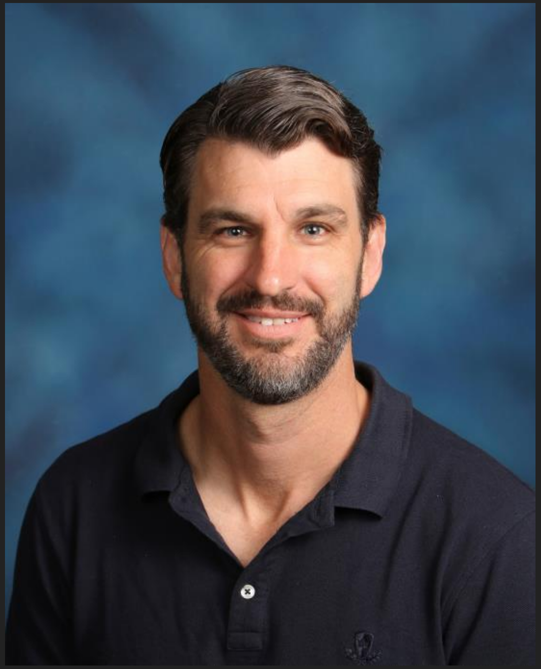
CNF Architectures for Edge and Dataplane

Jarrod Lucia F5 Inc

Innovation Architect | Service Provider 5G Telco and Cloud | APCJ Regional Lead



I NEED TO KNOW WHY MOVING OUR APP TO THE CLOUD DIDN'T AUTOMATICALLY SOLVE ALL OUR PROBLEMS.
Dilbert.com @ScottAdamsSays
YOU WOULDN'T LET ME RE-ARCHITECT THE APP TO BE CLOUD-NATIVE.
JUST PUT IT IN CONTAINERS.
11-08-17 © 2017 Scott Adams, Inc./Dist. by Andrews McMeel
YOU CAN'T SOLVE A PROBLEM JUST BY SAYING TECHY THINGS.
KUBERNETES.

Today’s Challenges for Telco and Edge Deployments

INCREASING TRAFFIC AND COMPLEXITY OF NETWORKS BREAKS EXISTING MODELS

Increasing CapEx and OpEX

- Need higher network capacity at lower cost
- Need to increase revenue from 5G services and new edge opportunities

Scaling and automating functions

- Digital transformation needs to be simplified
- Deployment and upgrade cycles too slow

Maintaining and improving network security

- Larger threat surface
- Increasing DDoS attacks

Definition of Cloud Native

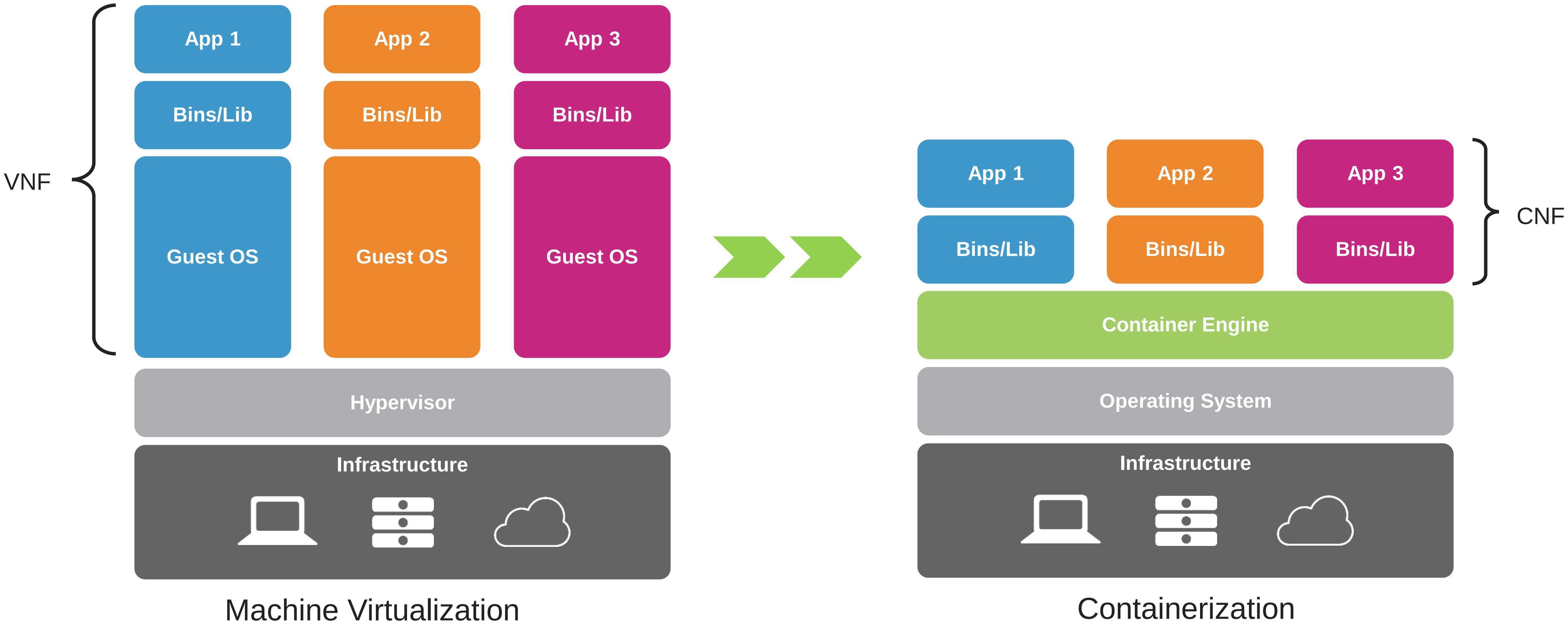
KEY ASPECTS OF BEING CLOUD NATIVE

Agility

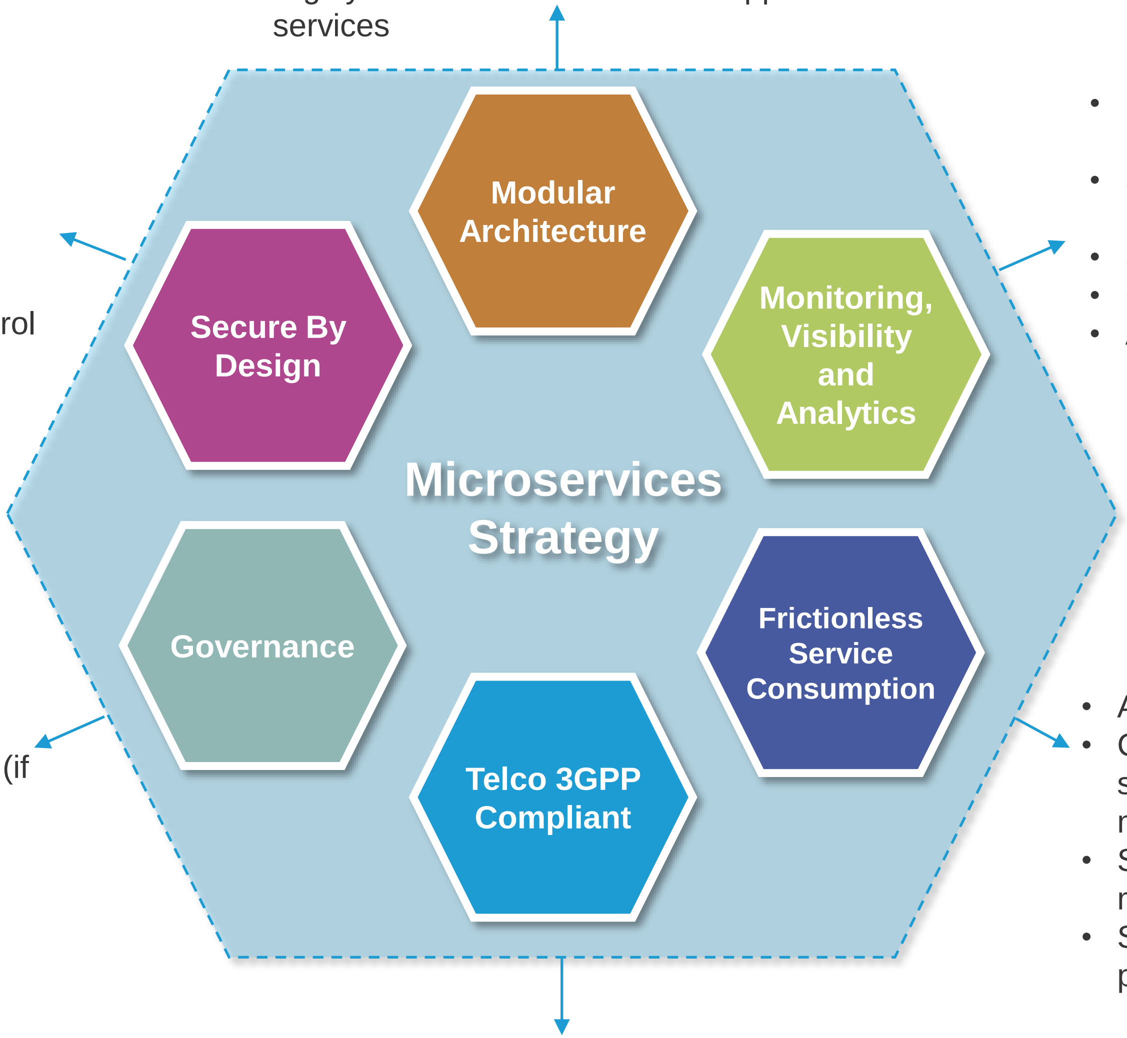
Resilience

Observability

Transformation to Containerization



Key Composition of a Microservices Strategy



- Portable and platform vendor independent
- Agile and scalable architecture – Works everywhere, infrastructure independent
- Environment elasticity
- API driven ecosystem
- Run in consistent and predictable manner
- Highly resilient architecture and application services

- Embed Security control in the beginning
- Secure image registries
  - Trusted signed images
  - Container Vulnerability Scan
  - Scan for unintentional private embedded data.
- Identity and access management
  - mTLS
  - Authentication Proxy
  - OpenID Connect
- RBAC – Role-based Access Control
  - Least privilege access model

- Insight on apps utilization & performance
- SLA and resiliency and availability monitoring
- Service oriented monitoring
- Cost consumption monitoring
- Application performance (APM) monitoring

- Standardization
- Tooling and language framework
- Deployment pipeline
- Alignment with standards bodies (if possible – e.g. OCI, CNCF)
- Automation
- Consistent deployment patterns
- API definition and strategy
- Monitoring (e.g. metrics and dashboard)

- API driven ecosystem
- Consumable full application services (e.g. traffic management and security)
- Support Blue/Green deployment model
- Seamless integration into CI/CD pipeline

- Organization structure alignment

Cloud-Native is the evolution of Network Functions delivery

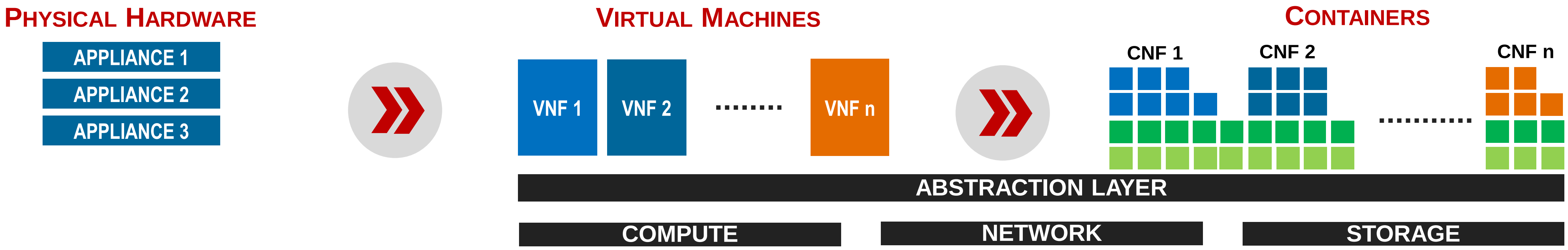
LONG TERM EVOLUTION TO IMPROVE OPERATIONS

Network Functions are elements that enrich network traffic (CGNAT, Firewall, DNS Cache, etc.)

- **1980s - 2000s**
- **Physical Network Functions**
- Network Functions implemented as hardware appliances & chassis with tightly coupled software

- **2010s**
- **Virtual Network Functions**
- Network Functions implemented as virtual machines running on Networks Functions Virtualization platforms

- **2020s - Future**
- **Cloud-Native Network Functions**
- Network Functions implemented as containers that can be controlled by Kubernetes



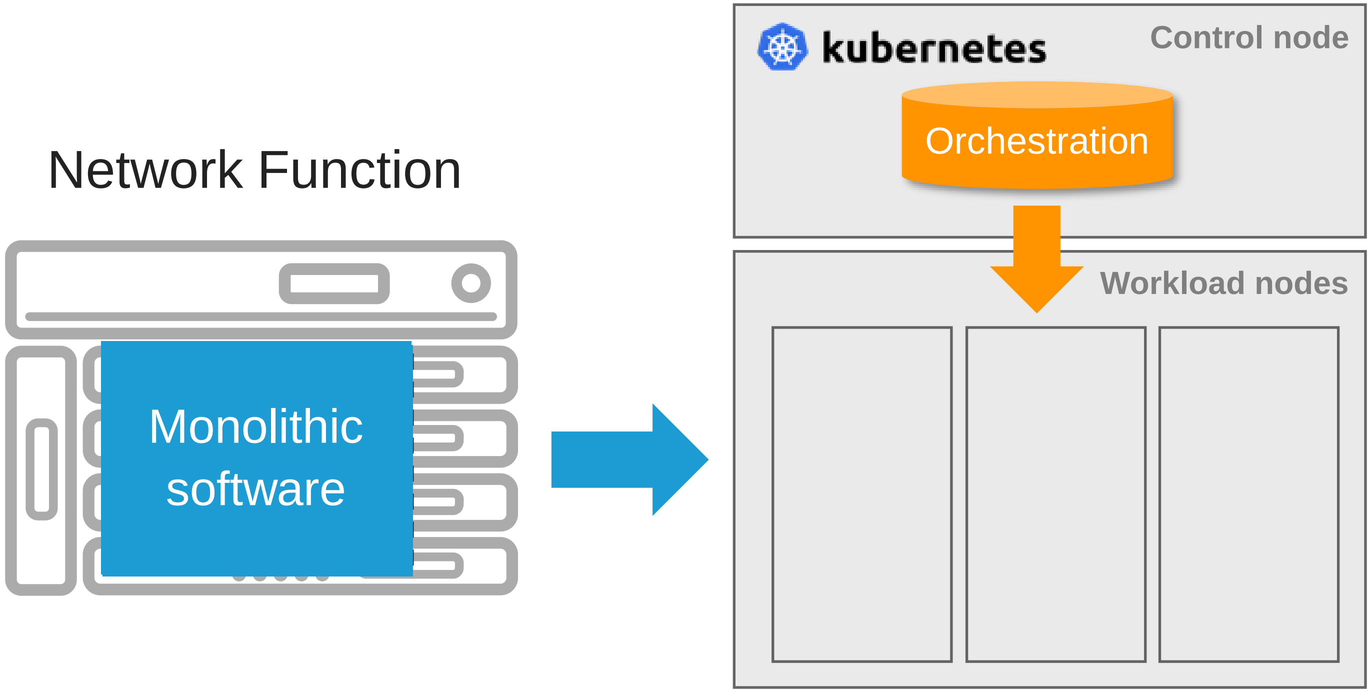
Comparison of Form Factor Architectures

WHERE ARE THE BENEFITS OF CLOUD-NATIVE OVER EXISTING ARCHITECTURES?

	Physical	Virtualised	Cloud-Native
Network performance	Dedicated performance enhanced with ASIC and FPGA technolgoies	Raw throughput typically limited by hypervisor performance	Raw throughput typically limited by commercial CPU performance
Resource utilisation	Dedicated Network Resource	Resource over-provisioning not desirable for telco workloads	Resource utilization based on available scheduling
Deployment speed	Deployment takes days - weeks	Deployment takes minutes	Deployment takes seconds
Automation	Proprietary and fragmented automation	Automation is complex with multiple interacting control planes	Kubernetes provides a common, consistent control plane

Creating a Cloud-Native Network Function

JUST PORTING EXISTING ARCHITECTURES HAS LIMITED BENEFITS

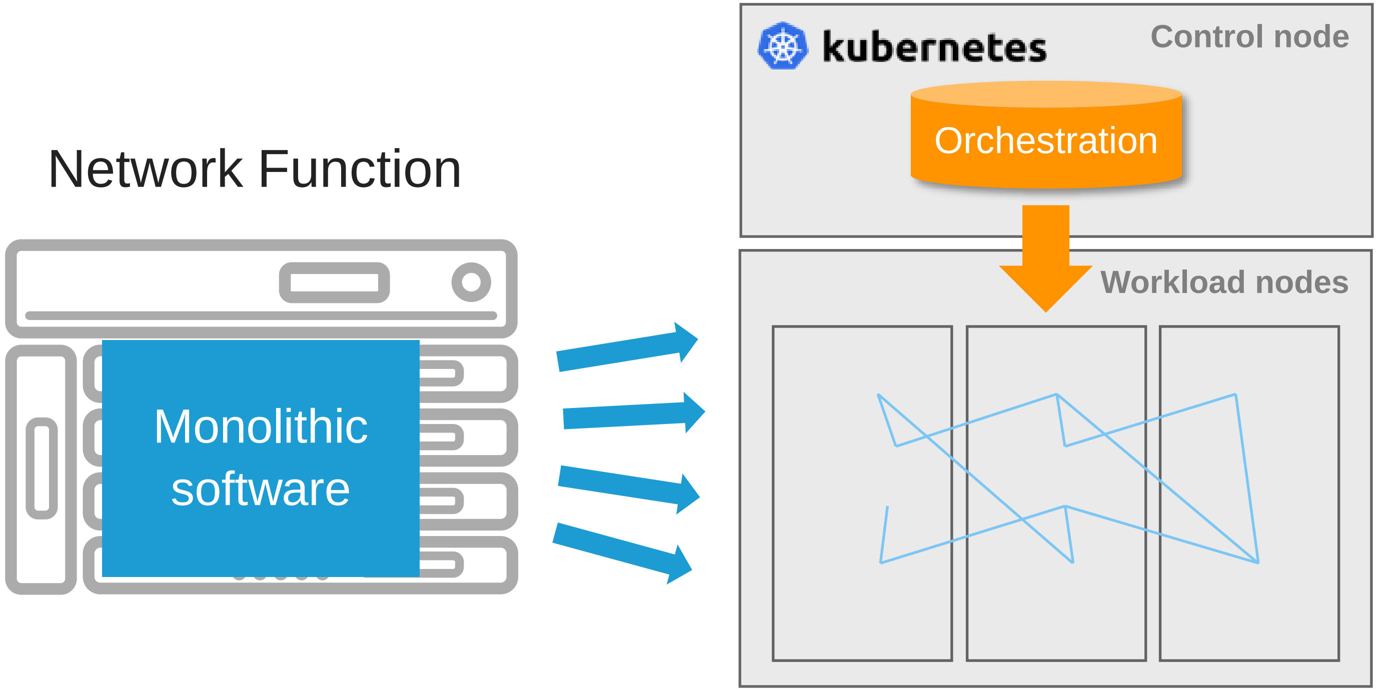


Monolithic CNF

- ✓ Orchestrated through Kubernetes control plane
- ✗ Slow to deploy
- ✗ Difficult to scale data plane
- ✗ Complex to upgrade and enhance
- ✗ Inefficient use of compute resources

Creating a True Cloud-Native Network Function

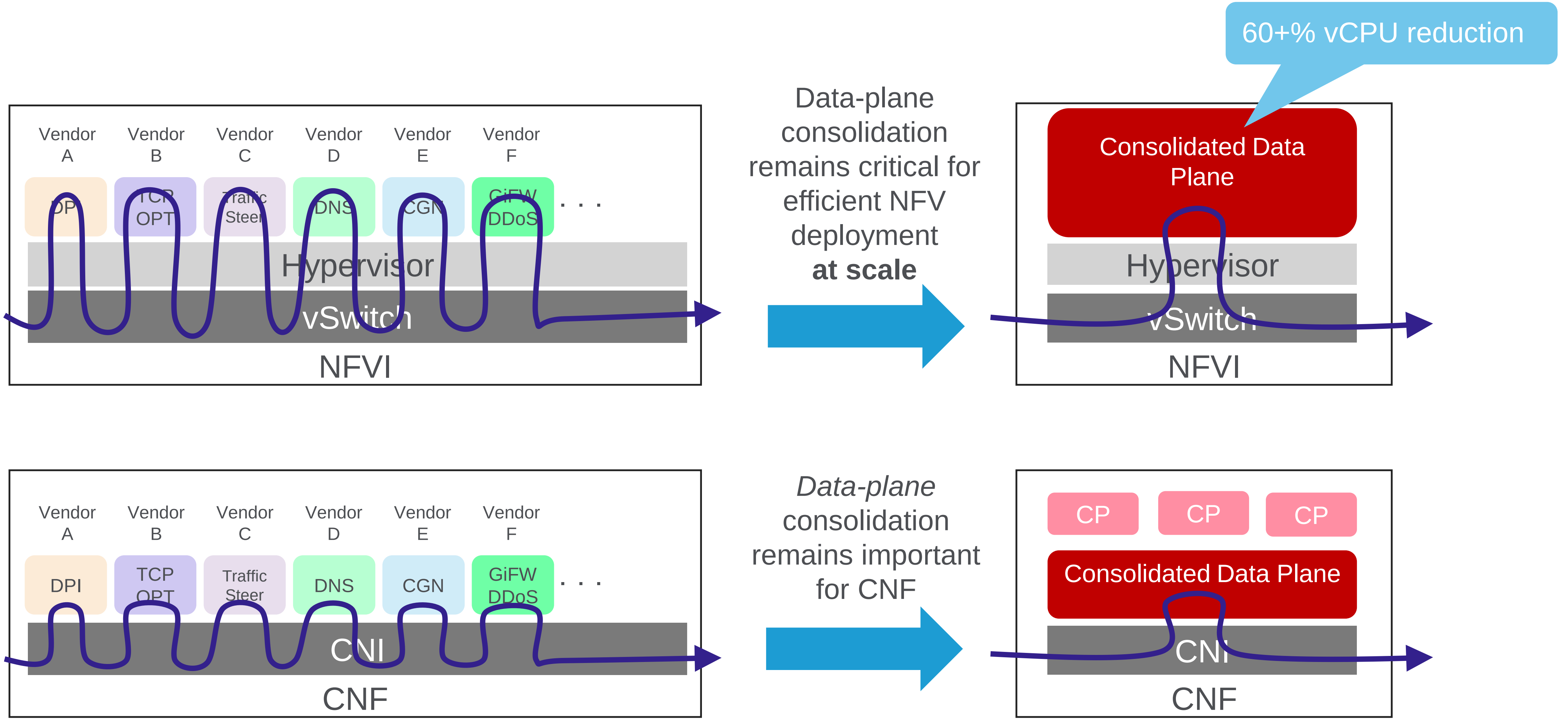
DECOMPOSING THE TRADITIONAL NETWORK FUNCTIONS MONOLITH



Microservices CNF

- ✓ Orchestrated through Kubernetes control plane
- ✓ Fast scaling of stateless dataplane
- ✓ Distributed state supports scaling and resilience
- ✓ Control functions rapidly added and upgraded
- ✓ Lightweight, efficient components

Service Chaining in the Data Plane

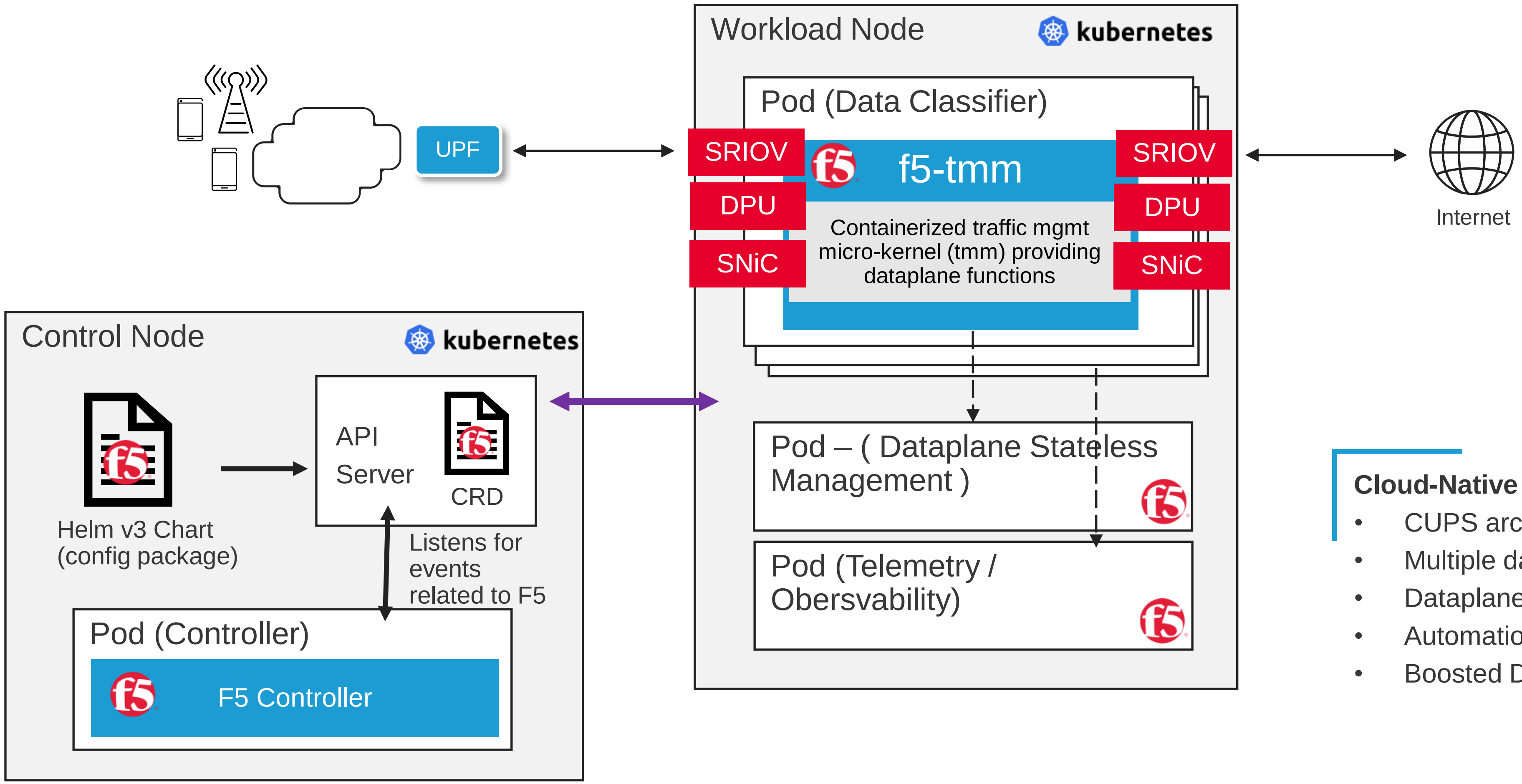


60+% vCPU reduction

Data-plane consolidation remains critical for efficient NFV deployment **at scale**

Data-plane consolidation remains important for CNF

F5 CNF Architecture Overview



Cloud-Native Functions

- CUPS architecture
- Multiple dataplane pods for increased scale
- Dataplane Stateless management
- Automation / Kubernetes Control Plane
- Boosted Dataplane Performance Capability

CNF Key Characteristics

BEST OF SUITE CONSOLIDATED CLOUD-NATIVE NETWORK FUNCTIONS

- Control Plane – User / Data Plane Separation
- Containerized micro-services that communicate via APIs
- Small footprint with horizontal scaling

CNF Key Characteristics

BEST OF SUITE CONSOLIDATED CLOUD-NATIVE NETWORK FUNCTIONS

- **Orchestrated by Kubernetes API**
- **Integrations with popular Kubernetes distributions**
- **Boosted performance - hardware offload (DPU / FPGA / SmartNIC)**