Multi Site Orchestration with ONAP4K8s (ONAP-for-K8s)
A foundation for Multi Edge & Cloud Orchestration
Srinivasa Addepalli
Contact: Srinivasa.r.addepalli@intel.com
ONAP Overview (till R4)

ONAP Design time

External API

ONAP Run time

Instantiate, Modify, Terminate (Network Service LCM)

Auto closed loop

Managed Environments

Openstack site

Openstack site

Openstack site

Automation platform for managing network services and VNFs across Openstack sites
<table>
<thead>
<tr>
<th>R1 – Amsterdam</th>
<th>R2 – Beijing</th>
<th>R3 – Casablanca</th>
<th>R4 – Dublin</th>
<th>R5 – El Alto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successful merge of ECOMP and Open-O in ONAP</td>
<td>Containerized</td>
<td>5G work started</td>
<td>Support for Kubernetes based sites introduced: Support for VNFs and CNFs on Kubernetes sites – vFirewall and EdgeXFoundry use cases</td>
<td>Reduce technical debt</td>
</tr>
<tr>
<td>Established Structure</td>
<td>S3P introduced</td>
<td>CCVPN use case</td>
<td>Footprint optimizations</td>
<td>Security by design</td>
</tr>
<tr>
<td>Validated with vFirewall, vDNS, vCPE, VoLTE</td>
<td>Change Management &amp; Scaling foundations</td>
<td>Increased standard alignment</td>
<td>Model driven closed loop</td>
<td>ONAP4K8S</td>
</tr>
<tr>
<td><strong>HPA introduced</strong></td>
<td><strong>HPA introduced</strong></td>
<td>Started Control loop subcommittee</td>
<td>Introduction of vIPSEC use case</td>
<td>ISTIO security for ONAP4K8S</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>HPA Matured</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ONAP – K8S Support (till R5)

1. Uniform API across cloud technologies (HEAT, K8S, Azure etc.)
2. K8S Multi-Cloud Service plugin
   - Support for deployment and services.
   - K8S yaml artifacts
   - Networking – OVN, flannel and Multus
   - Mongo DB for storing config/RBs, etcd for Day 2 configuration
3. Kubernetes Deployment (KuD)
   - Installation of software & configuration to make K8S based sites.
   - Additional of virtlet, Multus, OVN and flannel.
4. K8S-OVN4NFV (OPNFV project, visualized as part of ONAP work)
   - Support for multiple virtual networks
   - Support for dynamic creation/deletion of virtual networks
5. ONAP Integration
   - SDC for onboarding VNF/App with Helm artifacts
   - Distribution of Helm artifacts to MC.
   - SO based instantiation
   - Two modes - Self containers and with rest of ONAP
R6 and beyond: The “Edge Application” Challenge

- **Large Number of Edge Clouds**
  - Network edge clouds could be in the order of 10,000
  - On-Premise edge clouds could be in the order of 100,000

- **Large Number of App Instances**
  - CNF, IoT, Analytics, and Cloud Native apps in the order of 1,000
  - With network slicing the instances in the order of 10,000

- **Dynamic Changes**
  - Network slicing, configuration changes, cloud migration, CI/CD, etc.
  - Termination will also be critical as edge resources are precious

- **Complex Geo Distributed Apps**
  - Geo distributed apps to span multiple clouds: edge, core, public etc.
  - Complex networking & security configs for geo-distributed apps

**Operational Implications**
- Overwhelming diversity of element managers and GUI screens, as well as vendor installers
- Confounding ad-hoc automation, and impractical manual break/fix methodologies

**Operational Requirements**
- App-agnostic, full automation of orchestration and lifecycle management
- Monitoring, analytics, and real-time “policy driven” closed loop automation
Towards ONAP4K8S R6 and beyond

Application Transformation – Centralized to Geo-Distributed

**Drivers**
- Proximity
- Data Sovereignty
- Economics

**Context**

- Public/Private cloud
- Network (LAN/WAN)
- Edge Platform

- Requires >20 manual operations on each edge. Think about the effort with multiple edges!!!!

**Need for Multi Edge/Cloud Orchestrator**
Manage On-Demand & Dynamic Distributed Apps

- **Problem**: Thirty-plus configuration steps are required to program security & connectivity functions at each location.
- **Assumption**: Curated security & connectivity functions are resident and available.
- **Assumption**: Security & connectivity functions are resource constrained.

Activate the application micro-services only when users are present.
• **Problem**: Thirty-plus configuration steps are required to program security & connectivity functions at each location

• **Assumption**: Curated security & connectivity functions are resident and available

• **Assumption**: Security & connectivity functions are resource constrained

**Activate the application micro-services only when users are present**

**Automate configuration of security & connectivity with application deployment**
Edge Multi-Cluster Orchestrator (ONAP4K8s)

- **Application/Service Orchestration**
  - For applications and services delivered across multiple administrative infrastructures
- **Resource Orchestration**
  - For management of virtual compute, storage & networking resources shared by apps/services in an administrative zone
- **Infrastructure Orchestration**
  - For the building and management of Kubernetes clusters, including deployment of infrastructure services such as software updates, networking, and storage

ONAP4K8s

- **Geo-Distributed Applications**
- **Cloud Native Applications**
- **Network Functions**

ONAP4K8s is developed in LFN/ONAP repo as a standalone project.
What is ONAP4K8S?

- Is Multi Site Distributed App Orchestrator
- Independently can be deployed or deployed with rest of ONAP
- Supporting deployment of both applications and network functions
- Supporting workload types - VMs, Containers, VNFs and CNFs
- Lightweight & high performance

Micro-service based architecture
(Leverages ISTIO, AuthService & KeyCloak for IAM, Mutual TLS
  FluentD for logging
  Prometheus for metrics
  Jaeger for distributed tracing.
No centralized configuration DB, One Document DB and One KV DB
High Level ONAP4K8s Architecture

**Software Platforms**

- **Cluster Registration Controller** registers clusters by cluster owners
- **Distributed Application Scheduler** provides simplified, and extensible placement
- **Hardware Platform Aware Controller** enables scheduling with auto-discovery of platform features/capabilities
- **Distributed Cloud Manager** presents a single logical cloud from multiple edges
- **Secure Mesh Controller** auto-configures both service mesh (ISTIO) and security policy (NAT, firewall)
- **Secure WAN Controller** automates secure overlays across edge groups
- **Monitoring** covers distributed application performance, and accesses
## ONAP4K8s Roadmap

### 2019

- **Q4**
  - Centralized deployment of applications & NFs into multiple Kubernetes clusters

### 2020

- **Q2**
  - One-click deployment of “Distributed Applications & Network Functions” across multiple Kubernetes clusters
- **Q4**
  - 1) Scheduling with awareness of platform-capabilities
  - 2) Service Mesh Orchestration across Kubernetes clusters

### 2021+

- **Q2**
  - 1) Security Orchestration
  - 2) SD-EWAN Orchestration
  - 3) AI Analytics & Closed loop
R6 and R7 Plan (With rest of ONAP)

- K8S Plugin Service for Distributed applications
  - MongoDB, etcdDB

Release 6
- Cluster Management
- Onboard API
- Profile API
- Deployment Intent API
- LCM API
- Status API
- Day2 Config API

Release 7 (Still in discussion)
- K8s Plugin as VNFM
- Helm as 1st class citizen

SDC Changes (Helm as 1st class citizen)
- SO Integration
- CDS integration
- SO-VNF M integration
- A&AI Integration
Deployment Intent

An App consisting of four Micro-services

- μs1 talks to μs2, μs2 to μs3 and μs3 to μs4
- μs1" is user-facing service and need to respond within 20 Micro-seconds
- μs1", “μs2” are expected to be there together
- μs3", “μs4” don’t have any latency requirements

Why?

- Geo replication
- Geo Distribution

New Edges locations -> No manual intervention

Not only for scheduling for apps, but also VNFs/CNFs.
Selecting right edge and flavor based on Edge/Cloud capabilities and Micro-service requirements.
Why?

To enable secure communication among microservices in different locations

To enable connectivity with users

How:
- Programming ISTIO egress/ingress
- Auto NAT and FW configuration
- Programming DNS entries (e.g. Route 53)
Multi-Tenant Distributed Cloud Manager

Why?
Easy creation of tenants across multiple edges using one user operation

How:
- Creating namespaces
- Users
- Roles
- Permissions
- Quotas
across multiple sites
Why?
To secure connect edges
No static public IP address

How:
- Auto configuration of IPSEC functionality of Edge platform.
- Support for tunnel mesh and Hub-and-spoke
## Edge Platform Requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Co-existence of multiple deployment types</td>
<td>(VNFs, CNFs, VMs, Containers and functions)</td>
</tr>
<tr>
<td><strong>Advanced Networking support</strong></td>
<td>(Multiple networks, Provider networks, Dynamic Route/network creation, Service function chaining)</td>
</tr>
<tr>
<td><strong>Soft and Strict Multi-tenancy</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AI based Predictive placement</strong></td>
<td>(Collection using Prometheus, Training and inferencing framework)</td>
</tr>
<tr>
<td>Slicing in each tenant</td>
<td>(QoS On per slice basis, VLAN networks for slices, VNFs/CNFs/VMs/PODs on per slice basis or slice configuration facility on shared VNFs/CNFs)</td>
</tr>
<tr>
<td><strong>Service Mesh for Micro-services</strong></td>
<td>(Acceleration using Cilium’ Kernel bypass among service mesh side cars - e.g. Envoys; and others)</td>
</tr>
<tr>
<td><strong>Programmable CNI</strong></td>
<td>(to allow SFC and avoid multiple protocol layers)</td>
</tr>
<tr>
<td><strong>Security Orchestration</strong></td>
<td>(Key orchestration for securing private keys of CA and user certificates)</td>
</tr>
</tbody>
</table>
Managed SDWAN and Compute use case

Service Orchestrator (ONAP4K8S) – Slicing (Network etc.)

- OSS/BSS
- Operator

Subscriber
- Subscriber Edge 1
- Subscriber Edge 2 (Dynamic)

Advanced network services (e.g. DLP)
- Security VNF
- WAN Opt CNF
- SDWAN VNF

Network Edge

Cloud

ISP A
- Tunnel
- ISP B
- ISP Y
- ISP X
- CoSP / Internet / Mobile Network

MEC Services
- MEC Services
- Training

AR/VR Services
- AR/VR Services
- Media Analytics

MEC Services
- Advanced network services (e.g. DLP)
- Security VNF
- WAN Opt CNF
- SDWAN VNF

Operator slice

Advanced network services (e.g. DLP)
- Security VNF
- WAN Opt CNF
- SDWAN VNF

Operator slice

Cloud

Subscriber Edge 1

SubscriberEdge 2 (Dynamic)

Kubernetes

May be tested with dummy apps and OpenWRT, if no real apps from community

THE LINUX FOUNDATION

View in slide show mode
How does NFV based deployment with Cloud-native applications look like (Taking SDWAN with security NFs as an example)

K8S Cluster

K8S Master

resident 1 Applications (Micro-
PCD POD POD)

resident 2 Applications (Micro-
POD POD POD)

Ingress (L7 LB)

Default Virtual network (OVN)

Provider network 1 (OVN using L2 breakout, OVN LB on L2 Switch)

SLB CNF

IPS/WAF VNF

Virtual Network1 (OVN with LB)

Virtual Network2 (OVN with LB)

SDWAN CNF

Provider Network 2 (OVN)

Hardware (Multiple Nodes)

Corp
corporations

M1

M2

M3

Mx

Desktop/laptop/servers

Internet

What it proves

Advanced networking support

Soft-Multitenancy

Multi-Cluster scheduler via ONAP

Service mesh for Micro-services

Programmable CNI

Coexistence of VNFs, CNFs and Micro-services

Advanced networking support

Soft-Multitenancy

Multi-Cluster scheduler via ONAP

Service mesh for Micro-services

Programmable CNI

Coexistence of VNFs, CNFs and Micro-services

Advanced networking support

Soft-Multitenancy

Multi-Cluster scheduler via ONAP

Service mesh for Micro-services

Programmable CNI

Coexistence of VNFs, CNFs and Micro-services

Advanced networking support

Soft-Multitenancy

Multi-Cluster scheduler via ONAP

Service mesh for Micro-services

Programmable CNI

Coexistence of VNFs, CNFs and Micro-services

Advanced networking support

Soft-Multitenancy

Multi-Cluster scheduler via ONAP

Service mesh for Micro-services

Programmable CNI
Cloud Native App & NFV Stack – BICN (Potential to use CNF test bed)

Value Added Services
- Sample Apps
  - VPP, DPDK, AF_XDP based NFs/apps
- EdgeXFoundry
- Streaming

Hardware
- S1
- S2
- S3

Host Operating System
- Ubuntu (start with this)
- RH
- Clear

Kubernetes
- K8S App Components
  - ISTIO
  - gVisor
  - kubernetes
  - CollectD
  - Virtlet, Kubevirt
  - Ceph/
    - Rook
- MetalLB
- Flannel, OVN
- Prometheus
- OpenNESS
- Inferencing
  - OpenVINO

NFV Specific components
- Multus
- SROVNIC
- NFD
- QAT
- Numa Mgr
- OVS-DPDK
- OVN4K8SNFV
- SFC Mgr
- OVN4SFC
- SDWAN + Security NFs

Analytics framework
- Data Lake
- Training
- Model Repo
- Messaging

Central/Regional location

ONAP4K8S

Centralized Infra Controller
(Leverage Cluster API + Workflow manager such as Argo/Tekton)

Infrastructure Provisioning & Configuration
- KUD
- Ironic with Metal3 (or equivalent) for bare-metal provisioning

Value Added Services
- SDWAN + Security NFs

Upstream communities: ONAP, OPNFV, Many CNCF projects, EdgeXFoundry, FD.IO, DPDK, Linux, Openstack Ironic, OVS, Many ASF projects, OpenNESS, Intel Open Source

Bold & Italic – Intel led initiatives

Streaming

Central/Regional location

Edge location

Inferencing OpenVINO

Logging

Ubuntu (start with this)
CNF/VNF support via K8s in ONAP R4/R5
ONAP – Support for K8S based Sites

- Current support as in R3: Openstack based remote Clouds, Support multiple Openstack variations – Windriver Titanium, VMWare VIO, Native Newton, Ocata. Only VM based VNFs.

- Goals for R4
  - Support containerized workloads
  - Support containerized VNFs
  - Support both VMs and containers on same compute nodes. (Bare-metal deployment)
  - Support for multiple virtual networks
  - Support for dynamic creation of Virtual networks
  - Support public cloud CaaS such as AWS EKS, GCP GKE and Azure AKS (Only containers, not VMs)
ONAP – K8S Support

1. Uniform API across cloud technologies (HEAT, K8S, Azure etc.)
2. K8S Multi-Cloud Service plugin
   • Support for deployment and services.
   • K8S yaml artifacts
   • Networking – OVN, flannel and Multus
   • Mongo DB for storing config/RBs, etcd for Day 2 configuration
3. Kubernetes Deployment (KuD)
   • Installation of software & configuration to make K8S based sites.
   • Additional of virtlet, Multus, OVN and flannel.
4. K8S-OVN4NFV (OPNFV project, visualized as part of ONAP work)
   • Support for multiple virtual networks
   • Support for dynamic creation/deletion of virtual networks
5. ONAP Integration
   • SDC for onboarding VNF/App with Helm artifacts
   • Distribution of Helm artifacts to MC.
   • SO based instantiation
   • Two modes - Self containers and with rest of ONAP
K8S Plugin Service as independent manager – Modular design

- Register Edges
- Onboard resource bundle (Helm Bundle)
- Create Day0 configurations for each RB
- Instantiate RB with profile and edge site
- Config RB

- Edge registration API
- Onboard API
- Profile API
- Deployment API
- Config API

K8S Plugin Service
K8S Plugin Service with rest of ONAP

- Edge registration API
- Onboard API
- Profile API
- Deployment API
- Config API
- MC-SDC Client
- MC Broker integration

A&AI Integration
R4 Scenarios – EdgeX deployment

1. One time: Prepare K8S based site using KUD (if it does not exist)
2. One time: Register the K8S Site in ONAP by adding Kubeconfig file in ONAP (if the site is not added earlier)
3. EdgeX onboarding: EdgeX deployment and service helm charts in SDC
4. Instantiate EdgeX (by calling SO API) via postman or via VID GUI
5. Check if all EdgeX containers are successful brought up on the site (using K8S utilities on the site)
6. Basic EdgeX testing to ensure that functionality also works
   • Use consul dashboard to check the services and their status

Repeat step 4 to 6 by bringing second instance of EdgeX on a different namespace.
Also, work with Edgex team to automate deployment verification
vFirewall scenario (as VMs and containers – Hybrid)

1. One time: Prepare K8S based site using KRD (if it does not exist)
2. One time: Register the K8S Site in ONAP by adding Kubeconfig file in ONAP (if the site is not added earlier)
3. vFirewall onboarding: Create deployment and service yaml
4. Instantiate vFirewall using SO API (or VID GUI)
5. Check if firewall is successfully brought up on the site (using tools) and also ensure that three additional virtual networks are created. Also ensure that firewall belongs in all data networks. Ensure that generator and sink belong to different data networks.
6. Basic firewall testing to ensure that functionality also works
   • Check the sink dashboard to ensure that right packet streams are received by sink.
ONAP R6 and R7
R6 and R7 Plan

Release 7 (Still in discussion)
K8s Plugin as VNFM
Helm as 1st class citizen

SDC Changes (Helm as 1st class citizen)

SO Integration
CDS integration
SO-VNFM integration
A&AI Integration

Cluster Management
Onboard API
Profile API
Deployment Intent API
LCM API
Status API
Day2 Config API

K8S Plugin Service for Distributed applications
MongoDB, etcDB

Release 6