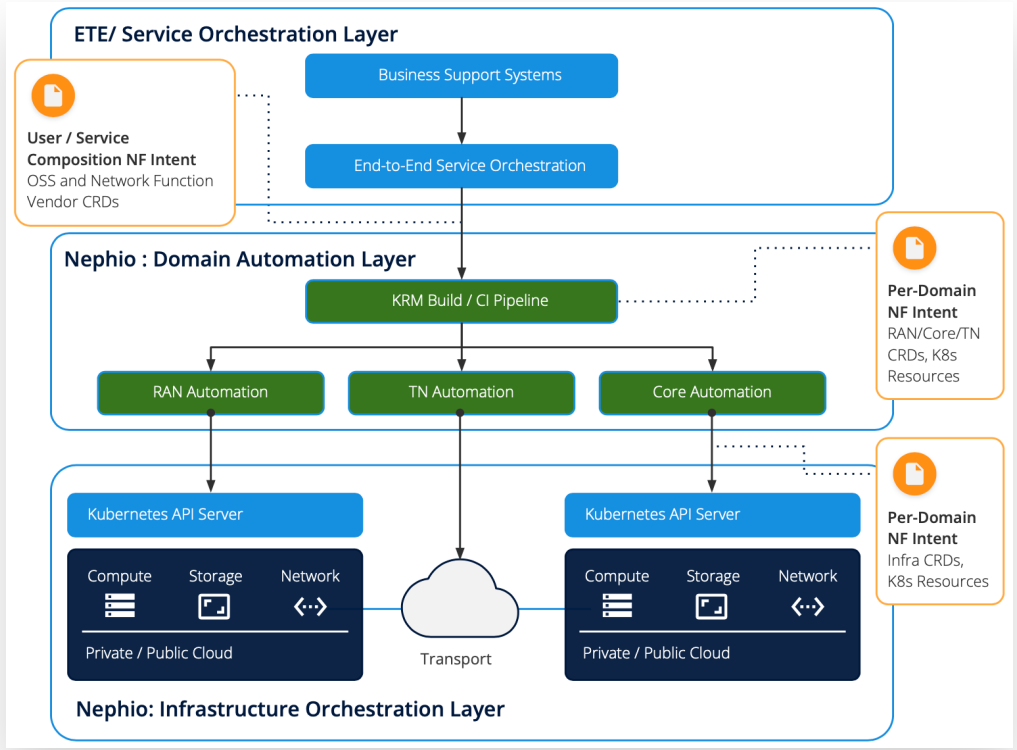
Evolution Towards P4-PINS / Stratum

Bell

Tenet #3
Cloud Native Transformation

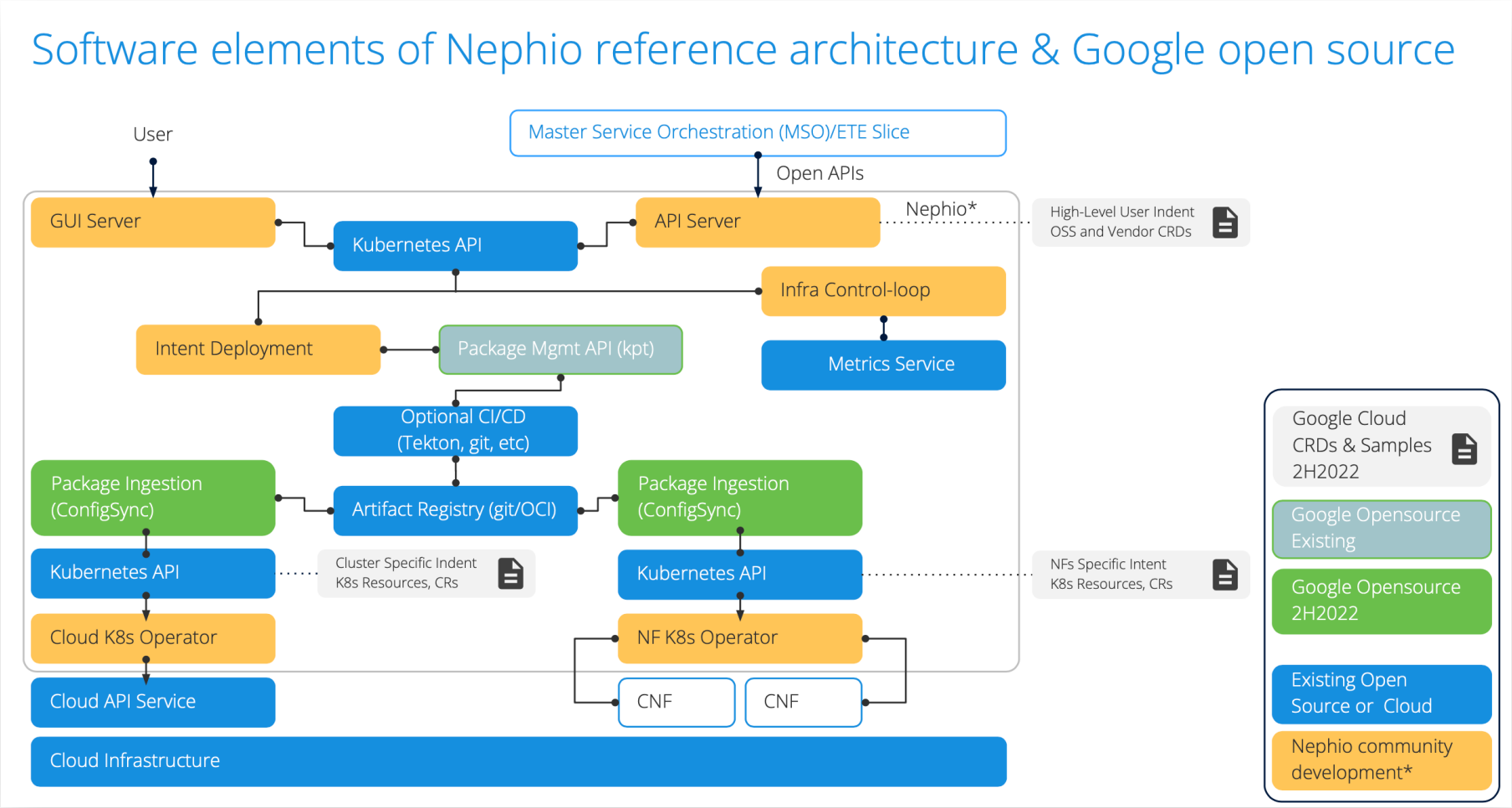
TNA (Google's Nephio)

- Is Google's implementation of Nephio (formerly CNNA)
- Built natively on the Kubernetes Resources Model (KRM)
- Uses an intent-based, declarative deployment model (CaD)
- Follows a GitOps approach with a specific toolset (defined by Google)
- Leverages Kpt functions for implementation of assignment, design & configuration mapping (CRD to CRD / K8S resources, CRD to config)
- Built on top of Google's cloud ecosystem
- Initially built for CNFs but to be extended covering VNFs/Physical devices (DC Networking, external/transport)
- Heavily relies on the controller/operator pattern of Kubernetes, and requires blueprints & artifacts for the NF's to be built & supplied by NEP's



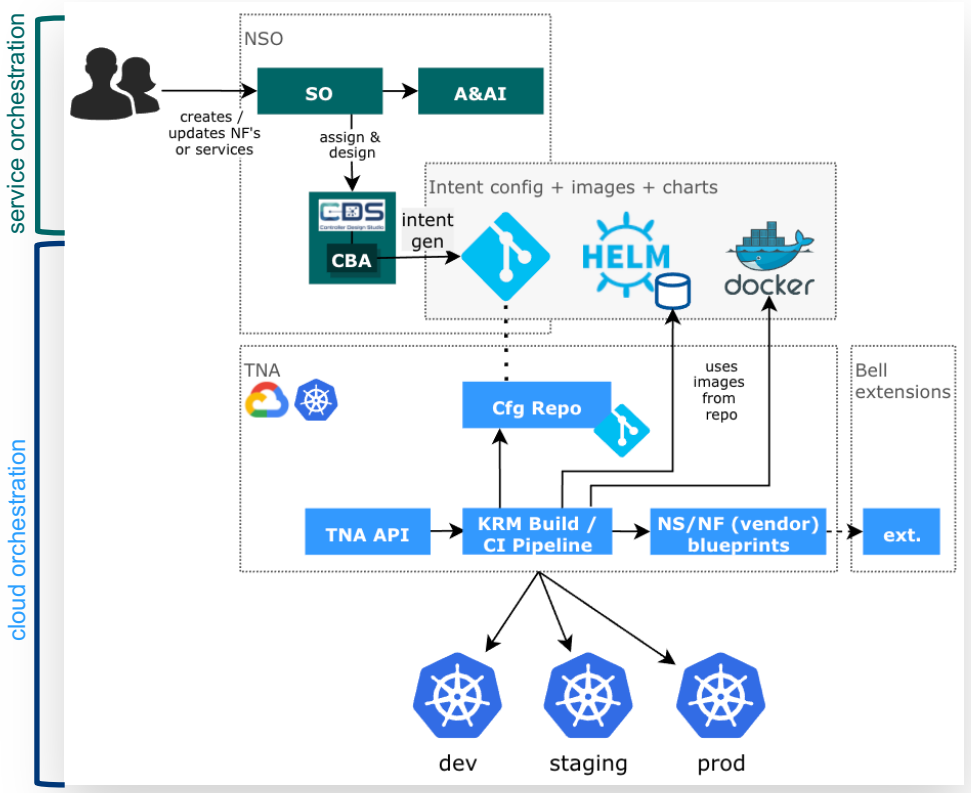
TNA is Google's implementation of the Nephio open-source project

Nephio Reference Architecture



E2E service orchestration w/ TNA

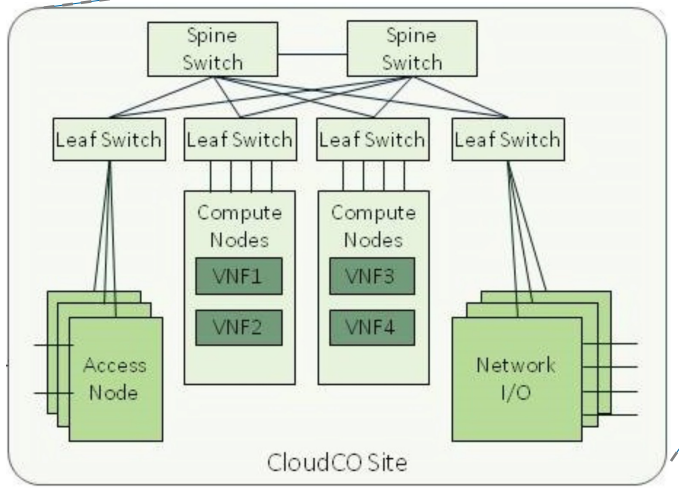
- Developers define a **high level** service model (once, which includes the CRDs used) in SDC, and define corresponding attributes to be used in overrides
- They build (once) CDS packages (CBA's) that implement the automated assignment & design logic, but **only for high-level CRD attributes**
- When deploying, they instantiate the service instances via SO (providing any parameters that are not auto-assigned or generated, and specifying the environment to deploy to)
- CDS leveraged to generate/push the intent to the intent repo
- TNA handles the deployment, technology-specific assignment & config generation per vendor-supplied blueprints + can handle assignment & design specific to Bell via extension of those blueprints
- Kubernetes operators (vendor supplied) handle configuration generation and lifecycle of the cloud-based services / CNF's



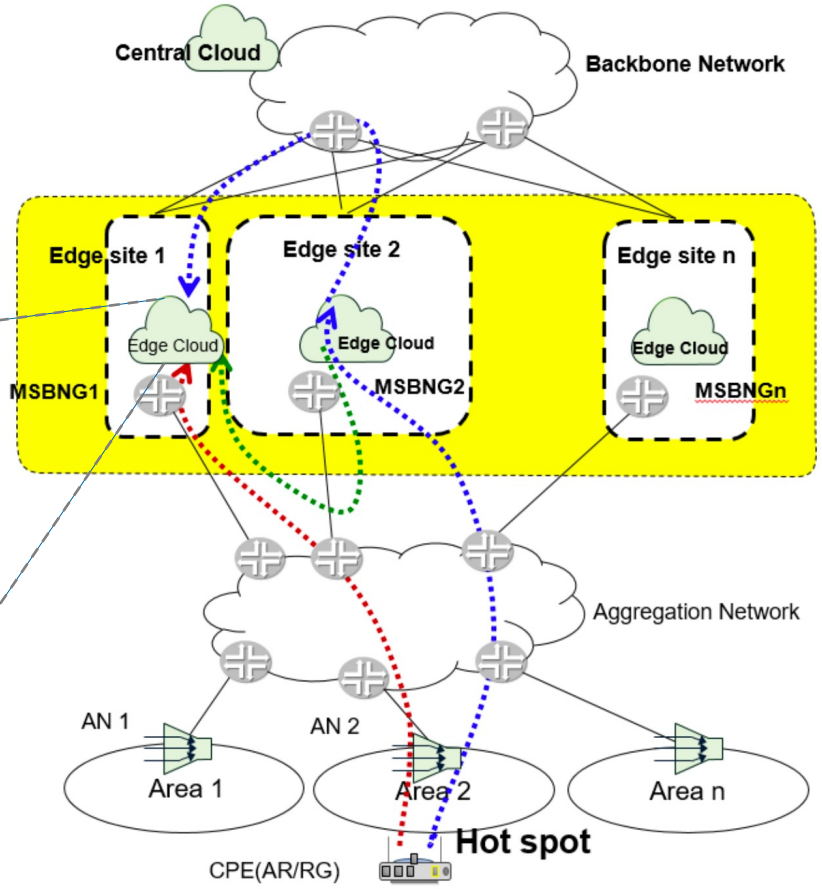
NGCO Effort

Path Towards a Next-Generation CO

- **Cloudification of Central-Offices**
  - Design for Macro and micro-nodes (C-RAN, D-RAN, URLLC)
  - Decentralized Management Control
  - PNF/VNF/CNF & VAS support
  - Legacy and transformed CO Co-Existence (TR-408)
- **Embedded within a Metro Computing Fabric**
- **NGCO Evolution**



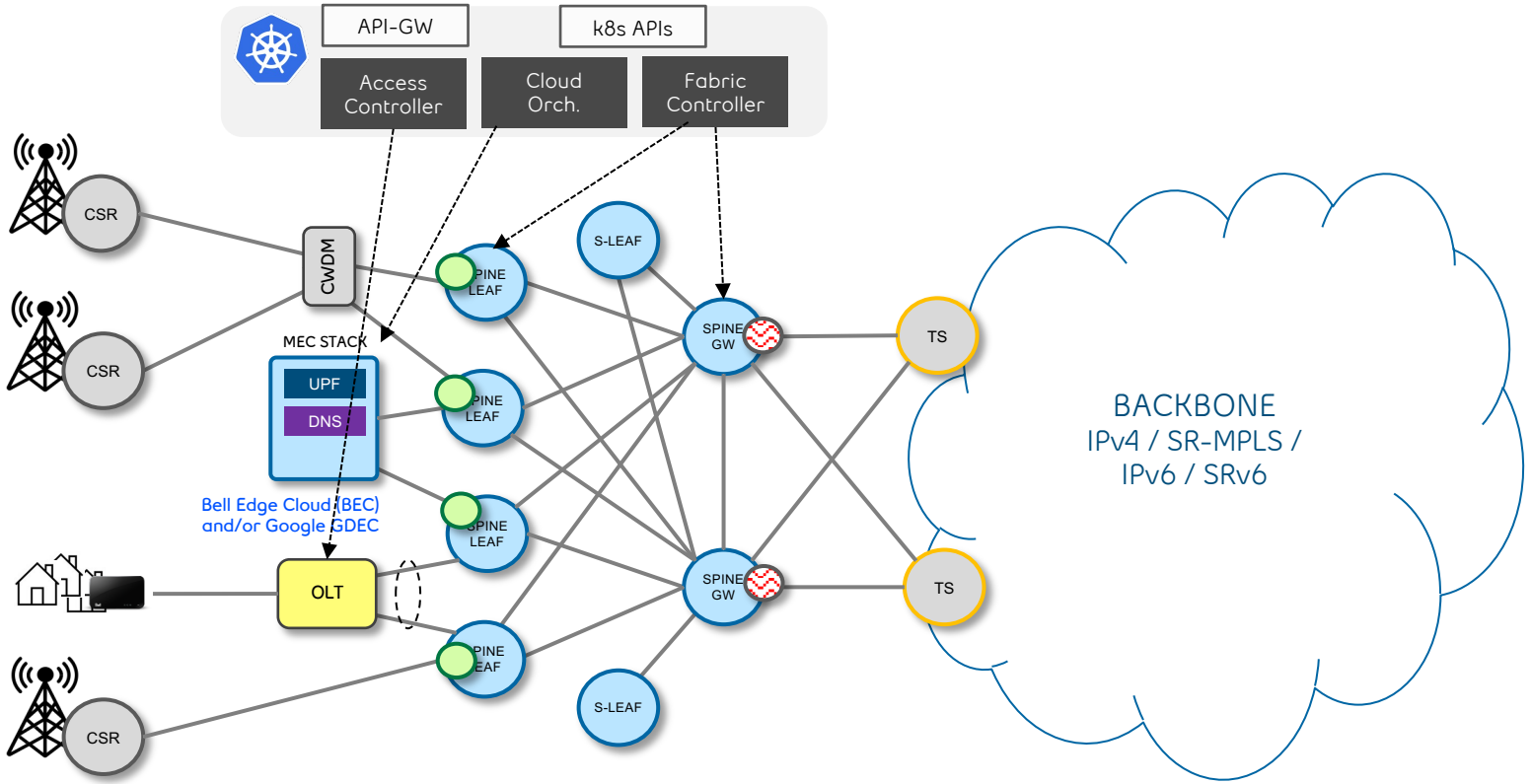
CloudCO Ref Architecture (BBF TR-384)



Metro Computing Networks (BBF TR-466)

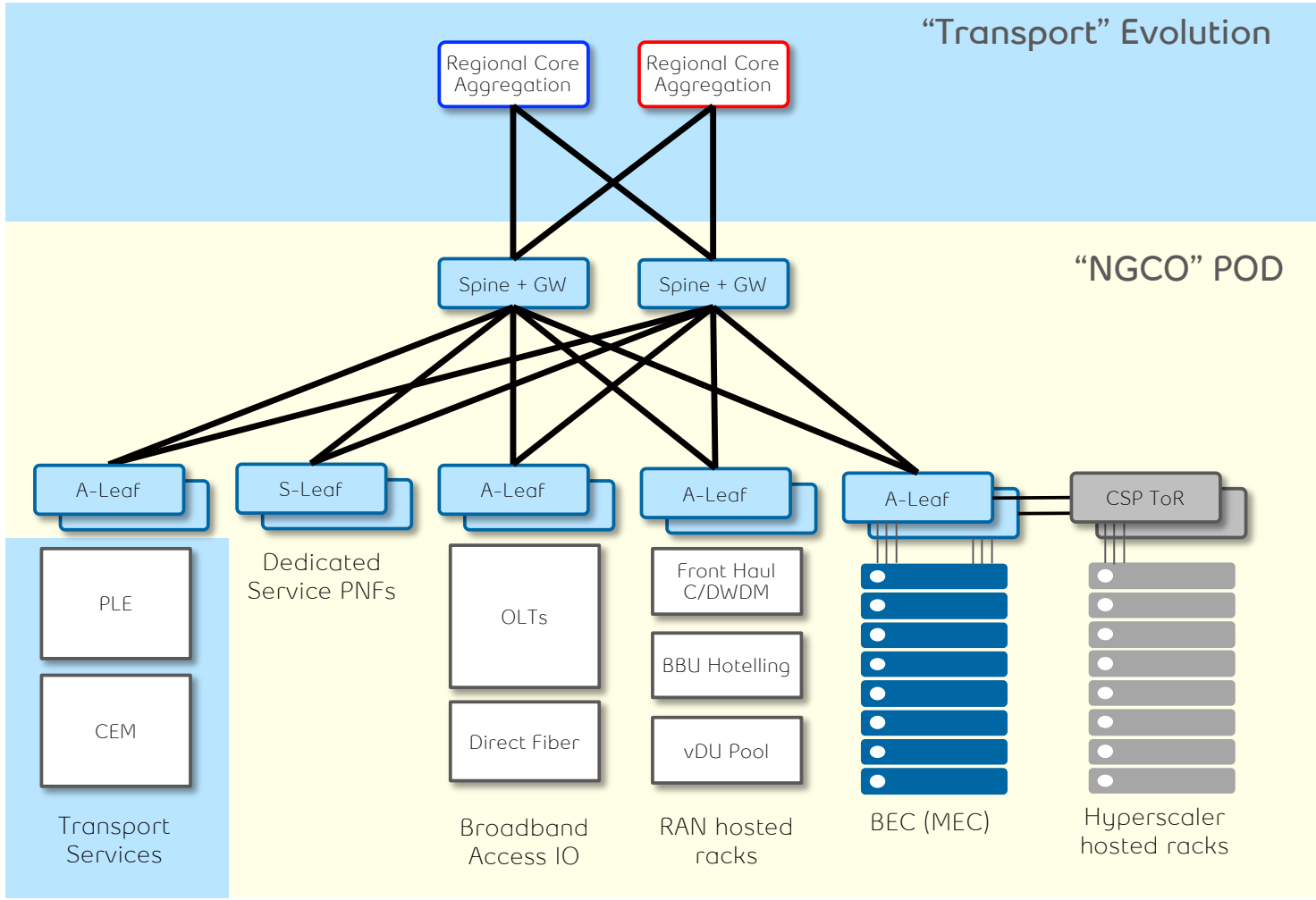
NGCO Vision

- **Disaggregated and Cloudified Fabric to support**
  - 5G xHAUL evolution
  - PON rollout and evolution
  - Edge Computing Deployments
  - Metro Network Refresh



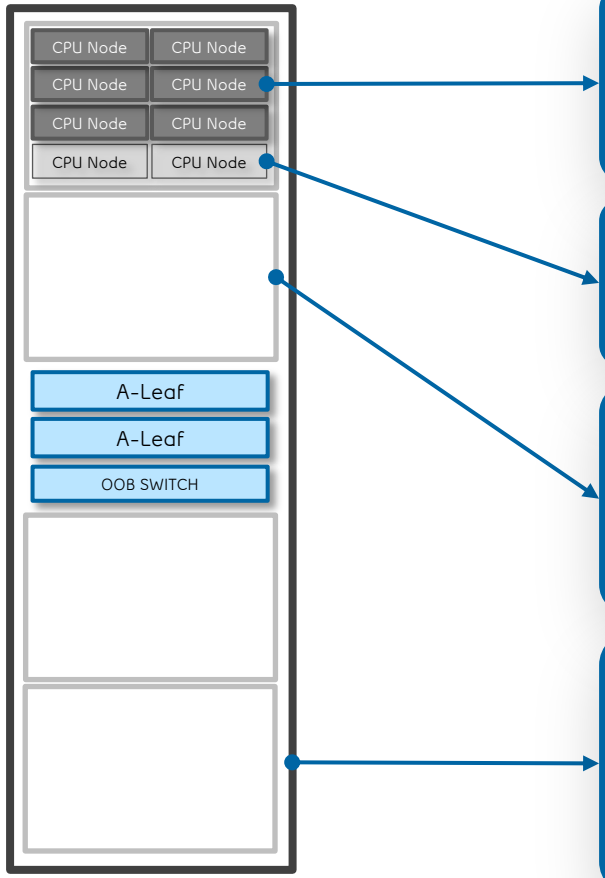
NGCO POD Target Design

- Evolution of Broadband and IP Edge
- Cloudification of CO architecture
- Cloudification of CO orchestration
- Cloudification of VAS services



NGCO POD – Composable Infrastructure

- **Leveraging from Open Compute Project and hyper-scalers practices**
  - https://www.opencompute.org/wiki/Open_Rack/SpecsAndDesigns
  - https://www.systemverilog.io/modular-design-in-open-compute-project



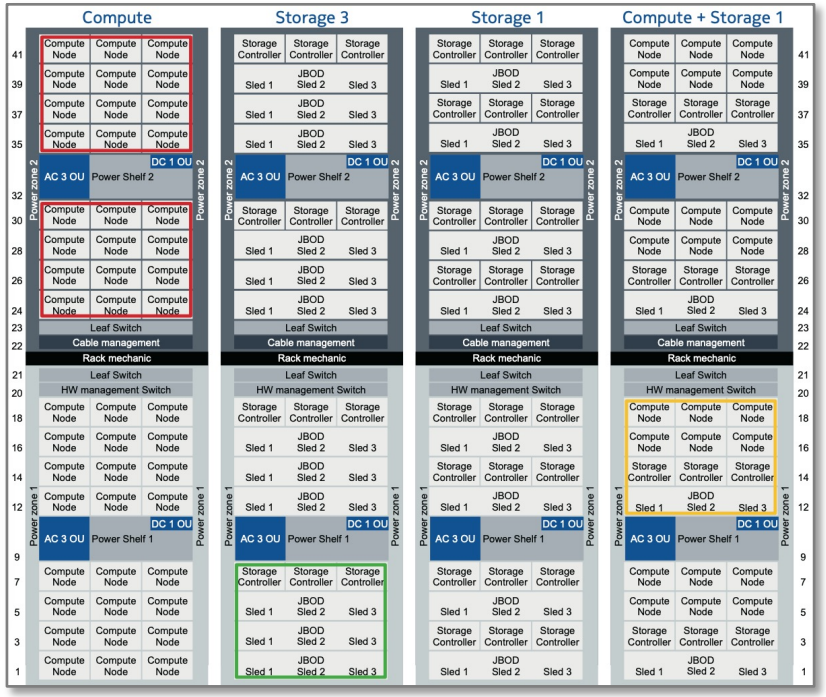
RSU : Rack Scale Unit

- Modular SLED based form factor for easy insertion/removal
- Easily adaptable to varying workload requirements (CPU Node, Storage Node, Network IO, GPU, etc.)
- Simplified life cycle evolution over time

- Initial deployment with base required workload (reduces over engineering)
- Leverage HPE greenlake* (or similar) for pre-installed ready to use additional blades (pay on use)

- Rack decomposed in various “blocks” that can be purposed for specific needs.
  - Standard compute clusters
  - NF server farm (ie Lanner)
  - Network IO (OLT, vDU, etc.)

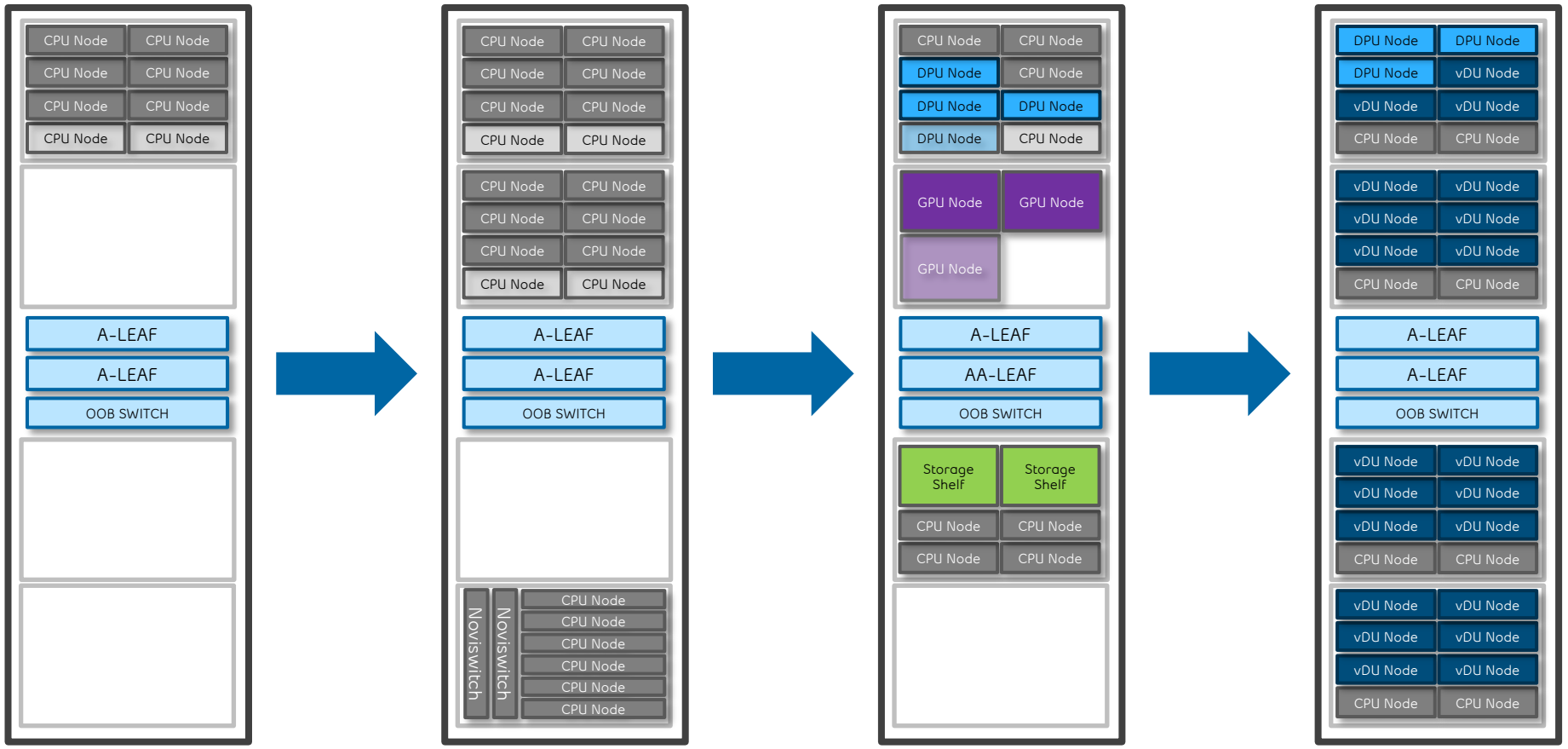
- Factory or integrator assembled for quick, cost effective, turn-up
- **Leverage expertise from industry (WWT, Vertiv, Schneider, HPE, Mobia, etc.) rather than custom in-house engineering**
- SKU based rack layout for easy ordering
- ”Just in time” deployment.



OCP Rack v2 – Modular Blocks

NGCO POD – Enhanced Modularity

- Only achievable with well defined design blueprints and effective supply-chain



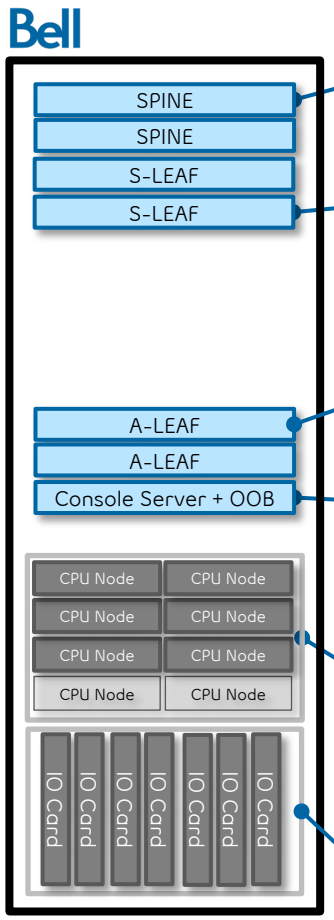
Pre-built, pre-cabled with base config and spares

Capacity addition at Day 2 also with alternate platforms

Supporting use case driven modular workloads

Specialized SKUs for focused deployments

Anatomy of the NGCO POD



SPINE

Provides external network connectivity, MPLS interop ("stitching" PE functionality)

S-LEAF

Standard S-Leaf when deploying centralized or specialized PNF function

A-LEAF

Provides COMPUTE/OLT/DIRECT(CSR) connectivity, SRv6 PE functionality, SRTE policy endpoint (ingress/egress).

Console Server

Provides **serial** and **IPMI** access to POD elements for lights-out management. Targeted LTE backup support.

Compute Nodes

Provides flexible compute resources for MEC workloads to support various requirements (GPU, CNF, Storage, etc.)

Can support various cloud platform implementations (**Openshift**, **Anthos Bare Metal**, EKS Anywhere, Azure Edge)

Open approach that can be offered through various suppliers.

Access I/O

OLT Fixed or modular chassis, Fiber Cross Connects.