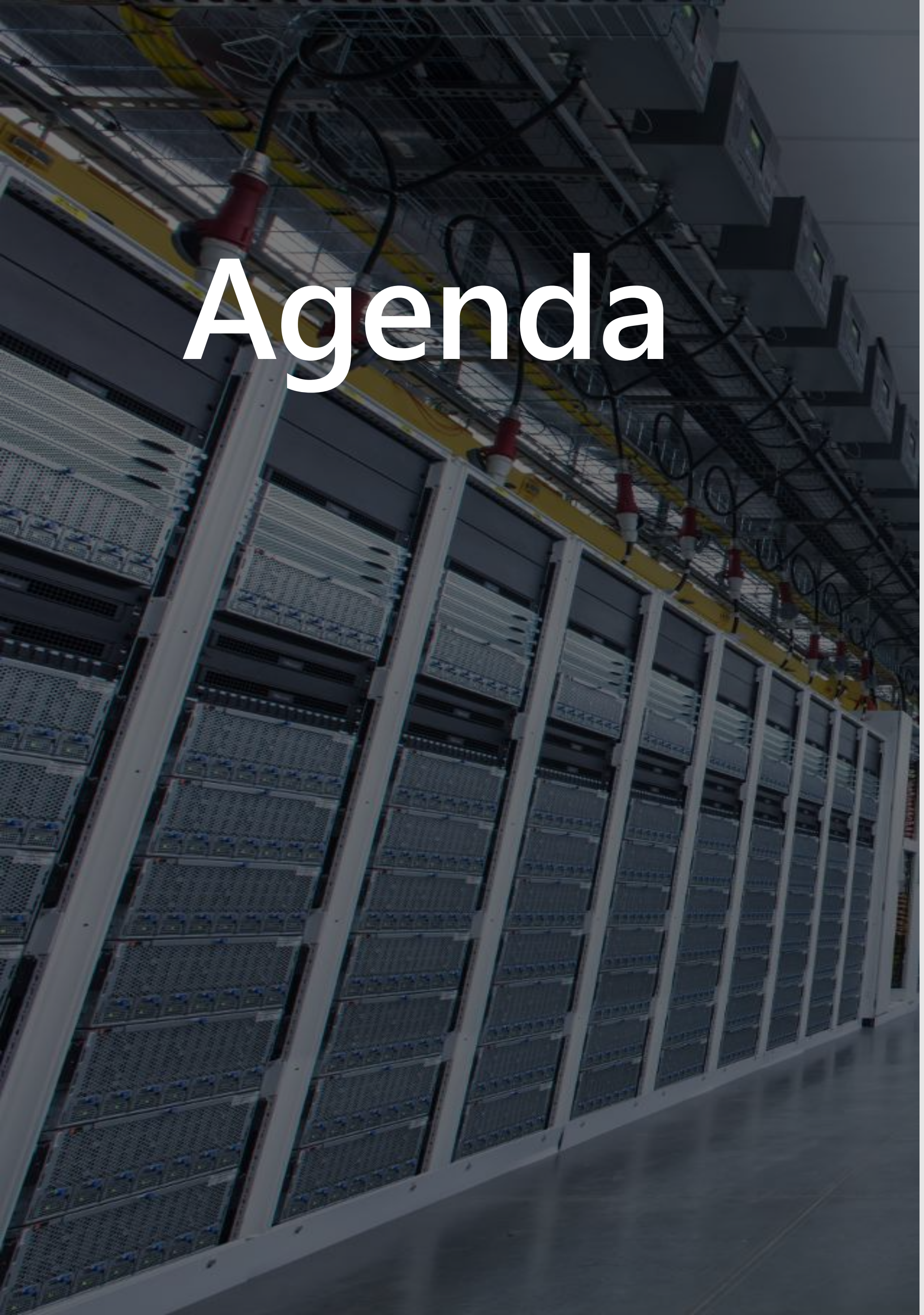
# SDN at Hyperscale

## Azure networking services for modern infrastructure

Rui Carmo
Cloud Solution Architect
@rcarmo

Microsoft



# Agenda

Azure Datacenters

Datacenter Networking

Azure SDN

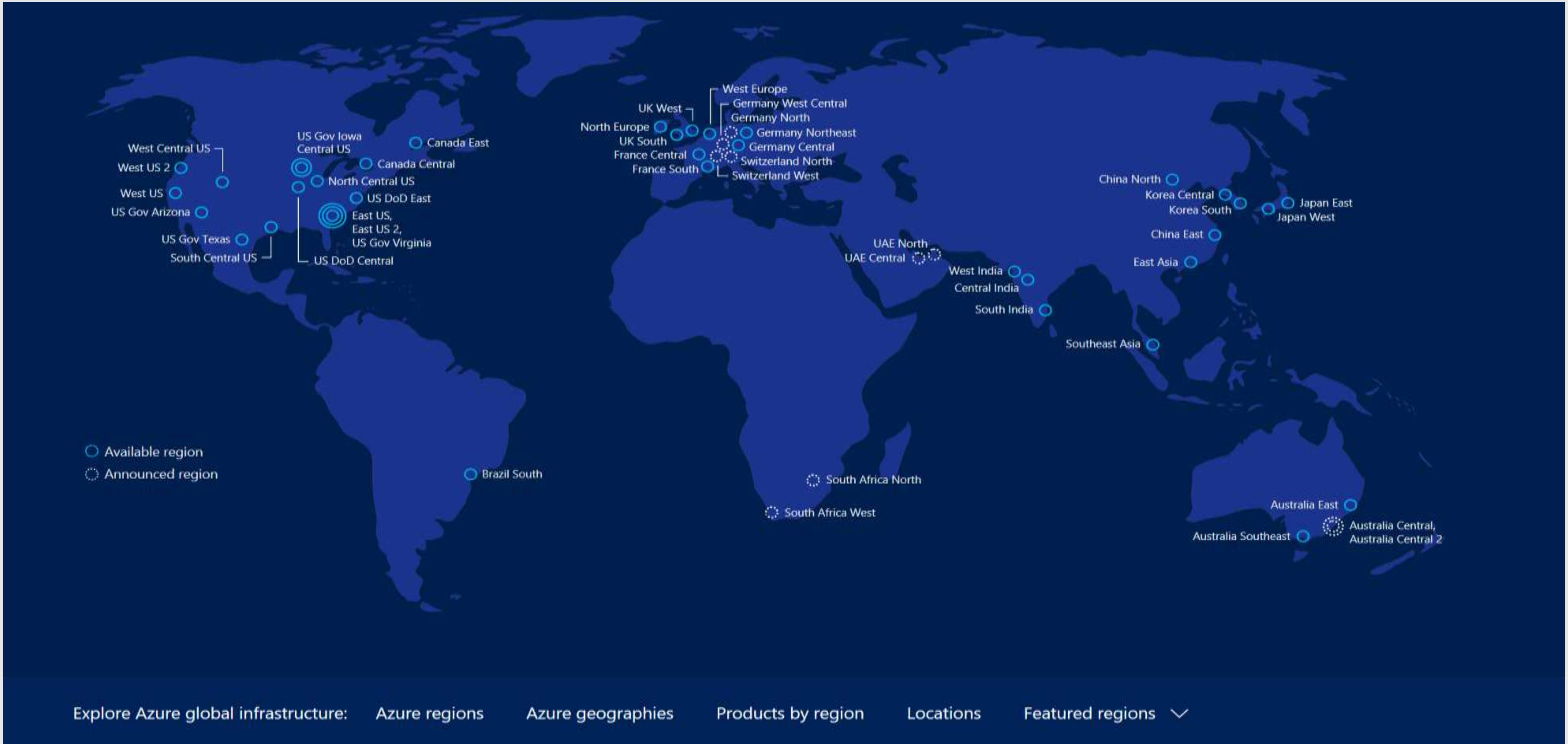
Container Networking

Looking Forward

# Azure Datacenters

# Azure Hyper-Scale Global Infrastructure

## 100+ Datacenters Across 46 Regions +4 new regions announced



# Geos and regions

## The world is divided into geographies

Defined by geo-political boundaries or country borders

Defines the data residency boundary for customer data

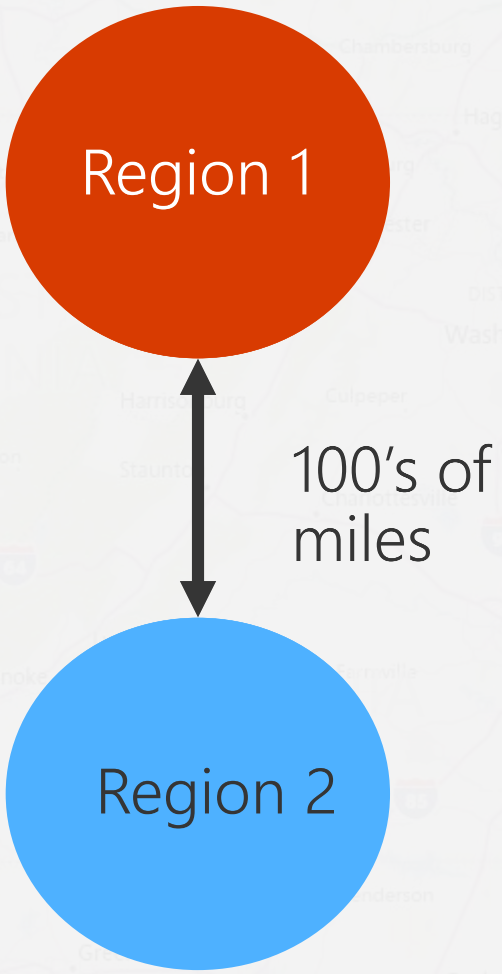
## A region is defined by a bandwidth and latency envelope

<2ms latency diameter (round trip)

Customers see regions, not DCs

Different fault and flood zones, electrical grid, hurricane zone

Typically hundreds of miles apart

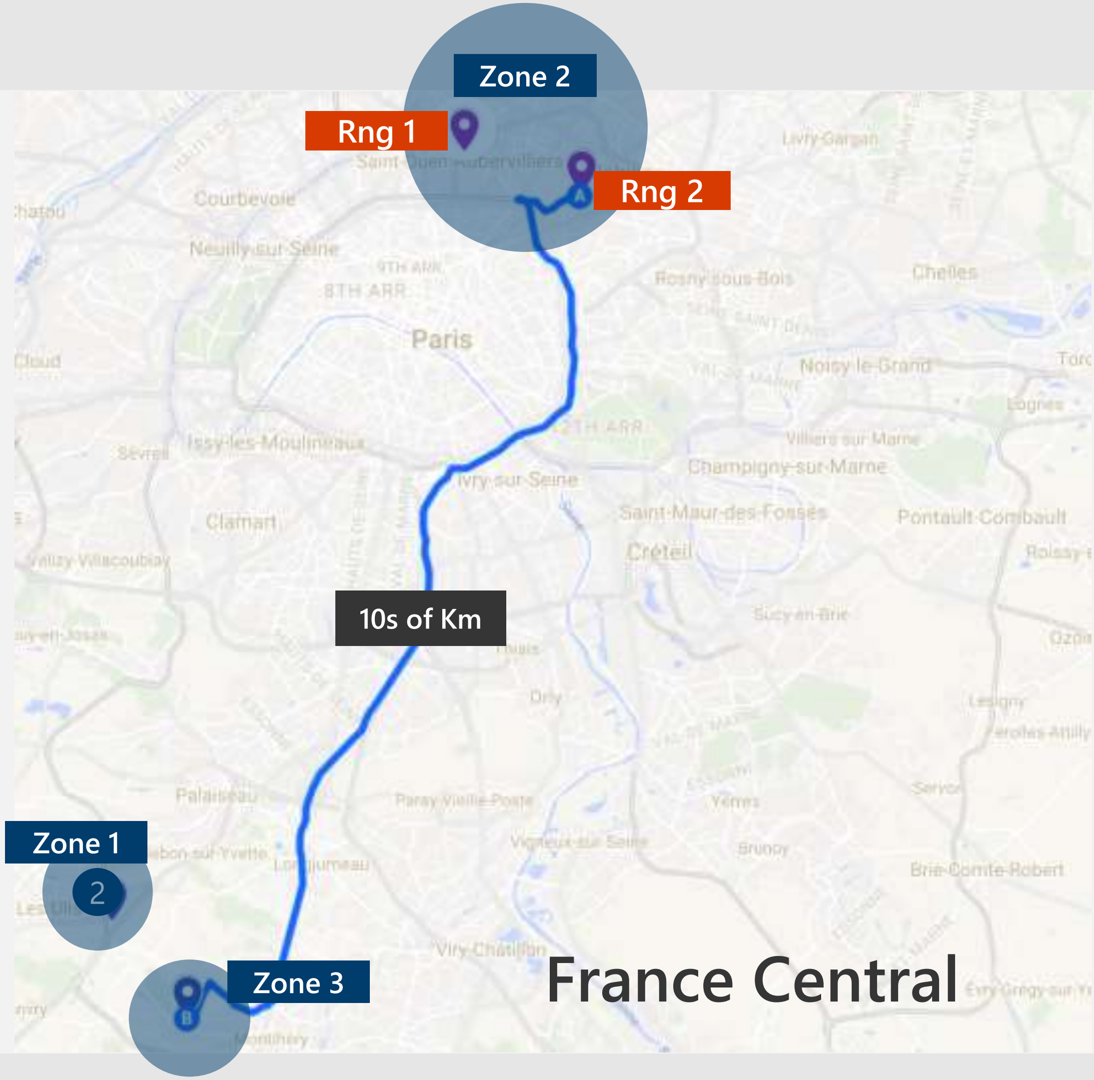


# Regions > Availability Zones > Data Centers

A region has at least 3 Availability Zones

- Three is enough for quorum
- 600 µSec latency diameter
- Different water, power lines, network, generators
- Customers can do application-level synchronous replication between AZs

Each Availability Zone consists of one or more data centers




Quincy, WA

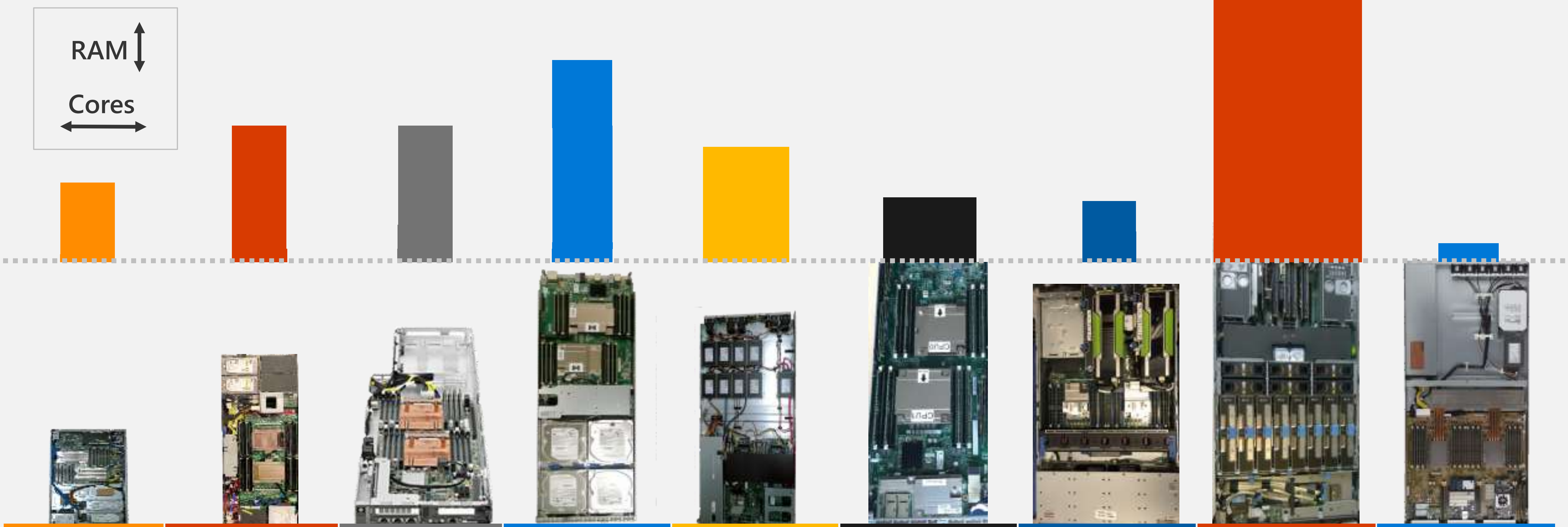

Quincy, WA

# Amsterdam, NL


Cheyenne, WY


Cheyenne, WY

# Azure server generations




| | Gen 2 | Gen 3 | HPC | Gen 4 | Godzilla | Gen 5.1 | GPU Gen 5 | Beast | Gen 6 |
|---|---|---|---|---|---|---|---|---|---|
| Processor | 2 x 6 Core 2.1 GHz | 2 x 8 Core 2.1 GHz | 2 x 12 Core 2.4 GHz | 2 x 12 Core 2.4 GHz | 2 x 16 Core 2.0 GHz | 2 x 20 Core 2.3 GHz | 2 x 8 Core 2.6 GHz | 4 x 18 Core 2.5 GHz | 2 x Skylake 24 Core 2.7GHz |
| Memory | 32 GiB | 128 GiB | 128 GiB | 192 GiB | 512 GiB | 256 GiB | 256 GiB | 4096 GiB | 192GiB DDR4 |
| Hard Drive | 6 x 500 GB | 1 x 4 TB | 5 x 1 TB | 4 x 2 TB | None | None | 1 x 2 TB | None | None |
| SSD | None | 5 x 480 GB | None | 4 x 480 GB | 9 x 800 GB | 6 x 960 GB PCIe Flash and 1 x 960 GB SATA | 1 x 960 GB SATA | 4 x 1920 GB NVMe and 1 x 960 GB SATA | 4 x 9600 GB M.2 SSDs and 1 x 960 GB SATA |
| NIC | 1 Gb/s | 10 Gb/s | 10 Gb/s IP, 40 Gb/s IB | 40 Gb/s | 40 Gb/s | 40 Gb/s + FPGA | 40 Gb/s | 40 Gb/s | 40 Gb/s |
| GPU | | | | | | | 2 x 2 Compute GPU | | |
| FPGA | | | | | | | | | Yes |

# Project Olympus

**Flexible and Modular design to handle wide variety of public cloud workloads**

**Open Compute Project open source design**

## Compute

Intel, AMD, ARM64 CPUs

High density GPU expansion for HPC/AI

NVM (DRAM+battery) and 3DXP for low-latency

## Storage

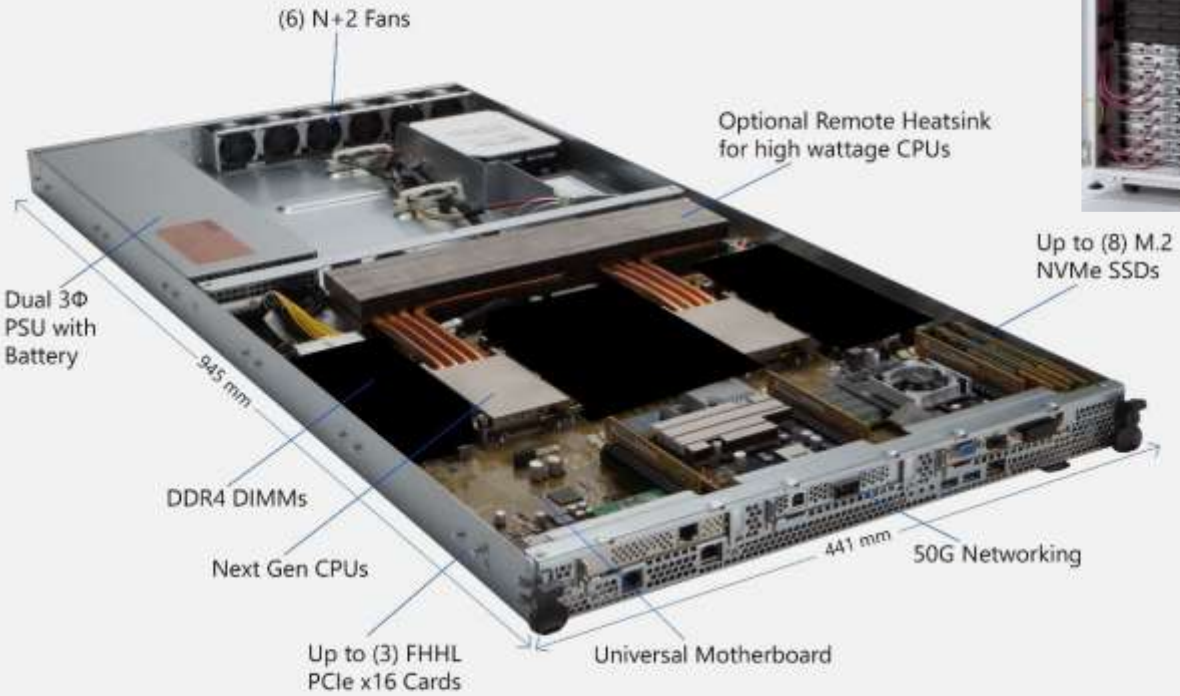
High density HDD and Flash expansion

Microsoft custom designed SSDs

## Networking

50 Gbps networking

Accelerated VMs using FPGAs



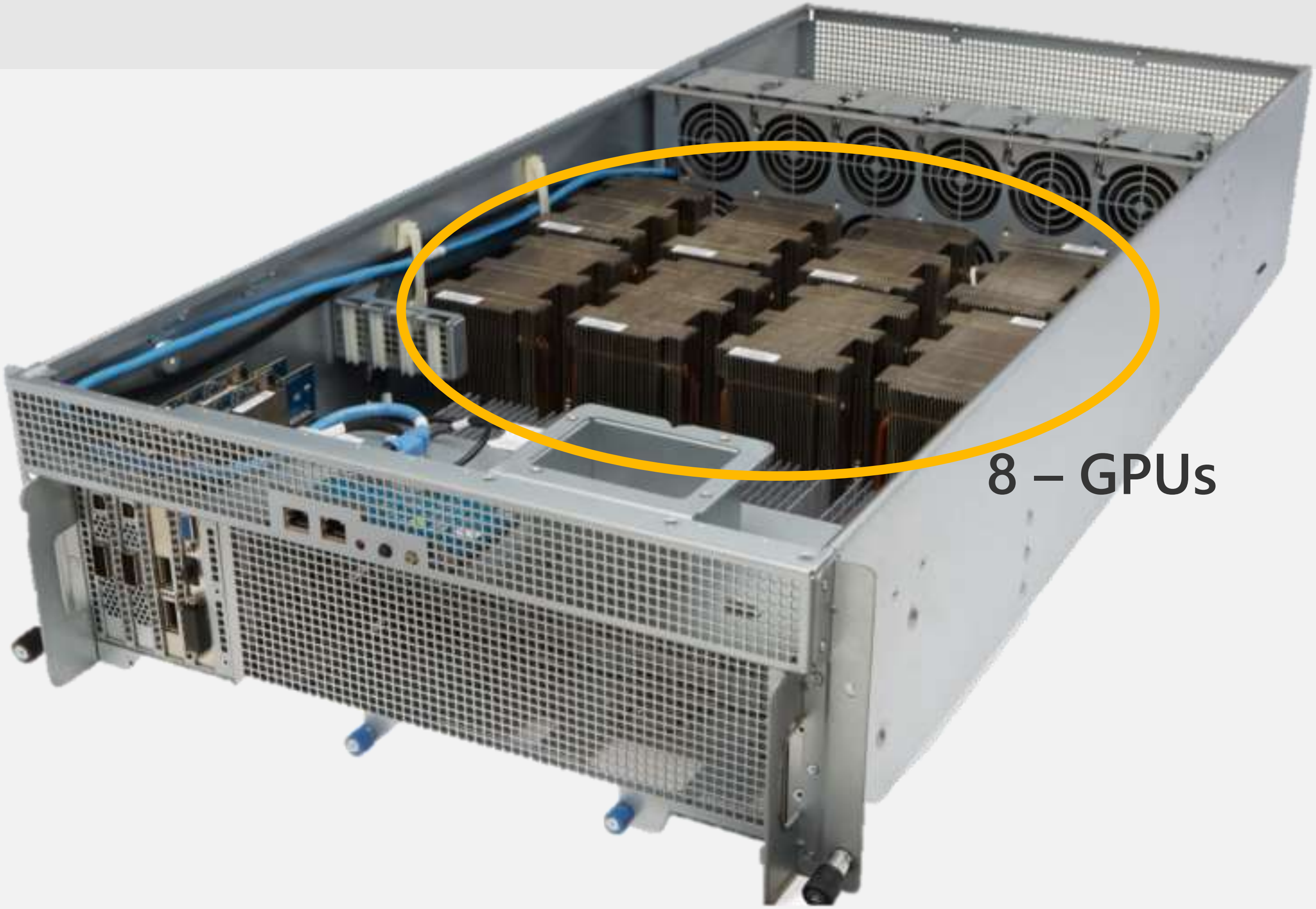

Microsoft SSD

# **High-density** GPU SKU for AI

Microsoft

NVIDIA

ingrasys




New industry standard design on **Project Olympus** for machine learning

Extreme performance scalability - Interconnectivity for up to 32 GPUs

*Powered by NVIDIA Pascal and NVLINK*

# Storage

**Rack-scale appliance**

52U rack, front-to-rear cooling, 3-phase 220V AC power

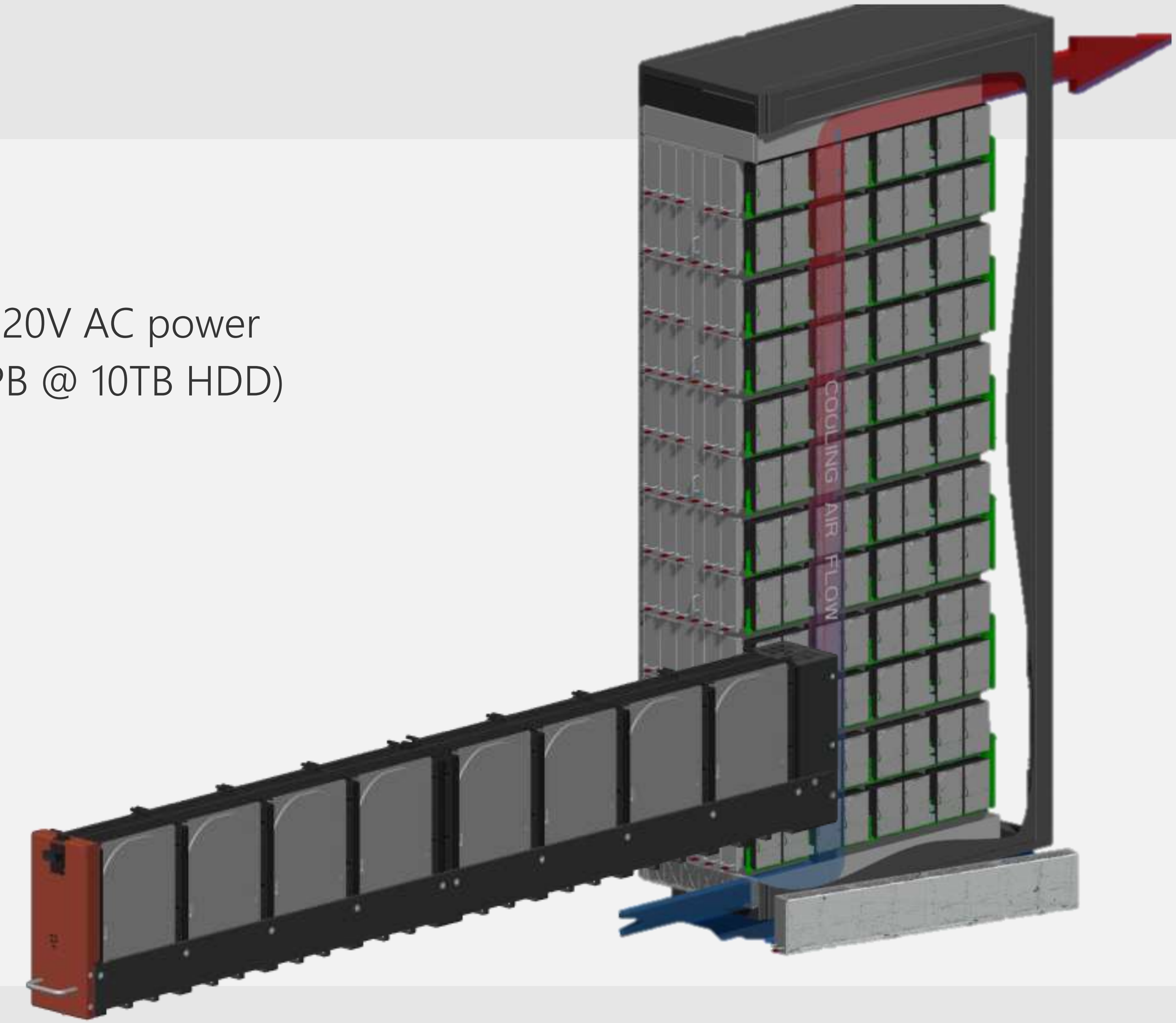
1152 3½" HDDs in 72 drive trays (raw 11.5PB @ 10TB HDD)

~3000 lbs (1.4 tons)

2 servers, PCIe bus stretched rack-wide

3.5 kW/rack

2 x 40GigE to datacenter network



# Datacenter Networking

# Azure's inter-DC network

**Global optical**

MSFT dedicated optical network

Pure dark fiber in regions and between large regions

Private waves

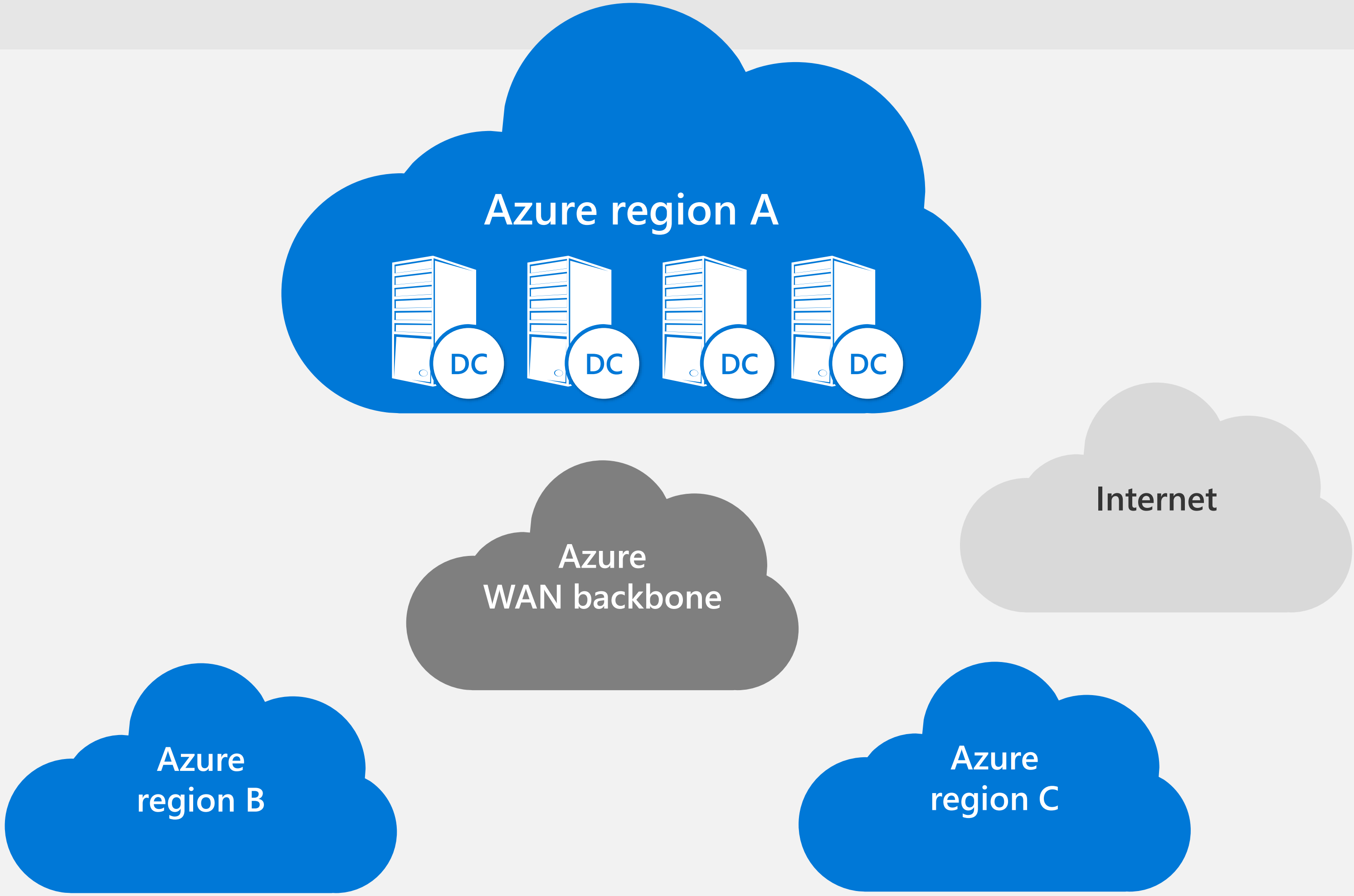
**Inter-region**

Backbone

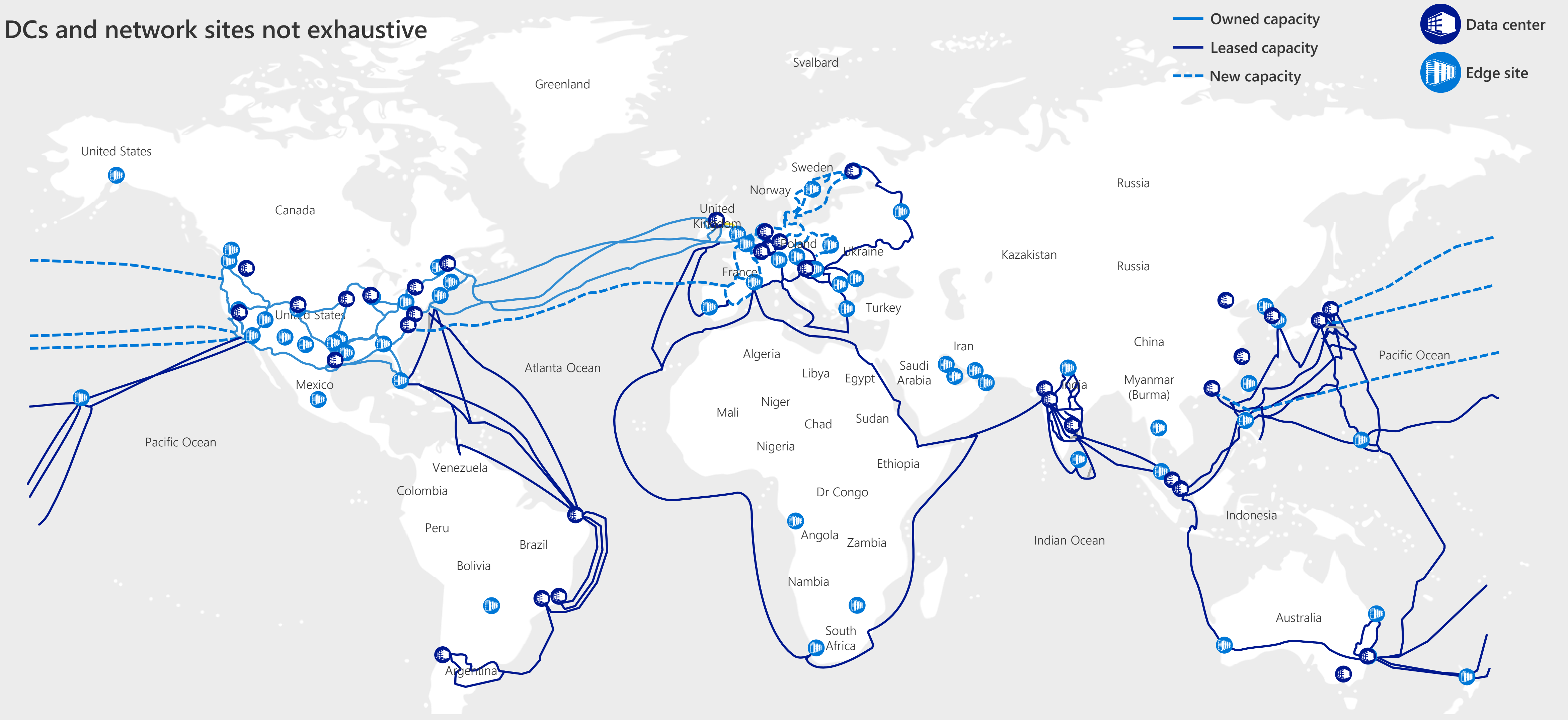
SWAN – Custom 100Gb Optical

**Intra-region**

Regional Network Gateway



# Azure inter-DC dark fiber backbone



# RNG regional architecture

**Regional network gateway**

Massively parallel, hyper scale

DC interconnect

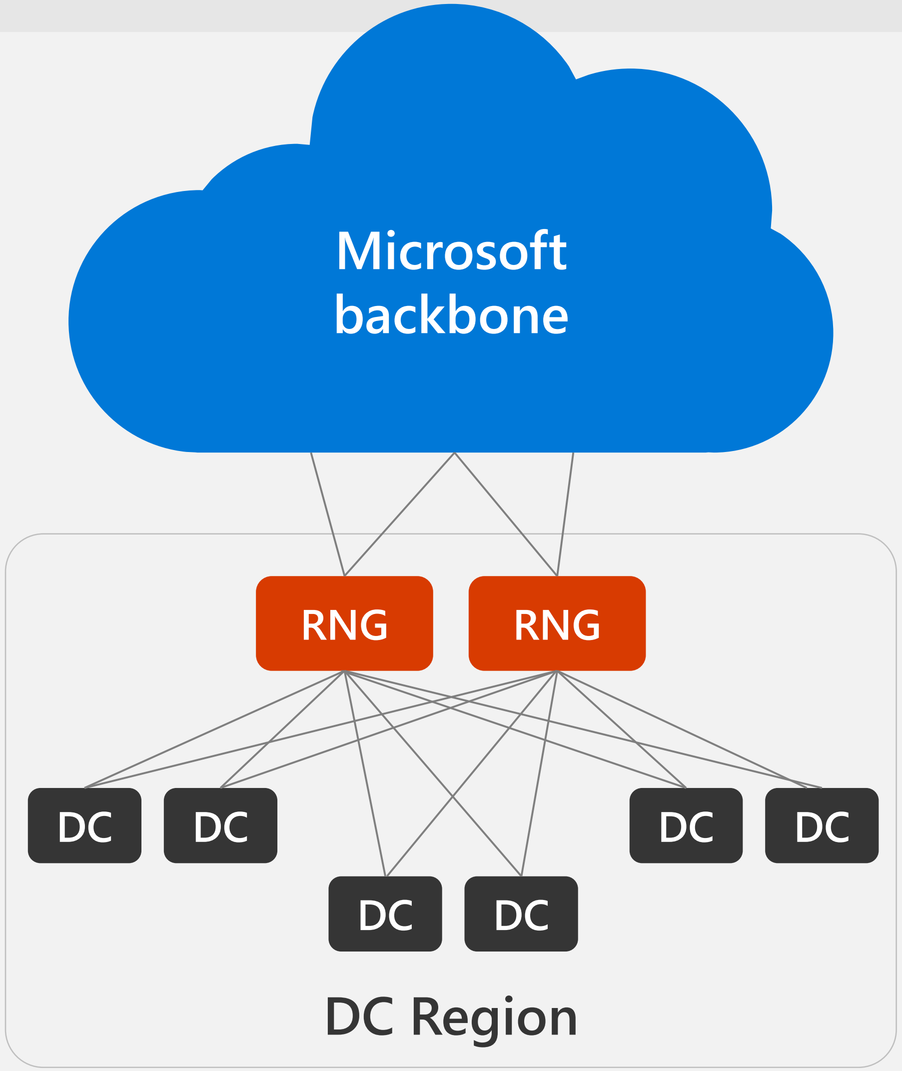
Space and power protected

**RNG data centers**

Small, Medium, or Large (T-shirt sizes)

Only contains server racks, DC network

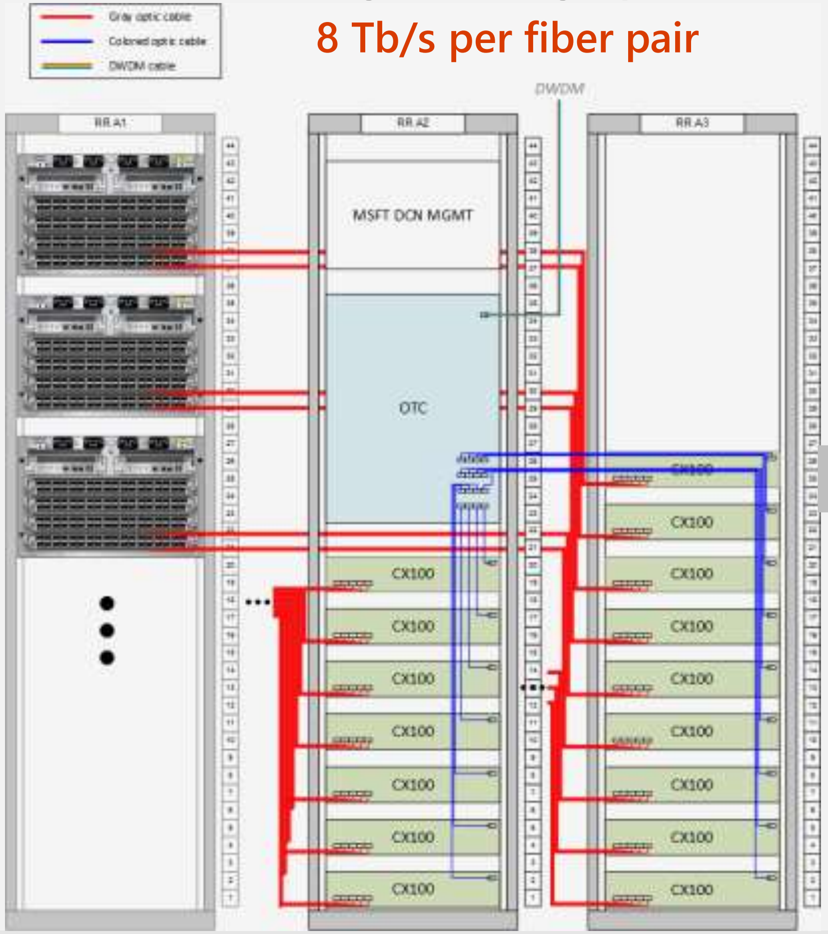
RNGs are sized to support growing the region by adding data centers



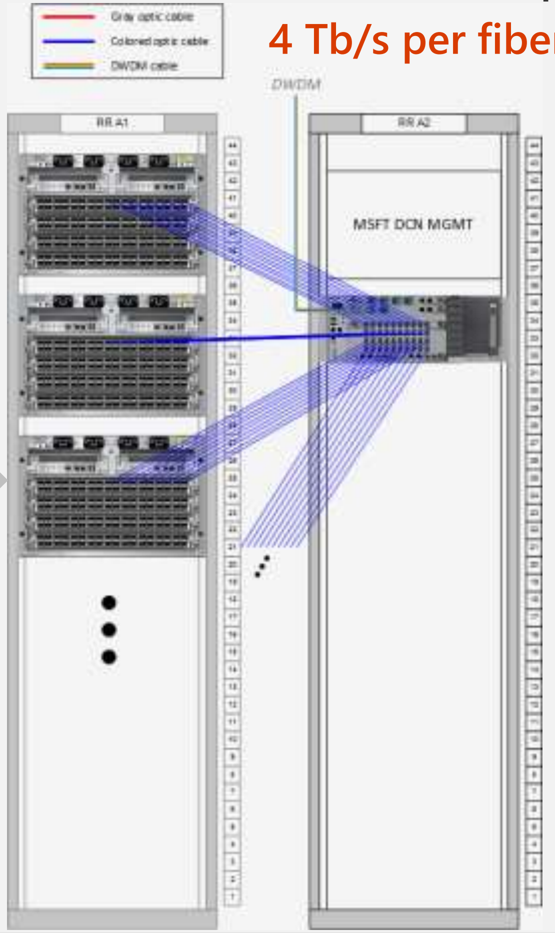
# Madison architecture

Allows us to cost effectively deploy 1.689 Petabits/sec of inter datacenter switching

High-cost high-power Coherent:
8 Tb/s per fiber pair




Low-cost low-power Madison:
4 Tb/s per fiber pair



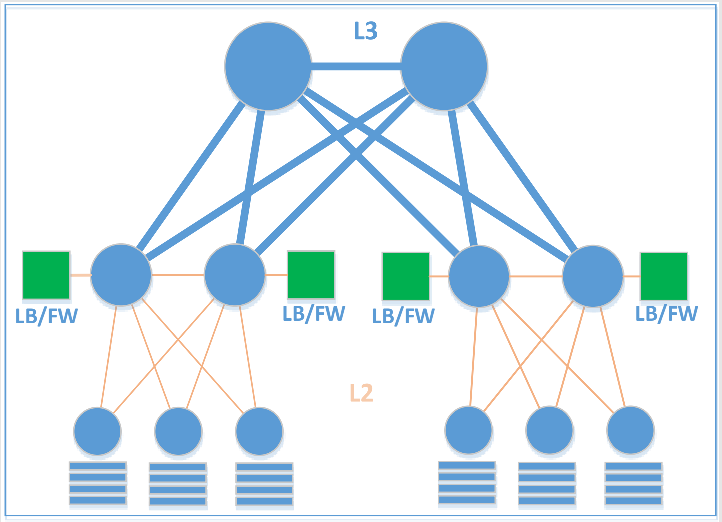
# Azure SDN

# Classic network vs. Hyper-scale network architecture

Large L2 Domains

HW-based service modules

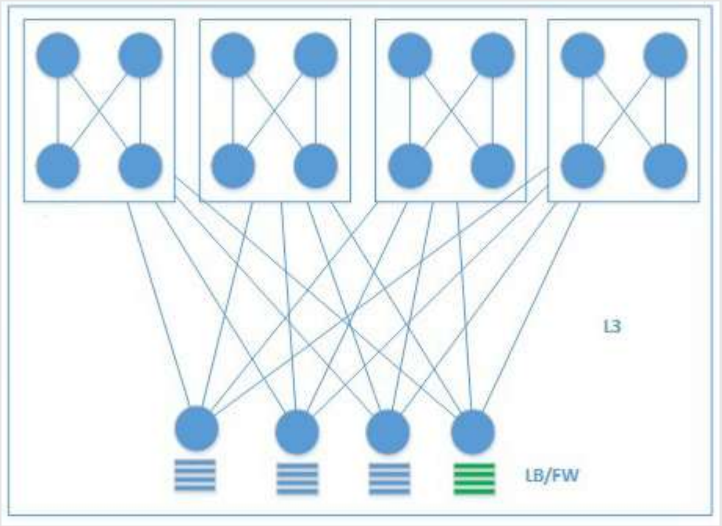
Simple Tree Design




L3 at all layers

Services in software

Clos*-based design




| Classic | | Hyper-scale |
|---|---|---|
| Low due to diversity and manual provisioning process | **Agility** | Automated network provisioning, integrated process |
| Low due to complex hardware and lack of automated operations | **Efficiency** | Simplify requirements, optimize design, and unify infrastructure |
| Low due to high complexity and human error | **Availability** | Resilient design, automated monitoring and remediation, minimum human involvement |

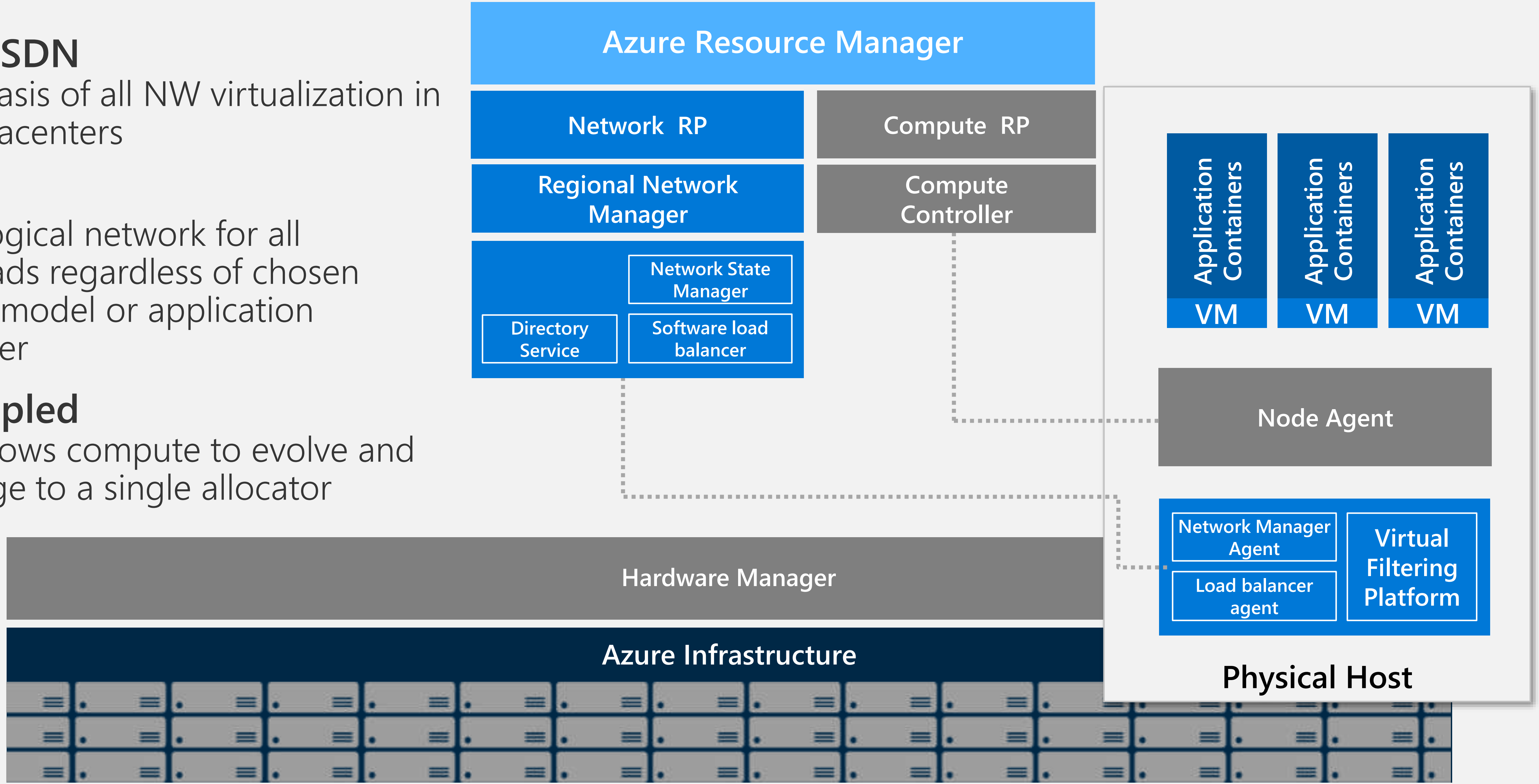
* Charles Clos, 1952

# SDN Logical Components

**Azure SDN**
is the basis of all NW virtualization in our datacenters

**VNet**
is the logical network for all workloads regardless of chosen service model or application container

**Decoupled**
SDN allows compute to evolve and converge to a single allocator



# Azure Network Services

**Virtual NETwork**
(contains subnets, DHCP and DNS)

**NIC**
(owns IPs, is assigned to VNET)

**Load Balancer**
(Internal/External)

**Network Virtual Appliance**
(owns NICs)

**User-Defined Routes**
(applied to VNETs)

**Network Security Group**
(ACL, for NICs or VNETs)

**DNS**
(Private or Public)

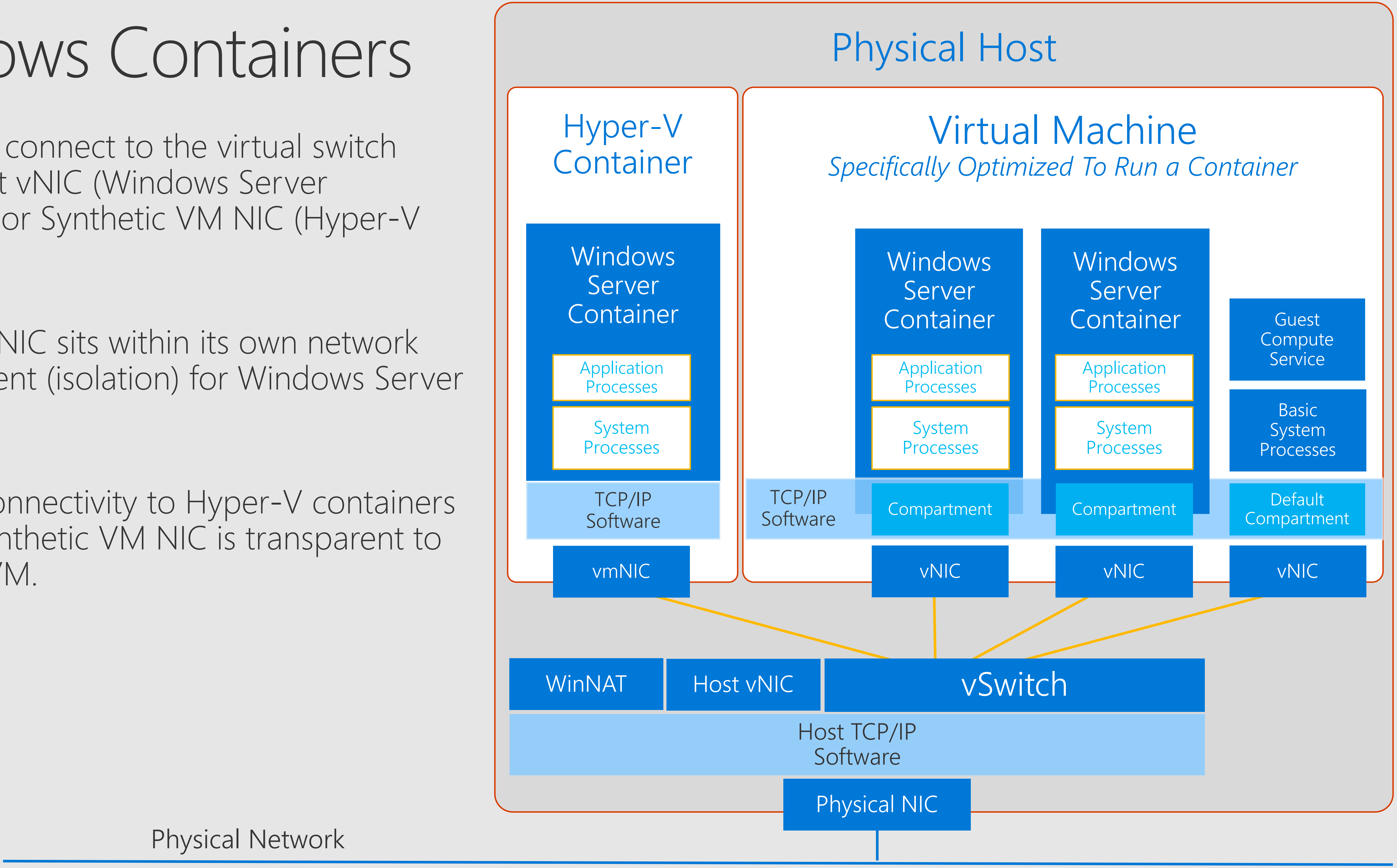
**VPN Gateway**

**Distributed processing, Pure SDN** ⟷ **Guaranteed resources, NFV-like**

# Container Networking

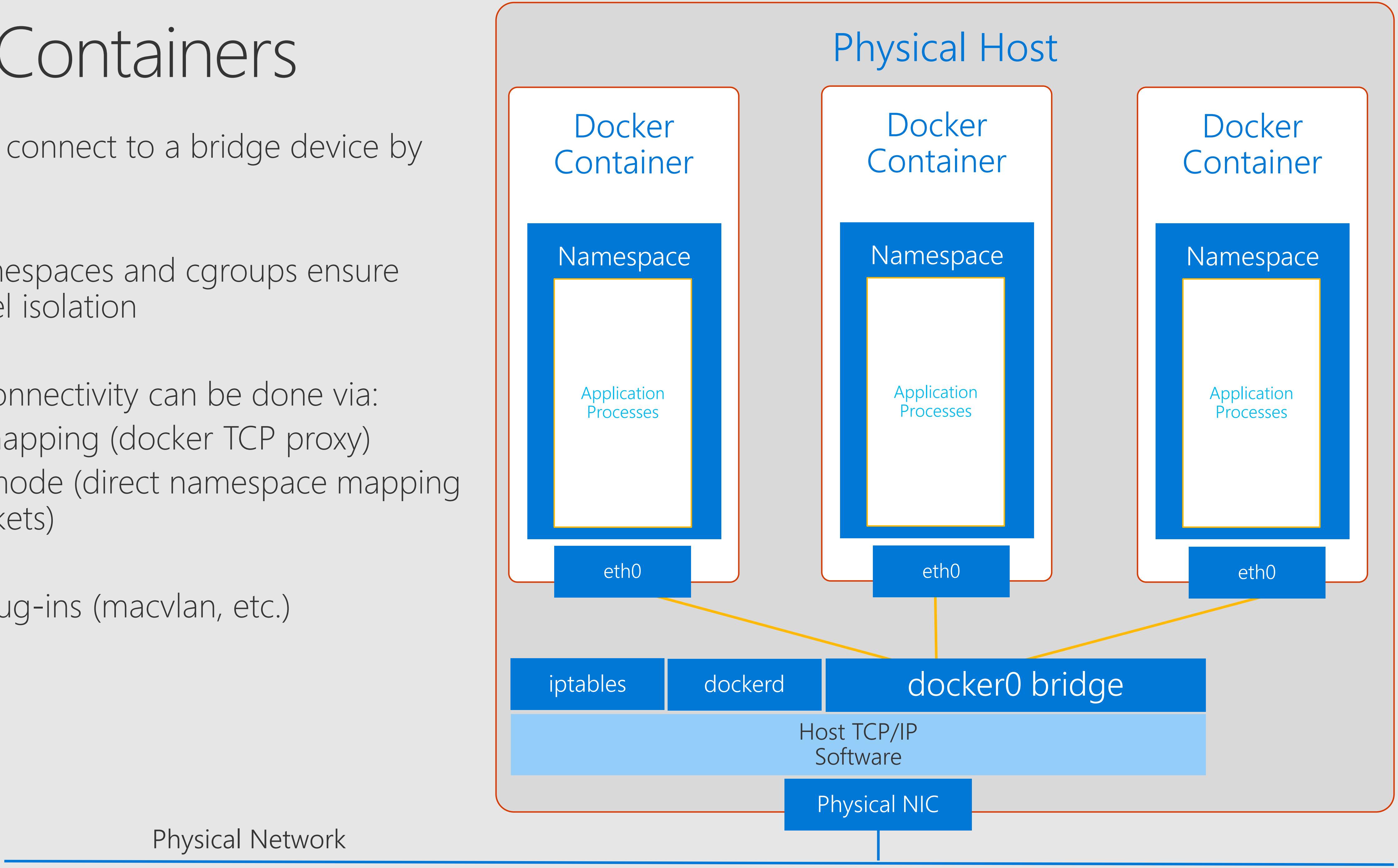
# Windows Containers

- Containers connect to the virtual switch over a Host vNIC (Windows Server Container) or Synthetic VM NIC (Hyper-V Container).
- The Host vNIC sits within its own network compartment (isolation) for Windows Server containers.
- Network connectivity to Hyper-V containers through synthetic VM NIC is transparent to the utility VM.



# Linux Containers

- Containers connect to a bridge device by default
- Kernel namespaces and cgroups ensure device-level isolation
- Network connectivity can be done via:
    - Port mapping (docker TCP proxy)
    - Host mode (direct namespace mapping of sockets)
    - NAT
    - CNI plug-ins (macvlan, etc.)



# Container Networking Challenges

**Performance**

Default Docker networking is slow and introduces 30-70% overhead depending on OS/kernel/versions, due to bottlenecks, repeated transitions between kernel/userspace, etc.

**Transparency**

TCP proxying through dockerd masks origin IP addresses, and NAT/overlay networks introduce further complications (MTUs, IP address allocation, etc.).

**Scalability**

Managing connectivity between multiple hosts via port mapping or NAT is just… insane.

**Orchestration**

Real world deployments require well-defined, open APIs that tie in to orchestrators like Swarm, Mesos and Kubernetes

CNM (Container Network Model)

VS

CNI (Container Network Interface)

IPAM (address management)

CISCO vmware

CLOUD NATIVE COMPUTING FOUNDATION

MESOS

rkt

README.md

# Microsoft Azure Container Networking

## Overview

This repository contains container networking plugins for Linux and Windows containers running on Azure:

- CNM (libnetwork) network and IPAM plugins for Docker Engine.
- CNI network and IPAM plugins for Kubernetes and DC/OS.

The `azure-vnet` network plugins connect containers to your Azure VNET, to take advantage of Azure SDN capabilities. The `azure-vnet-ipam` IPAM plugins provide address management functionality for container IP addresses allocated from Azure VNET address space.

> Azure VNET plugins are currently available as a **public preview**.

The following environments are supported:

- Microsoft Azure: Available in all Azure regions.
- Microsoft Azure Stack: The hybrid cloud platform that enables you to deliver Azure services from your own datacenter.

Plugins are offered as part of Azure Container Service (ACS), as well as for individual Azure IaaS VMs. For ACS clusters created by acs-engine, the deployment and configuration of both plugins on both Linux and Windows nodes is automatic.

## Documentation

See Documentation for more information and examples.

## Build

This repository builds on Windows and Linux. Build plugins directly from the source code for the latest version.

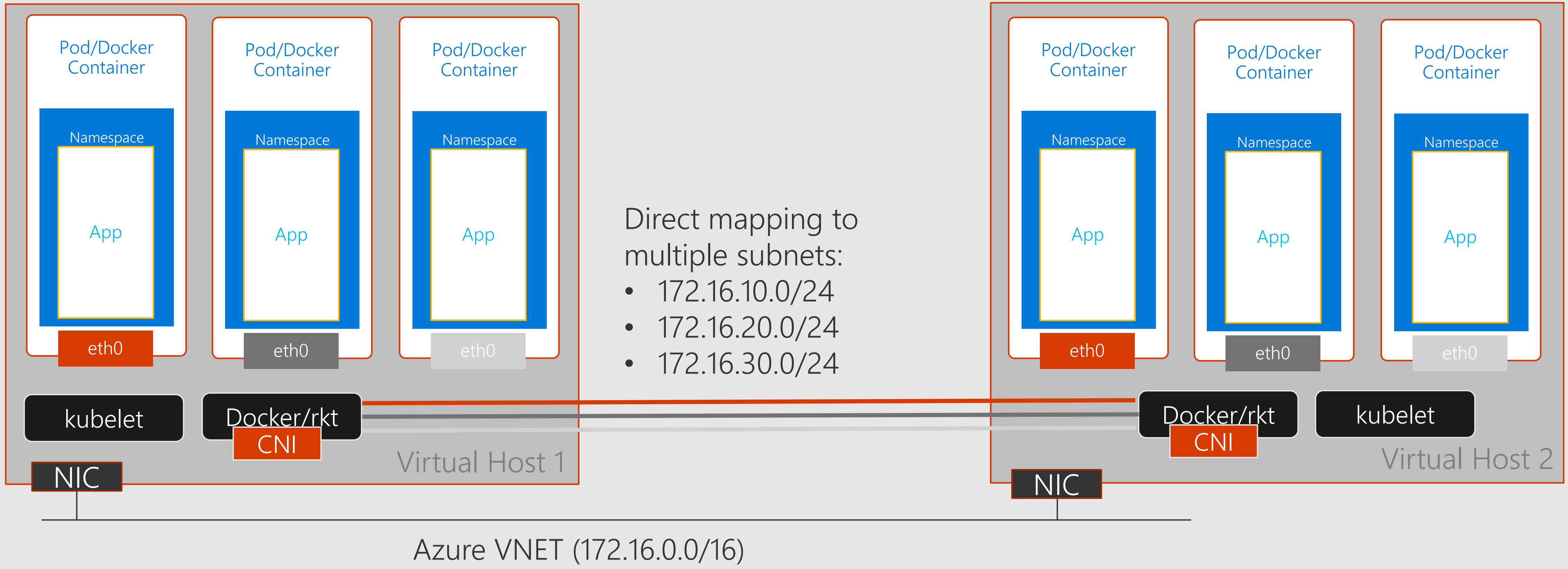
```
$ git clone https://github.com/Azure/azure-container-networking
$ cd azure-container-networking
$ make all-binaries
```

Then foll https://github.com/Azure/azure-container-networking

# CNI/IPAM on Azure

Direct mapping to multiple subnets:

- 172.16.10.0/24
- 172.16.20.0/24
- 172.16.30.0/24



# Looking Forward

# Host SDN scale challenges

**Hosts are Scaling Up:**
**1G → 10G → 40G → 50G → 100G → …?**

Reduces COGS of VMs (more VMs per host) and enables new workloads

Need the performance of hardware to implement policy without CPU

**Need to support new scenarios:**
**BYO IP, BYO Topology, BYO Appliance**

We are always pushing richer semantics to virtual networks

Need the programmability of software to be agile and future-proof

> How do we get the performance of hardware with programmability of software?

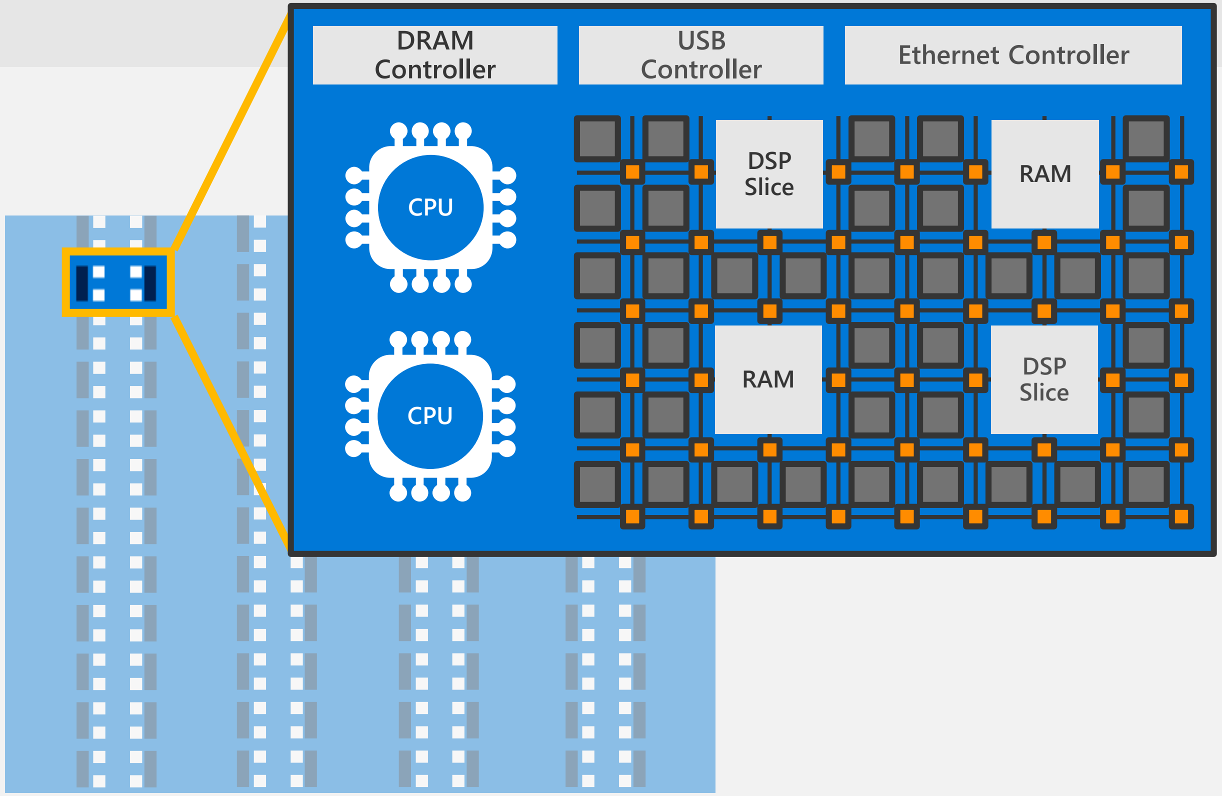
# FPGA | Field Programmable Gate Array

Programmable hardware

Chip has large quantities of programmable units

Program specialized circuits that communicate directly

FPGA chips are now large SoCs



# Azure SmartNIC | Accelerated Networking

**Use an FPGA for reconfigurable functions**

Roll out Hardware features as we do software

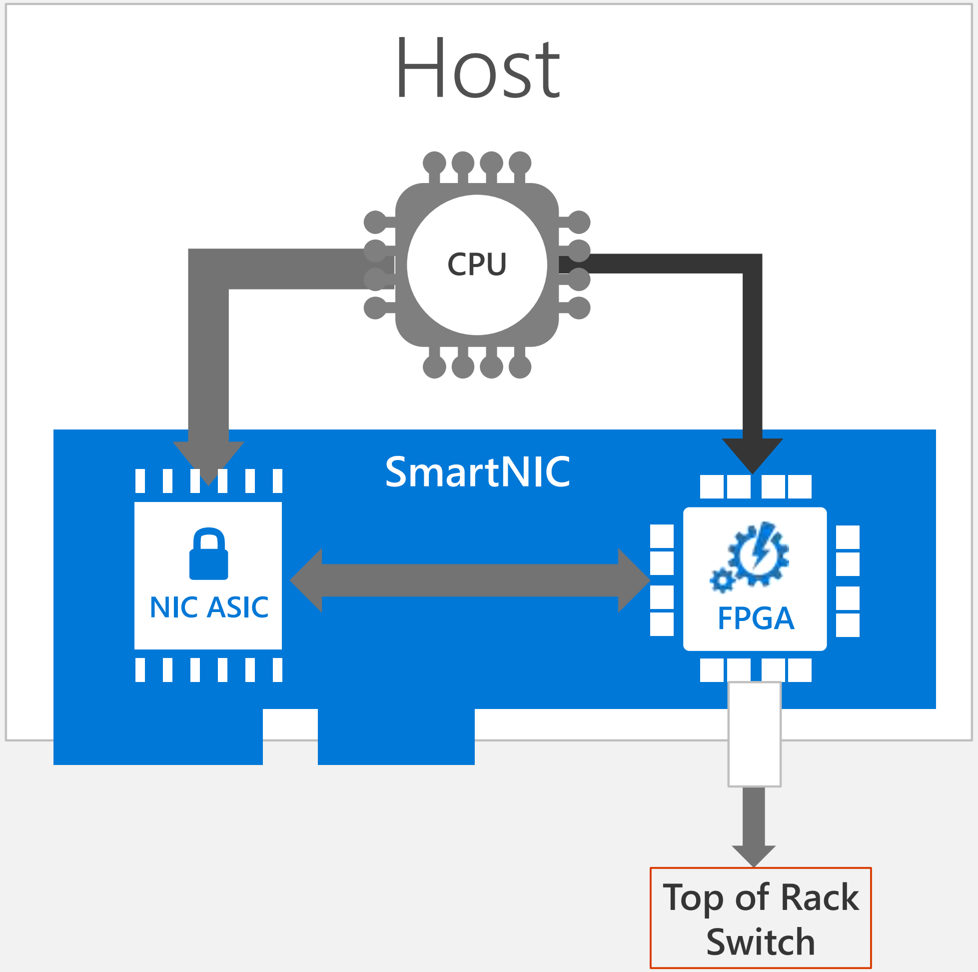
**Programmed using Generic Flow Tables (GFT)**

Language for programming SDN to hardware

Uses connections and structured actions as primitives

**Deployed on all new Azure compute servers since late 2015**

**SmartNIC is also doing Crypto, QoS, storage acceleration, and more…**



# Container Networking Challenges (Revisited)

**Performance**

Azure Kubernetes Service (AKS) and Azure Container Instance (ACI) already use CNI and IPAM by default

**Transparency**

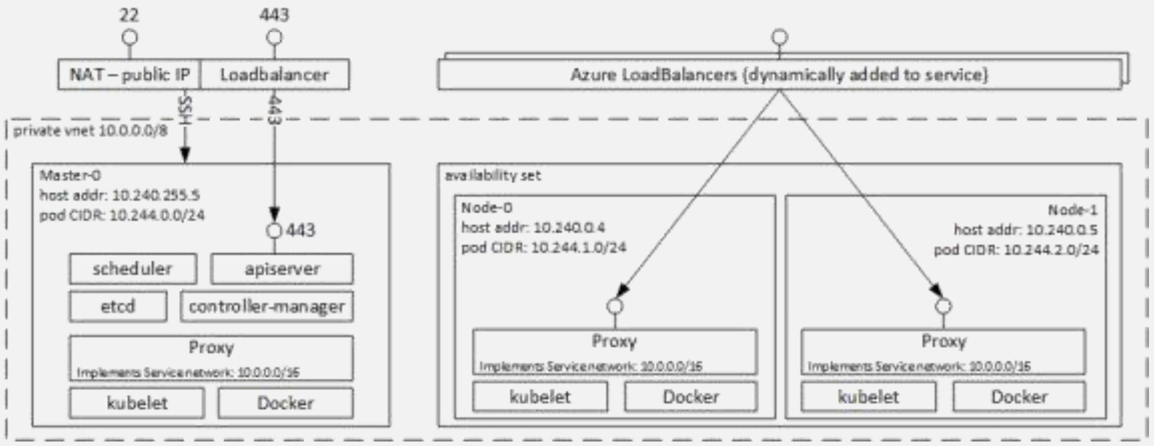
Containers as first-class SDN citizens (already 2/3 of the way there)

**Scalability**

Kubernetes DNS/IPv6 for service discovery/connectivity across datacenter regions (already possible via VNET peering, we want to make it simpler as K8s evolves)

**Orchestration**

Full integration with Azure Network Resource Provider/SDN management through Kubernetes network policy APIs



Microsoft

# Questions?

Microsoft

# Thank You