Fabric & Controllers Track Overview

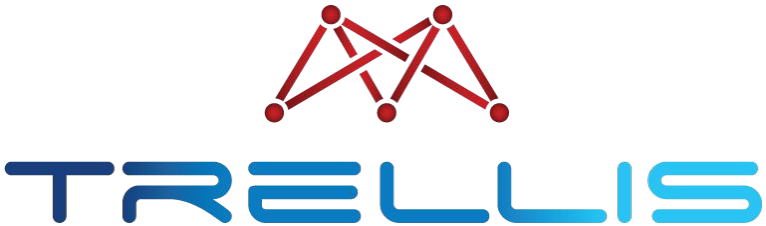
Co-Chair: Charles Chan (ONF)
Vignesh Ramamurthy (Infosys)

Track Overview (Day 1)

- 4:30-4:40 **Track overview and Trellis introduction**
  - Charles Chan (ONF)
- 4:40-4:45 **Trellis contributor award**
  - Saurav Das (ONF)
- 4:45-5:30 **Integrating Trellis into a carrier-grade NFV platform**
  - Subramanya Datta, Vignesh Ramamurthy (Infosys)
- 5:30-6:00 **One size does not fit all – Tungsten Fabric as an enabler of intent-based security in diverse multicloud architectures**
  - Richard Roberts, Ato Sanchez-Monge (Juniper)
- 6:00-6:30 **Transforming networks with ONF support**
  - Metin Balci (ULAK)

Track Overview (Day 2)

- 2:00-3:00 **Tuning and Hardening Trellis for Large-scale Deployment**
  - Hariprasad Rajendran (Infosys)
- 3:00-3:30 **VNFs in CNFs Environment**
  - Monika Antoniak, Piotr Skamruk (CodiLime)
- 3:30-4:00 **What Can SDN Do For NFV Cloud Network**
  - Ruixue Wang (China Mobile)

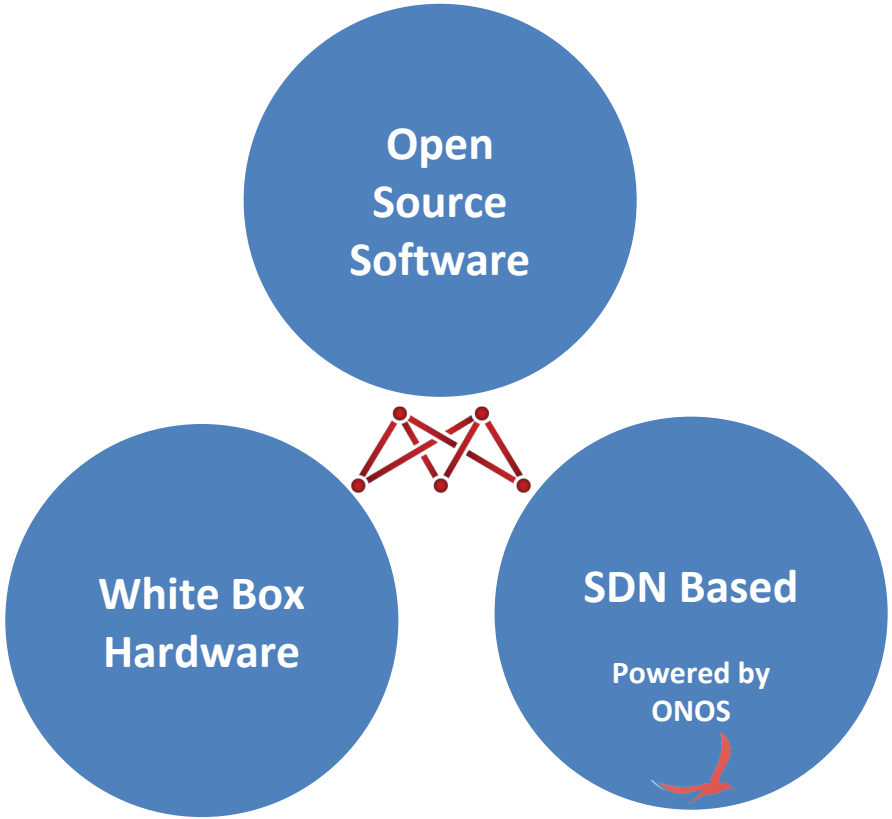


Trellis Introduction

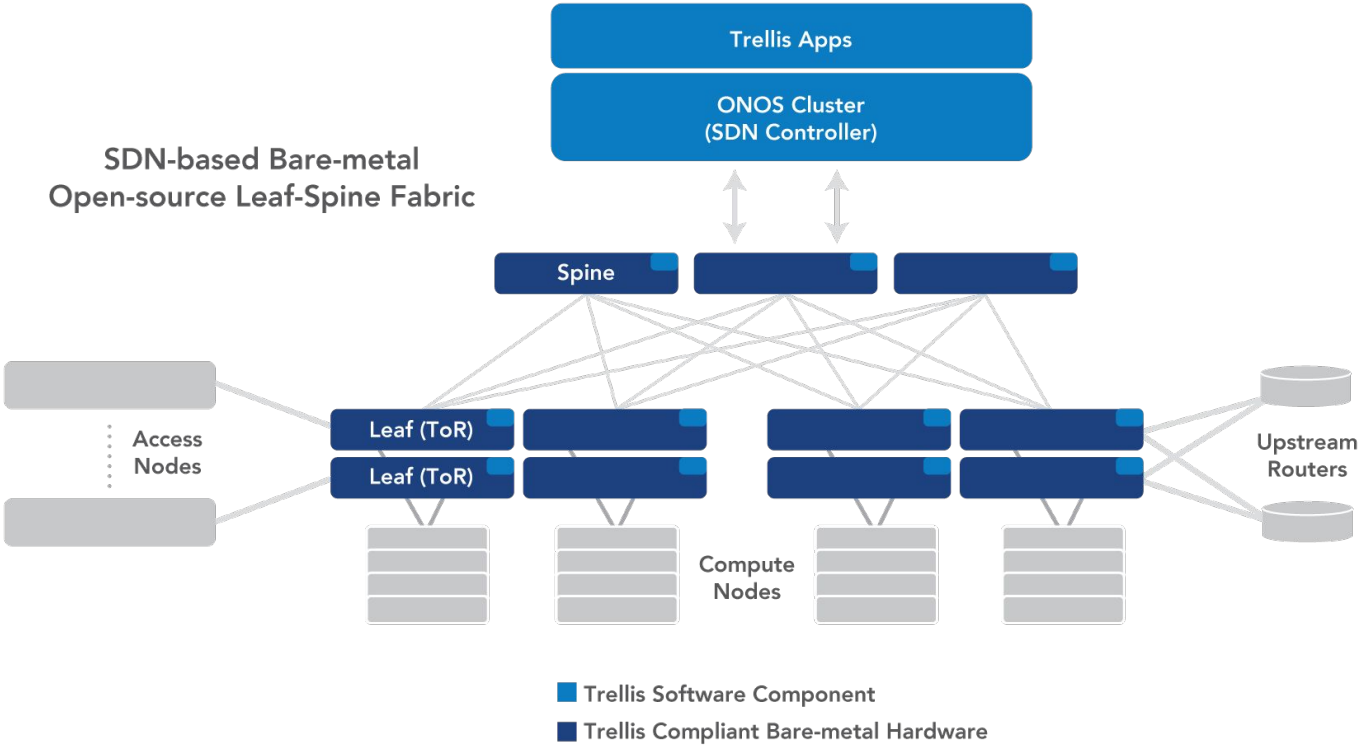
Charles Chan, Ph.D.
Member of Technical Staff, ONF

Trellis

production-ready
multi-purpose leaf-spine fabric



Trellis


Trellis Apps
ONOS Cluster
(SDN Controller)
SDN-based Bare-metal
Open-source Leaf-Spine Fabric
Spine
Access
Nodes
Leaf (ToR)
Leaf (ToR)
Compute
Nodes
Upstream
Routers
Trellis Software Component
Trellis Compliant Bare-metal Hardware

Why Trellis?

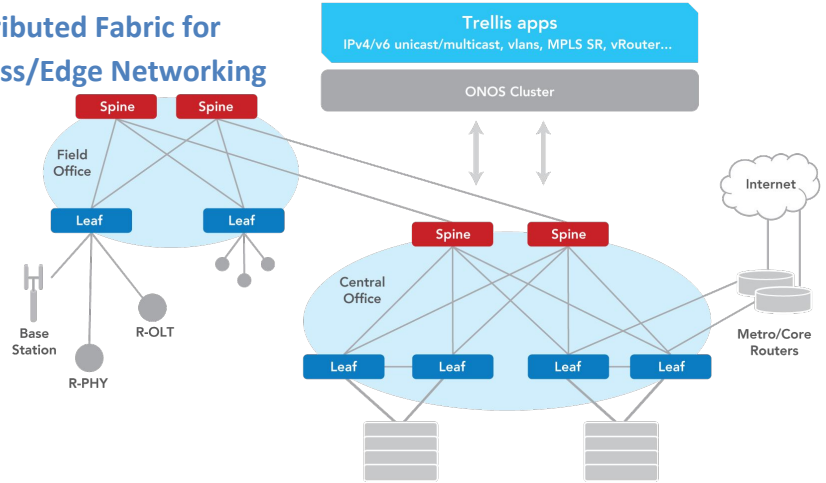
- Trellis is designed for service provider edge
  - Traffic types/encapsulations, topologies, ASICs
- SDN simplifies and optimizes existing features
  - Learn more at Trellis booth
- SDN & P4 switches enable new features
  - Learn more at SEBA BNG booth
- Open-source -> ownership & customizability
  - Learn more at Comcast booth

TRELLIS

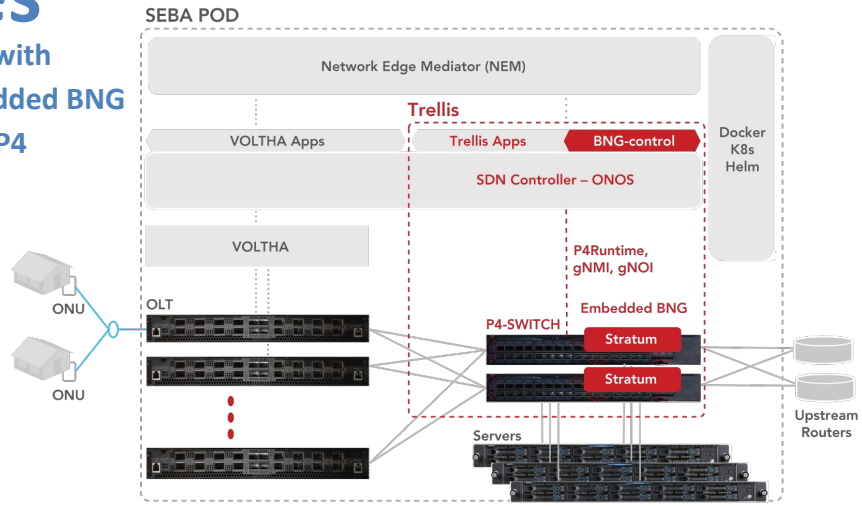
and more…

Use Cases

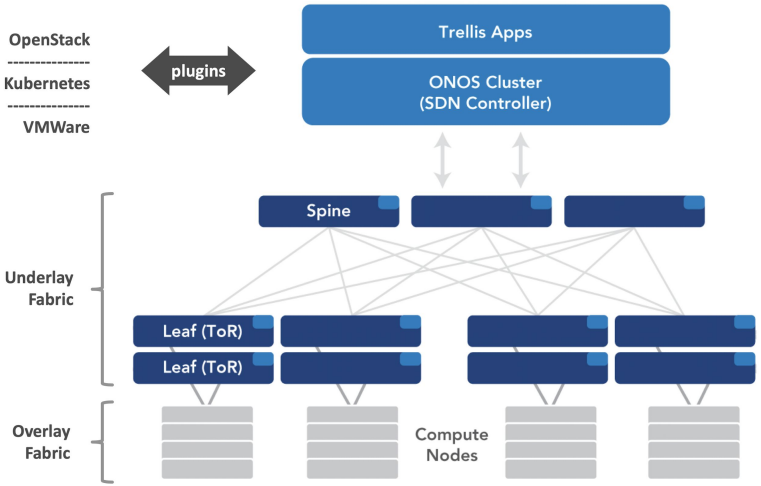
Distributed Fabric for Access/Edge Networking



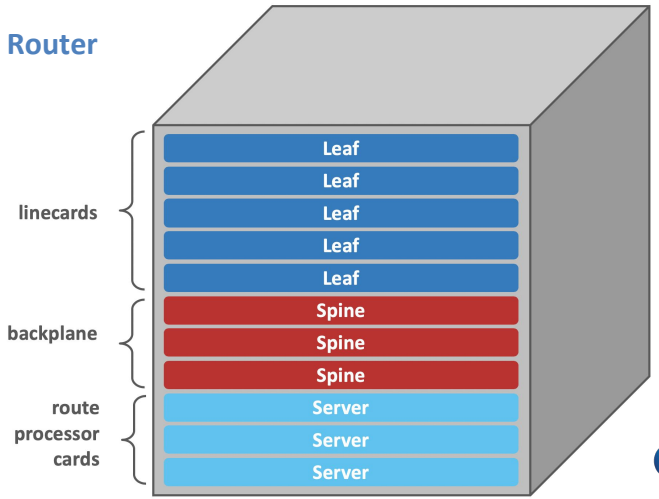
SEBA with Embedded BNG using P4



Enterprise DC Fabric



Chassis Router



Looking Forward

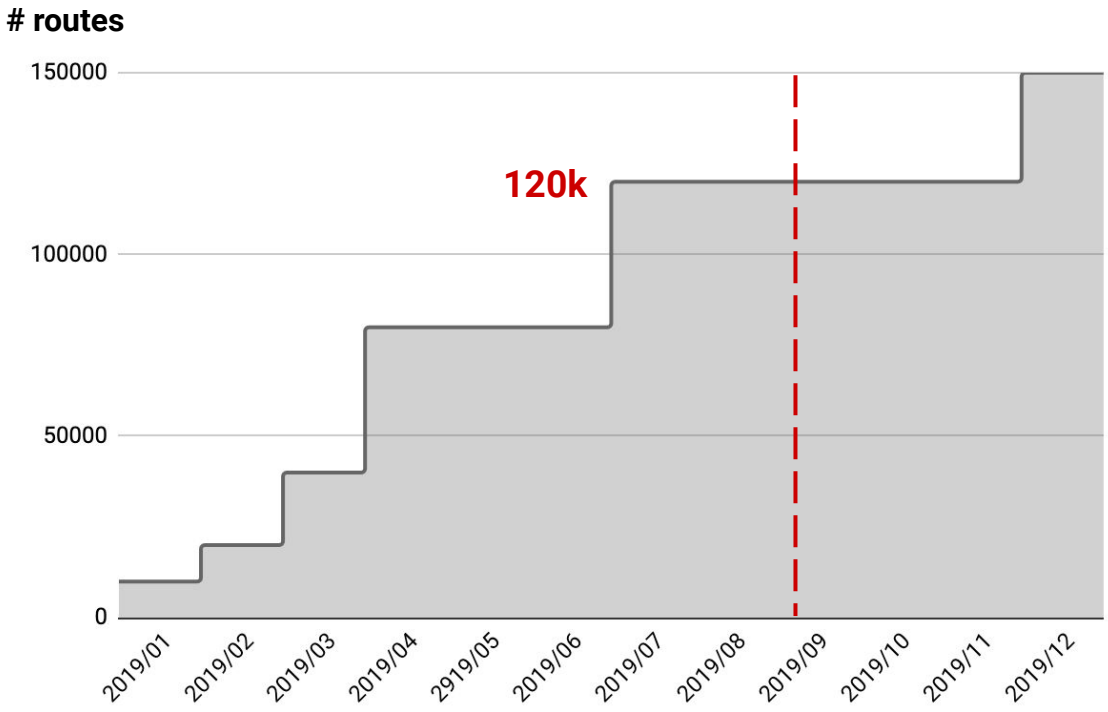
- (Ongoing) Scale & Performance improvements
- (Ongoing) Hardening
- (Ongoing) Stratum/P4 integration
- (Ongoing) BNG features (e.g. PPPoE termination, hierachical QoS)

- Dual homing for Access nodes (like OLTs)
- In Service Software Upgrades (ISSU)
- 5G user plane features

Get Involved

- Website: https://opennetworking.org/trellis
- Mailing list: trellis-dev@opennetworking.org
- Slack: #Trellis on onosproject.slack.com

Summary (oversimplified)



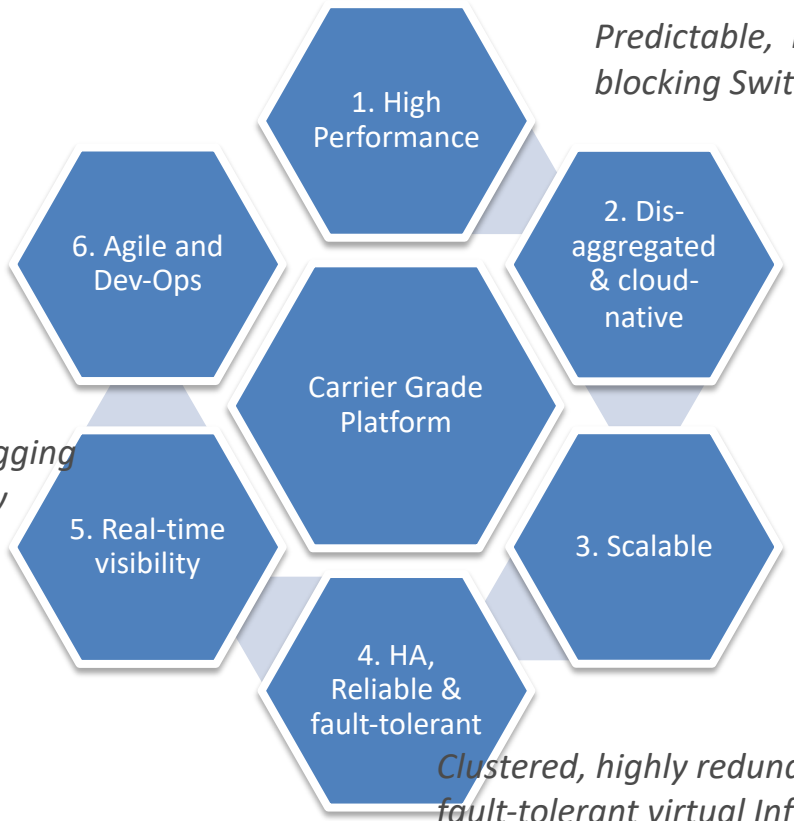
- 12 lessons learnt from production deployments

Trellis:
Integrating Trellis into a Carrier grade NFV Platform

Vignesh Ramamurthy
Subramanya Datta G
Balaji Thangavelu
INFOSYS

Trellis: Carrier grade NFV platform

What is carrier grade in the new paradigm of SDN/NFV ?



Predictable, High performance non-blocking Switching infrastructure

Fully dis-aggregated, virtualized and de-coupled control Plane

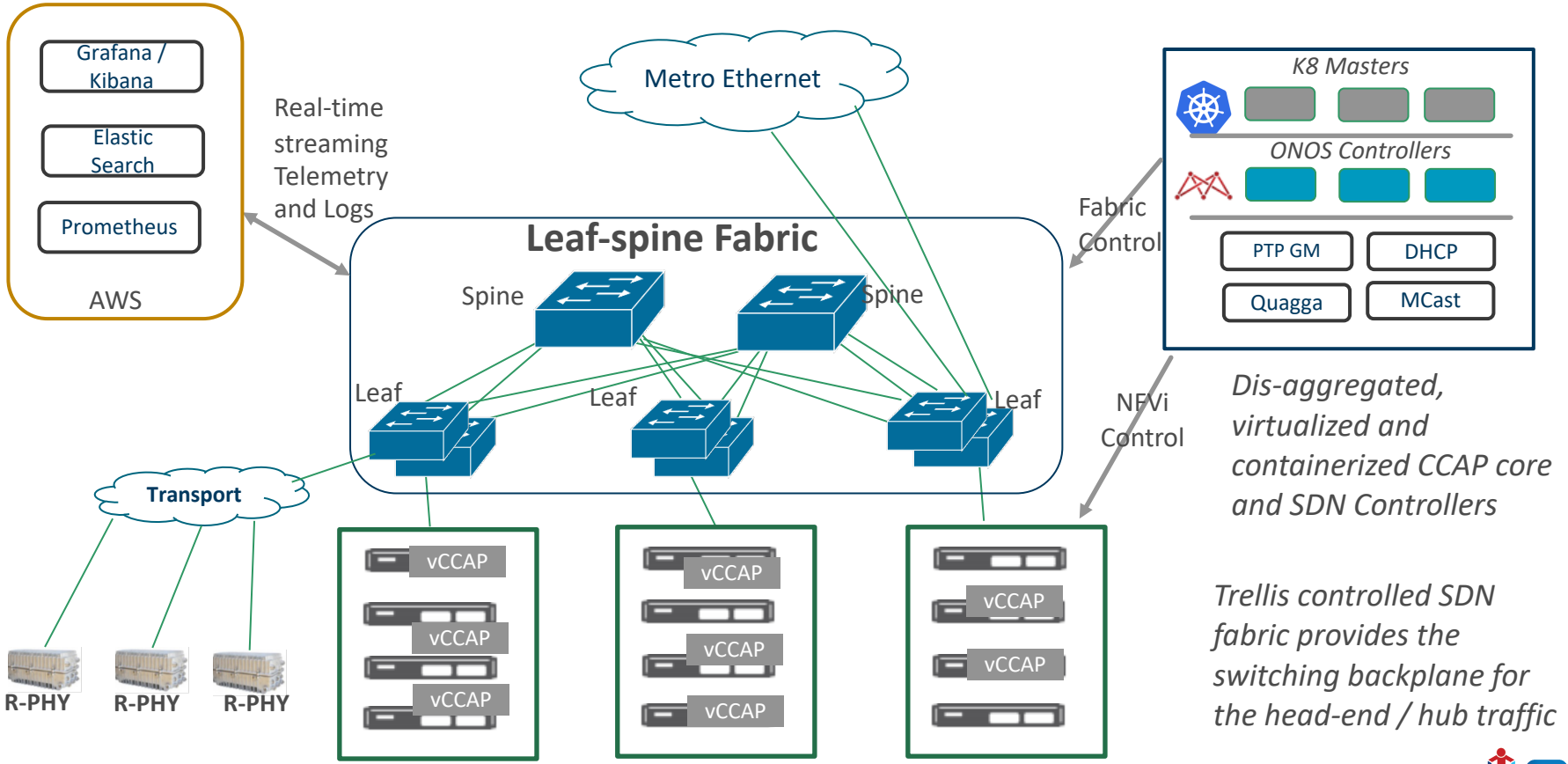
Cloud-style scaling of Control Plane functions

Clustered, highly redundant and fault-tolerant virtual Infrastructure

Real-time streaming and Logging pipeline giving deep visibility

High velocity software delivery and upgrades

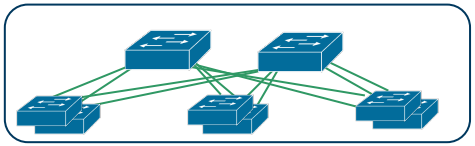
Trellis as NFV Fabric for Virtualized Cable Head-end



Dis-aggregated, virtualized and containerized CCAP core and SDN Controllers

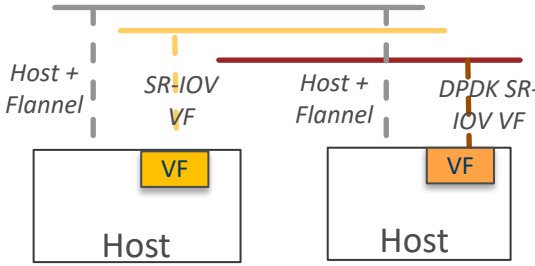
Trellis controlled SDN fabric provides the switching backplane for the head-end / hub traffic

Trellis as NFV Fabric for Virtualized Cable Head-end : Networking characteristics



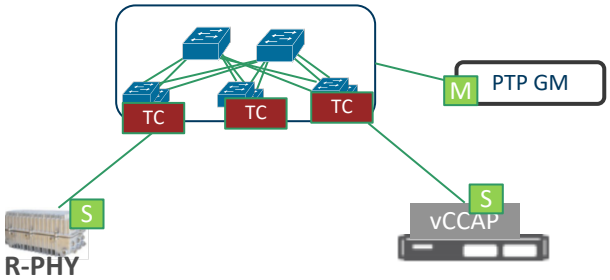
Fabric Underlay Networking:

- *MPLS Segment Routing based forwarding across the fabric*
- *IPv6 and IPv4 unicast, L2 bridging and Multi-casting*
- *MAC+VLAN based host recognition and management*
- *VLAN cross-connect feature*



Container and Overlay Networking:

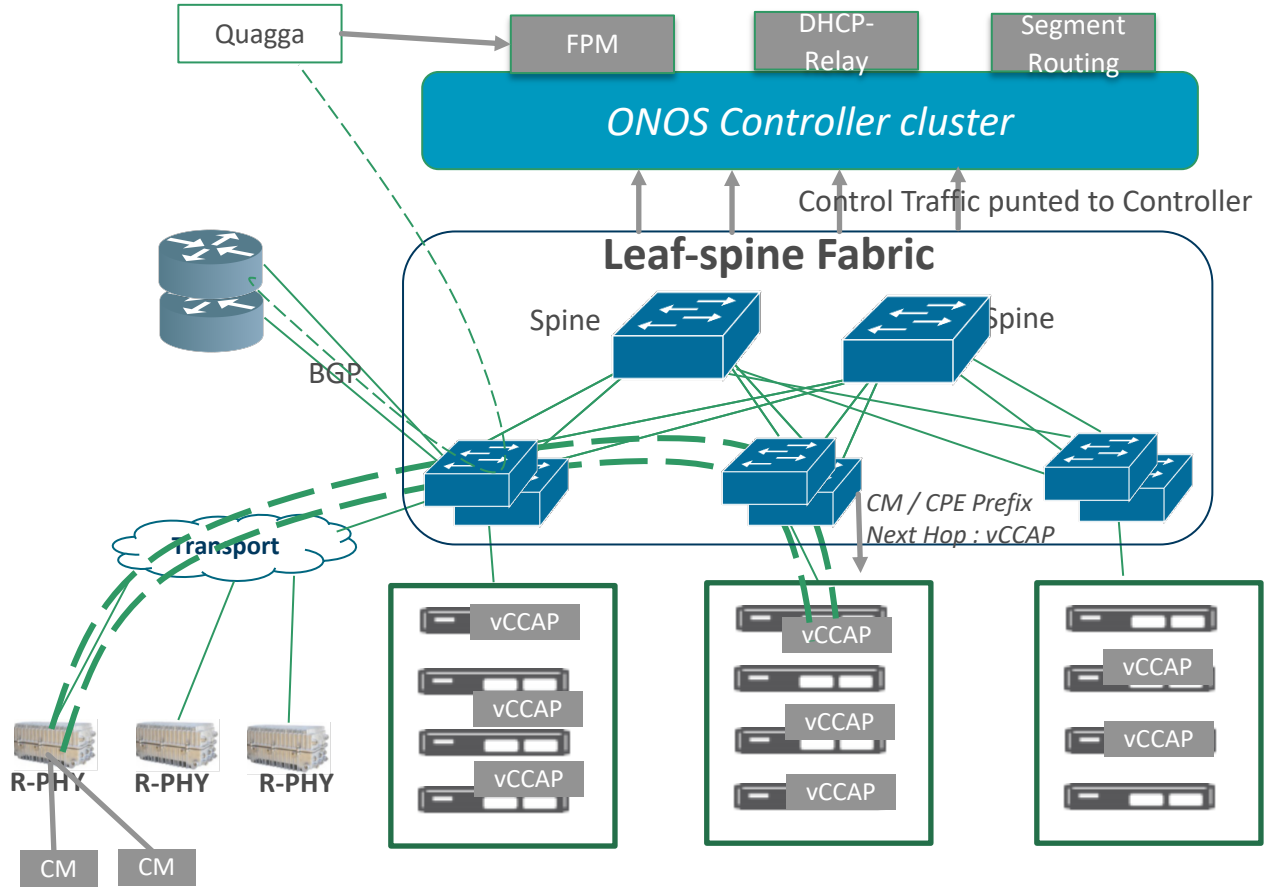
- *Multiple networks – L2 VLAN based bridging within a leaf switch pair*
- *Kubernetes Container networking – VXLAN networking via Host CNI*
- *SR-IOV based VFs managed as a separate L2 VLAN networks*
- *SR-IOV DPDK used for fast data-path networking for DOCSIS user-plane*
- *Container end-points managed in the underlay*



PTP 1588v2 Timer:

- *PTP 1588 based Timing / Phase synchronization*
- *Prioritized Forwarding flows for 1588 packets – Unicast UDP packets*
- *Peer to Peer Transparent clocks configured on the Ethernet PHY*

Trellis as NFV Fabric for Virtualized Cable Head-end : Networking characteristics



- *Virtual CCAP container dynamically binds to the R-PHY*
- *IPv6 Underlay DEPI / UEPI Tunnels between R-PHY to virtual CCAP*
- *Control Plane traffic (ARP, ND, RIP, DHCPv4, DHCPv6) punted to Controller and handled in ONOS Apps*
- *Successful DCHP transactions establish CM/CPE host routes in the fabric*
- *Dynamic fail-over virtual CCAP to a different container on a different host*

1. High Performance and Predictability

15k modems @ Start-up

120k Routes and 250k flows

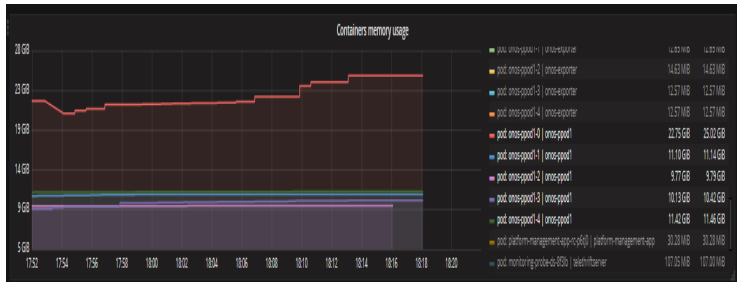
~1000 ARP / ND per second

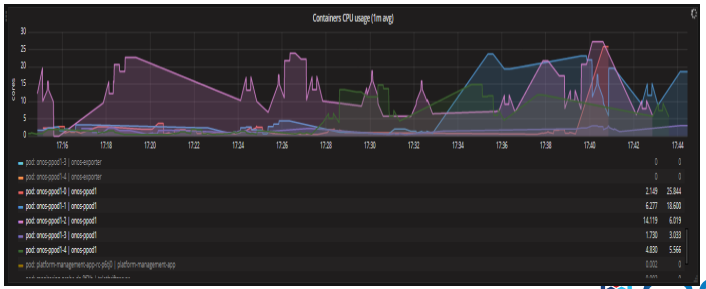
650 DHCP Transactions /sec

500 flows per second

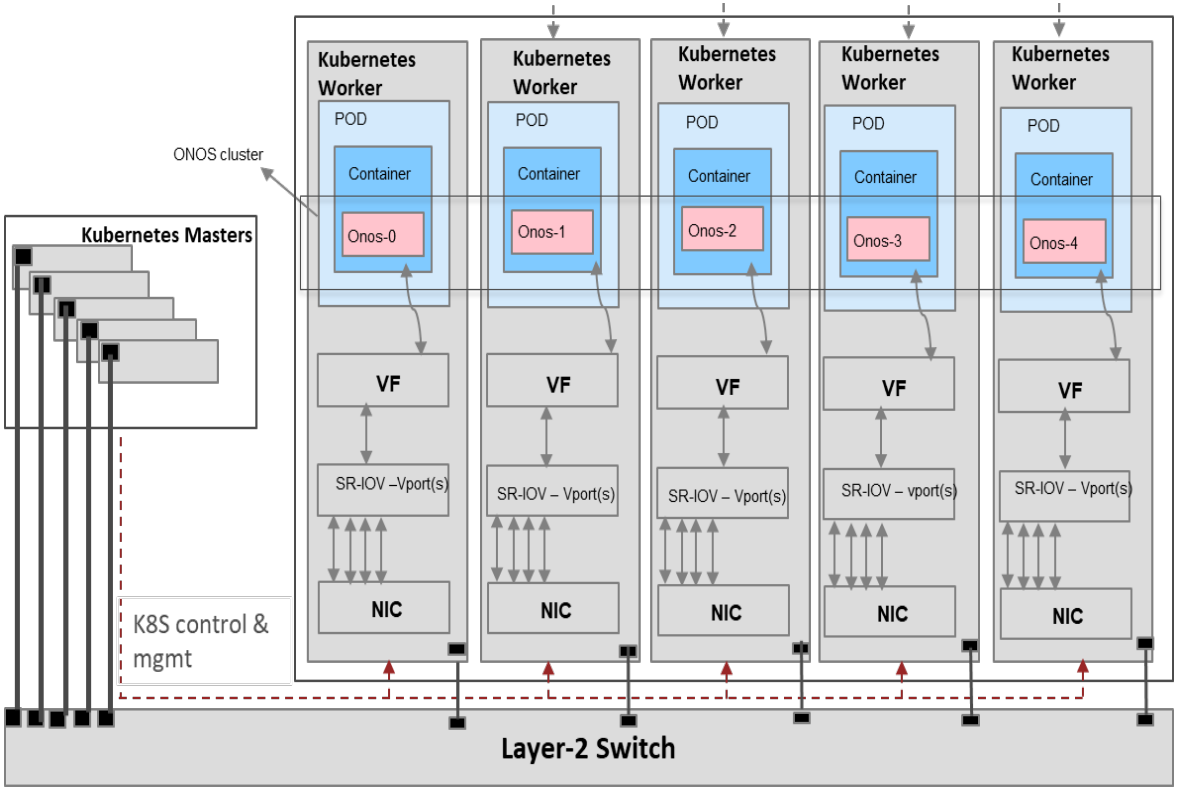
Controller synch - 120 k Routes < 2 min

- ✓ Topology specific Route optimizations
- ✓ Timers and Flows / Stats Polling Tuning
- ✓ Symmetric Probing for dual-homed hosts
- ✓ Flow-store back-up synchronization





2.Dis-aggregated and cloud-native



K8S ONOS POD Architecture

- ✓ *Kubernetes cluster design – 5 or 3 member cluster*
- ✓ *Server Redundancy and POD redundancy*
- ✓ *Servers dual-connected via Multi-chassis LAG to two switches*
- ✓ *Power it with Ansible automation for all POD deployments and installation*

2.Dis-aggregated and cloud-native

K8S POD Design Considerations:

Container Networking:

- ✓ *Multiple Network Interfaces – Flannel and SR-IOV*
- ✓ *Control Plane Interface – SR-IOV*

Efficient POD spec:

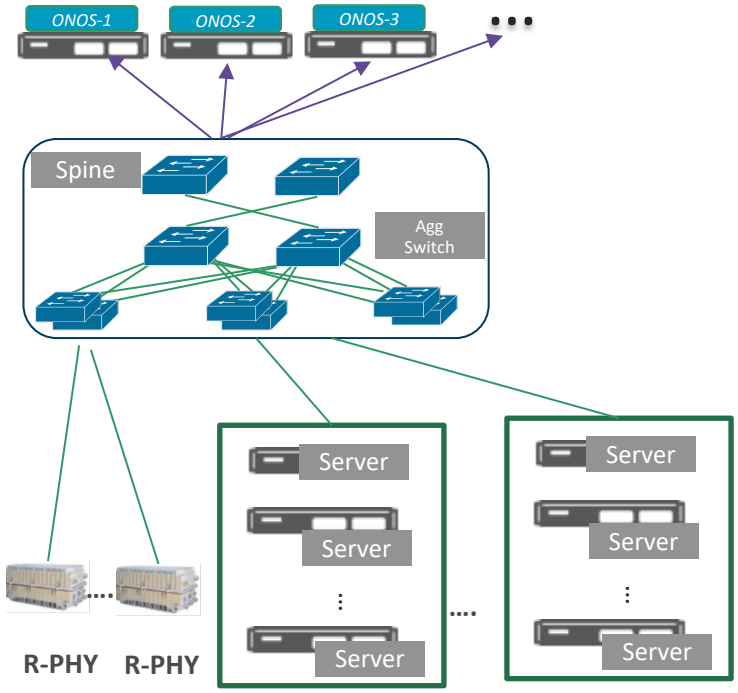
- ✓ *Statefulset POD*
- ✓ *Distributed Config store – GlusterFS*
- ✓ *Atomix File store – hostPath mode*

Config automation:

- ✓ *Maximizing Config automation through ConfigMap*
- ✓ *Deployment automation with Ansible plays*
- ✓ *POD liveliness check*
- ✓ *Pod rejoining Scenarios and Orchestration*

```
- name: "onos-{{PPOD_name}}-init"
  image: "{{docker_repo}}:{{docker_repo_port}}/{{onos_utility_version}}"
  imagePullPolicy: "IfNotPresent"
  args:
  - cp /interface-conf/* /shared-vol;cp /opt/* /shared-vol;cp /config/*.sh /shared-vol
  command:
  - "/bin/bash"
  - "-c"
  volumeMounts:
  - name: "shared-volume"
    mountPath: "/shared-vol"
  - name: "interface-conf"
    mountPath: "/interface-conf"
  - name: "pod-if-bond"
    mountPath: "/opt"
  - name: "conf"
    mountPath: "/config"
containers:
- name: onos-{{PPOD_name}}
  image: "{{docker_repo}}:{{docker_repo_port}}/{{onos_version}}"
  imagePullPolicy: IfNotPresent
  args:
  - chmod a+x /conf/*.sh;/conf/onos-{{PPOD_name}}-entrypoint.sh
  command:
  - "/bin/bash"
  - "-c"
  stdin: true
  tty: true
  securityContext:
    privileged: true
  livenessProbe:
    tcpSocket:
      port: 8181
    initialDelaySeconds: 60
    periodSeconds: 10
  volumeMounts:
  - name: karaflogs
    mountPath: {{ ONOS_HOME }}karaflogs
```

3. Scalability – Control Plane and data plane scalability



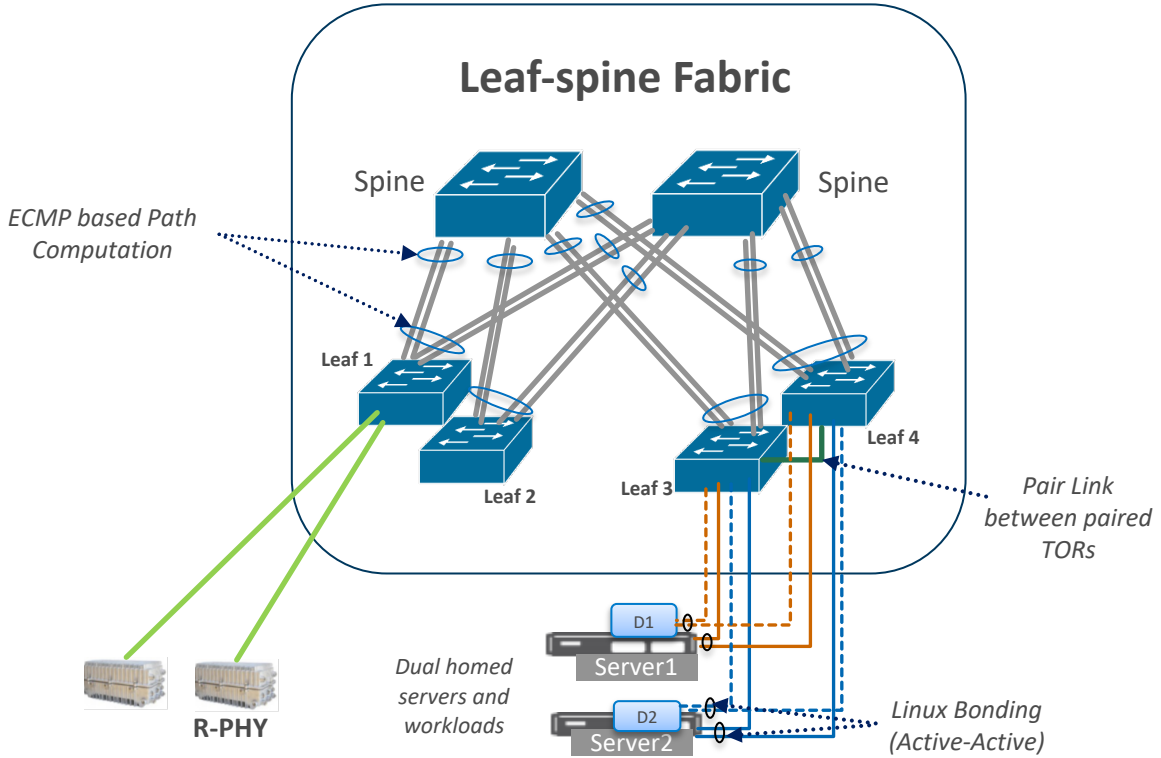
Control Plane Scaling

- *Horizontal scaling of control for increased Loads*
- *Global Network view - fully replicated across the cluster instances*
- *Master/Backup arrangement for replication of flow table entries with Partitions – advantageous for horizontal scaling*
- *Seamless Path computation on Data path scaling*
- *Fine-grain OF Packet-in Filters to handle control plane storms*

Data Plane Scaling

- *Unhindered addition of RPHY Nodes and Server on the GO*
- *Introduction of Aggregator Switch (2-level Leafs) for scalability of RPHY devices*
- *Scaling CMs behind the RPDs on the fly*
- *Bringing up more Cable Control components on the go*

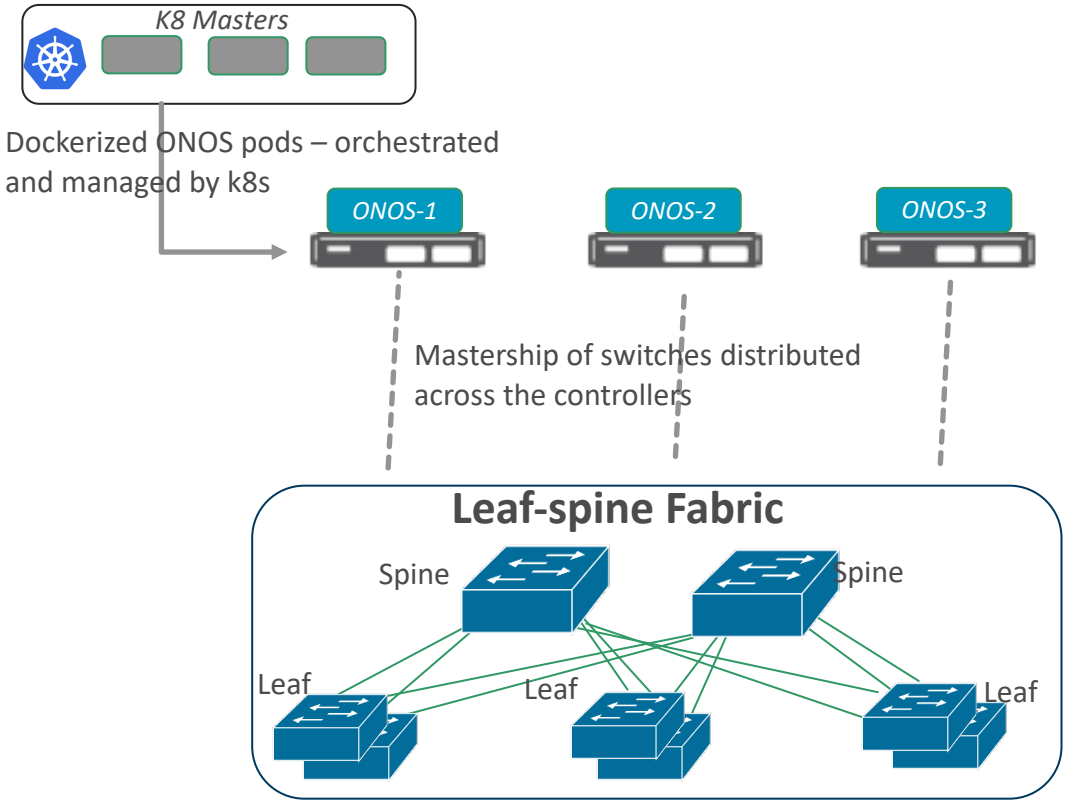
4. High Availability, Reliability and Fault-Tolerance – Data path



HA in the data-path

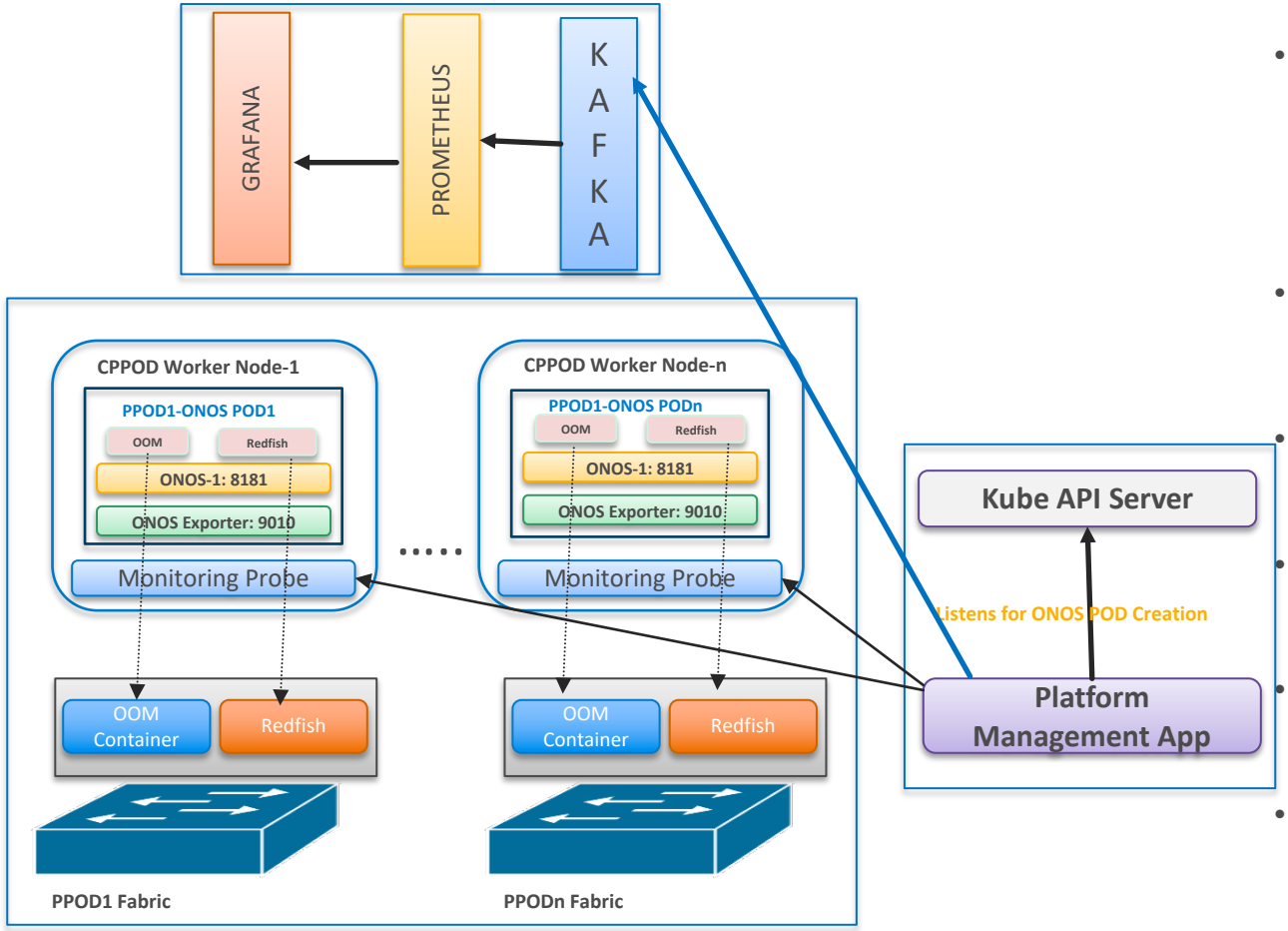
- *Servers all dual-homed*
- *Multi ECMP Links – OF-Any groups*
- *Linux bonding in active-active mode*
- *Linux bonding extended to K8S pods with VF Bonding*
- *Route reprogramming on POD HA*
- *Pair link between 2 Leaf pairs for switch local failovers*
- *Redundancy at each link level with dual links between Leaf and Spine*

4. High Availability, Reliability and Fault-Tolerance – SDN Controller



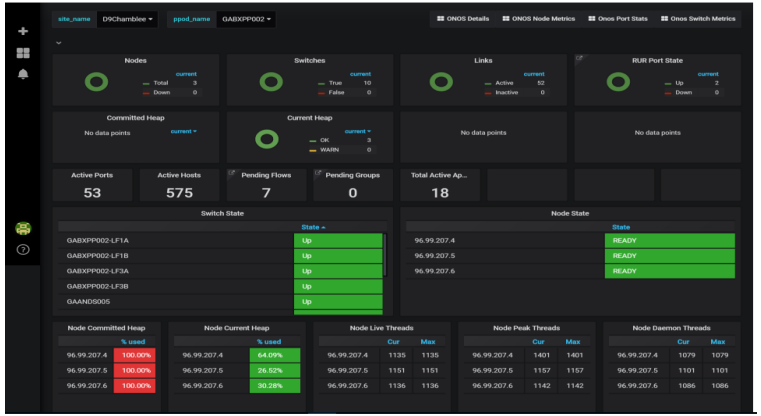
- *Switches simultaneously connects to multiple controller instances*
- *ONOS instances running as stateful set PODs with POD ordinality fixing the identity*
- *Losing ONOS instance redistributes switch mastership*
- *ONOS config state maintained in glusterfs and helps reload config on restart*
- *Redundancy of Server Hardware, Network connectivity and storage*

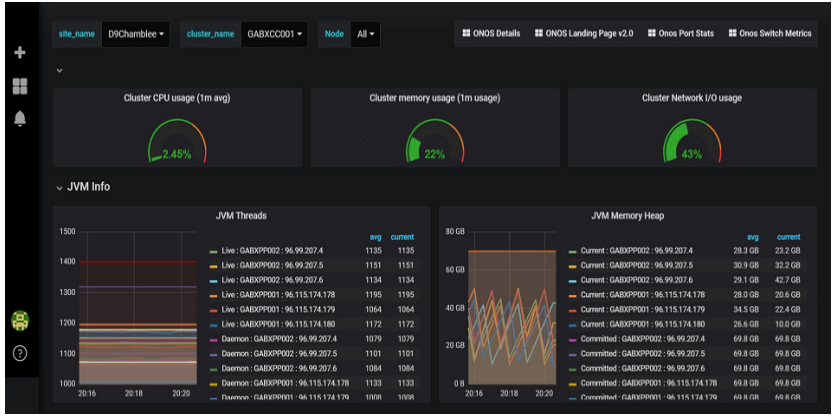
5. Real-time visibility – Telemetry based

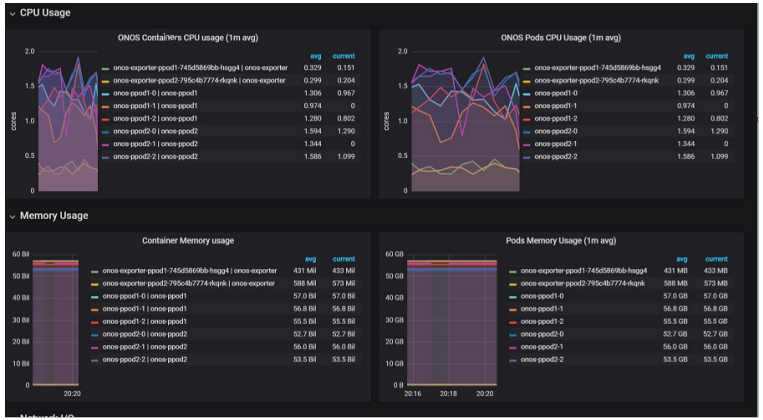


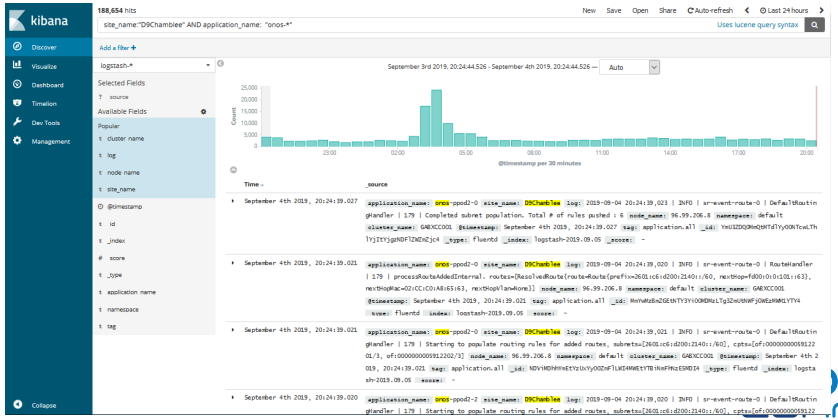
- ***ONOS Exporter**: A side car with ONOS*
  - *Collects statics using REST API*
  - *Polls switches to collect Redfish data*
- ***Platform Mgmt App**:*
  - *Detects ONOS POD creation*
  - *Sends details to Monitoring Probe*
- *OOM collects optical data and sends it out*
- ***Monitoring Probe** Sends Metrics to Kafka*
- *Consumers on cloud propagates data to Prometheus*
- *Grafana using the data on Prometheus plots various dashboards for OPs*

5. Real-time visibility – Dashboards and Logs

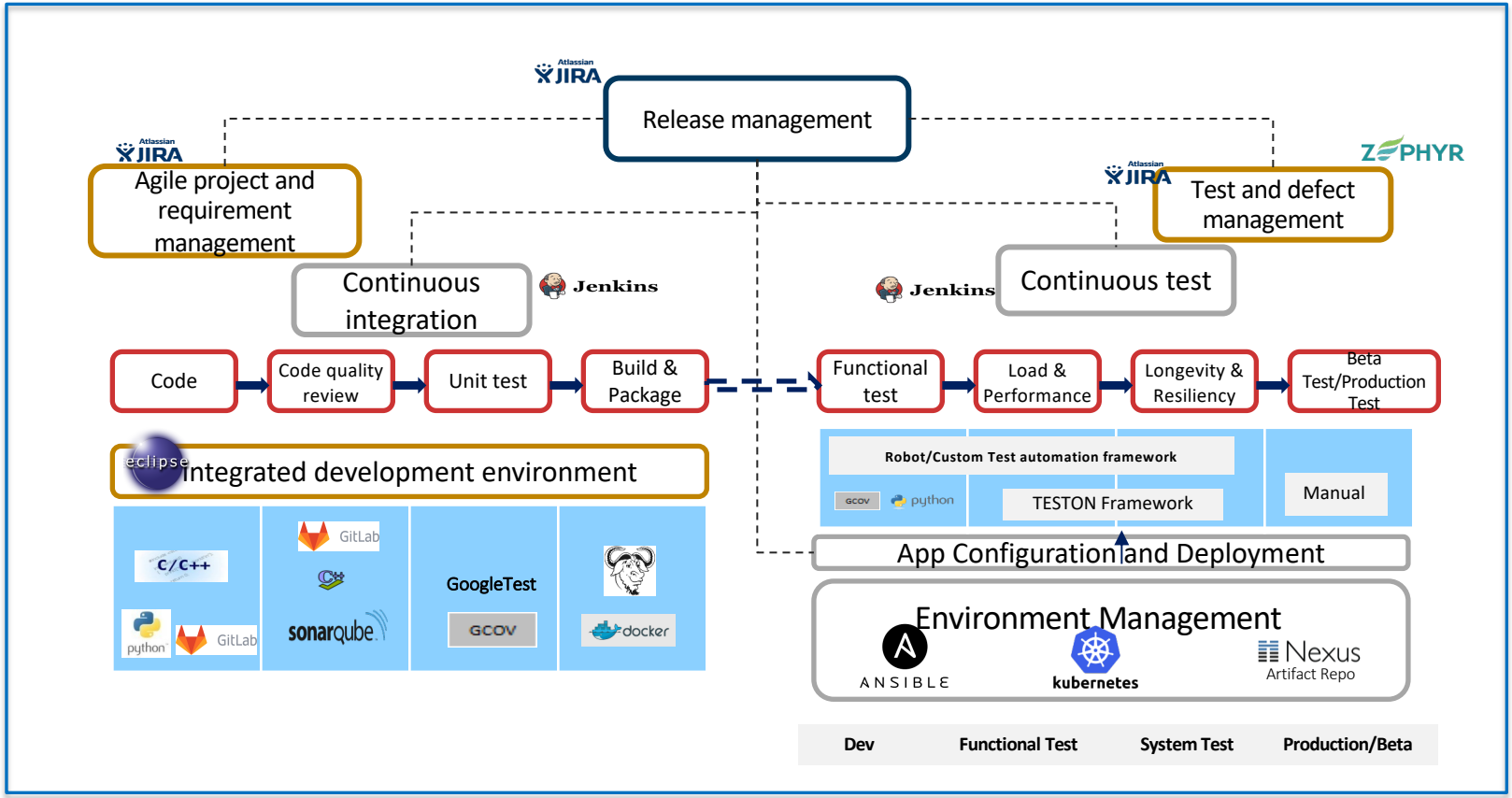






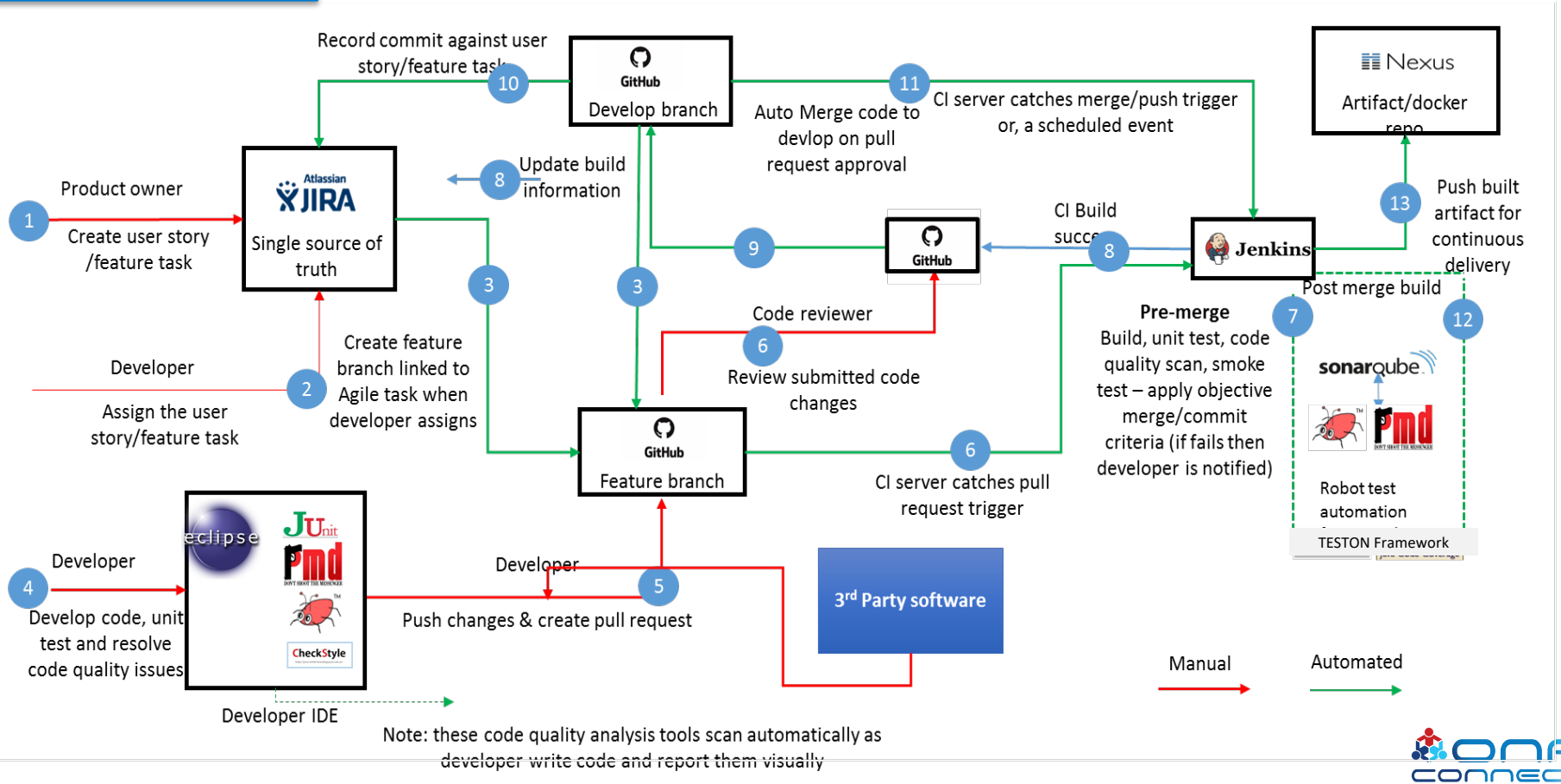


6. Agile Dev-Ops – CI/ CD Pipeline



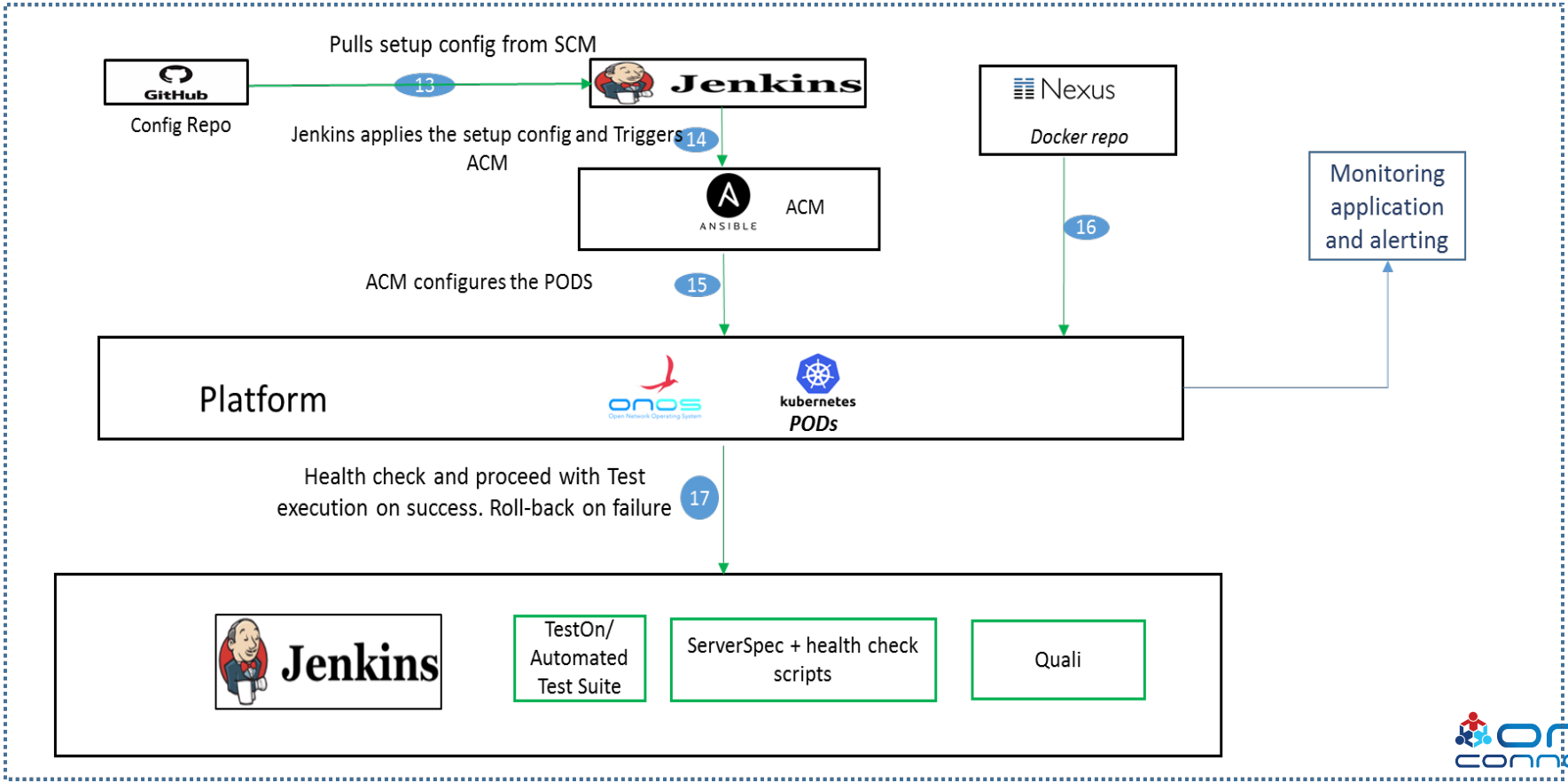
6. Agile Dev-Ops – CI/ CD Pipeline

Continuous Integration

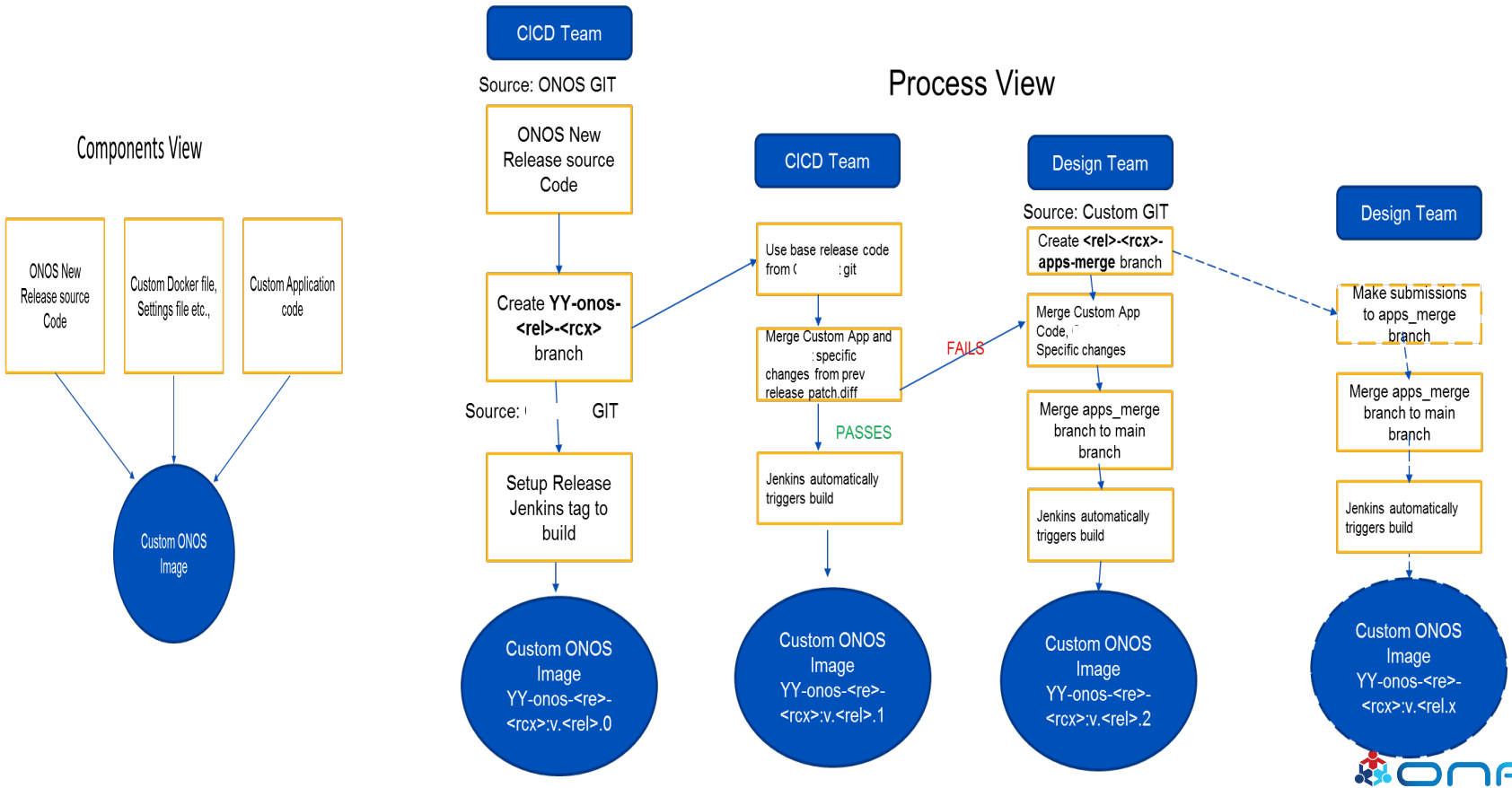


6. Agile Dev-Ops – CI/ CD Pipeline

Continuous Deployment



6. Agile Dev-Ops – CI/ CD Pipeline (Image build process)



Thank You