# **Tracking**

Author(s):

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Email | Version | Date |
| Deepak S | deepak.s@intel.com | 0.1 | 12/8/2020 |
| Deepak S | deepak.s@intel.com | 0.2 | 12/30/2020 |
| Deepak S | deepak.s@intel.com | 0.5 | 1/30/2021 |

Version:

0.5

**Infrastructure Networking enablement for Telco CNFs**

# **Overview**

This document describes the NF networking architecture, models and requirements for Telco’s consumption of xGVela. The architectures and requirements apply to general PaaS and xGVela components.

These requirements have been identified as MUST, SHOULD and SHALL to show mandatory, expected and optional behavior

Telcos deploy a cloud environment at the far edge, edge including regional locations and the core data centers. The networking requirements and scenarios differ at each location. While internal and external networking within a platform is a MUST HAVE requirement at the network edge or Core Telco aggregation points, it also be a SHALL HAVE requirement as you move towards the far edge site within the Telco infrastructure depending on the end user requirements. Capturing networking requirements for all the use cases and deployment models is a tall order and requires a full workstream in Telco infrastructure design.

The goal of this document is not to revisit the deployment models or define CNF networking for all the model but to highlight additional asks for Telco specific use cases and deployments

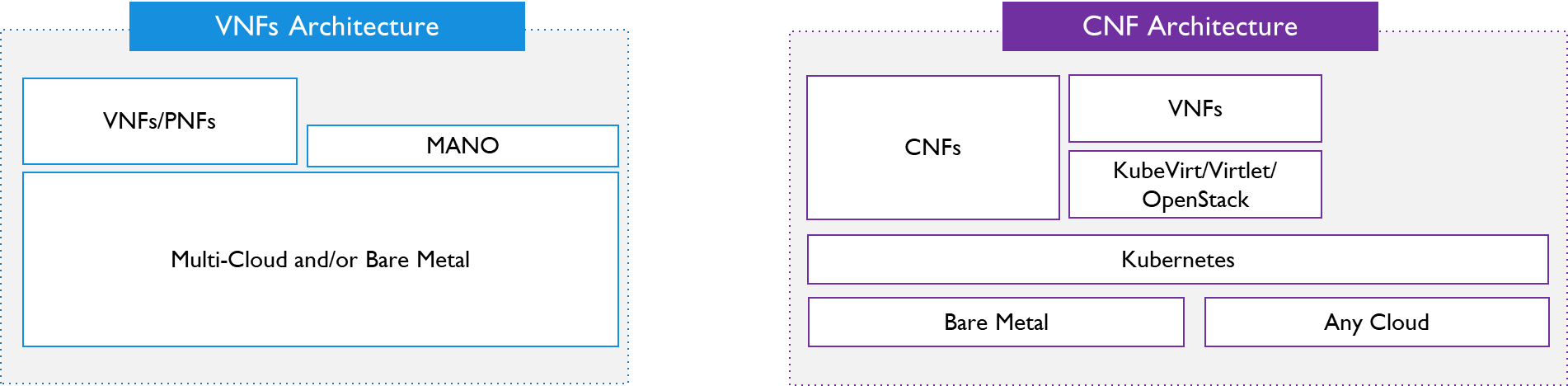
**Telco CNFs journey**

With the emergence of Edge computing and 5G technologies, telecommunication providers are looing at ways to migrate their existing monolithic services to microservices architecture and containers to address the nextgen bandwidth and throughout needs. With Network function virtualization (NFV), these providers started moving from legacy hardware towards virtualized network function (VNF) running on COTS to now moving towards containerized network function (CNF) on cloud infrastructure.

VNFs are the current defacto standard network architecture to help telco providers to move from physical dedicated hardware towards more agile services, but we still have limitation like it is hard to manage and scale without decoupling the virtual appliance from the underlaying infrastructure. Additionally, the scalability is the desired feature in the NFV being the backbone of 5G or edge computing which demands the large-scale deployment, portability, agility, scalability and lower overhead.

The cloud-native environment is built for scale, fault-tolerance, resilience and agility. The promise of CNFs is to address some of the fundamental limitation of the VNFs by moving many of these function into containers. Containerization of network architecture components makes it possible to run a variety of services on the same cluster and more easily on-board already decomposed applications, while dynamically directing network traffic to correct pods [3].

The below diagram shows typical VNFs and CNF architecture [[4].](#_References)



CNFs needs to interplay with VNFs and co-exits with legacy systems, are were much necessary from a smooth transition from traditional network architecture to cloud-native network architecture. Since network function are critical and backbone of the 5G network architecture, there is a need for CNFs to satisfy all the requirements of a network function like, networking (configuration, QoS etc), deployment, HA, scalability, telemetry, manageability etc.

This document covers the requirements of a networking for a production ready CNFs in telco infrastructure.

# **Requirements**

Key requirements of CNF networking are,

|  |  |  |
| --- | --- | --- |
| Networking requirements | | |
| Requirements | Existing Solution | Gaps w.r.t TelcoPaaS |
| It SHALL have ability to support dynamic virtual Networks & dynamic Network Creation/Termination | Multus CNI, OVN-Multi CNI (OVN4nfv), controller, OVNSFC, Kube OVN, SRIOV-CNI |  |
| It SHALL have ability for adding provider networks to requested CNF. |  |
| It MUST have ability to add multiple interfaces to a given network function. |  |
| It SHALL have ability to create a Static and dynamic Network function chaining between the CNFs, applications |  |
| It SHALL have ability to load balance across different instance of Network function |  |
| It MUST have ability to dynamically add and remove Network Policy for a given CNFs |  |
| It MUST have ability to route and manage the Co-existences of network functions and application |  |
| It MUST have ability to manage the interoperability of CNFs, VNFs and legacy device. | Virtlet, CRI Proxy, KubeVirt |  |
| It MUST have ability to dynamically configuration of Virtual switches/NICs | CNI – Kube OVN, | Network Operators? NMaaS? |
| It Shall have ability to configure the underlaying network with SDN support. | Kube OVN, OVN4NFV |  |

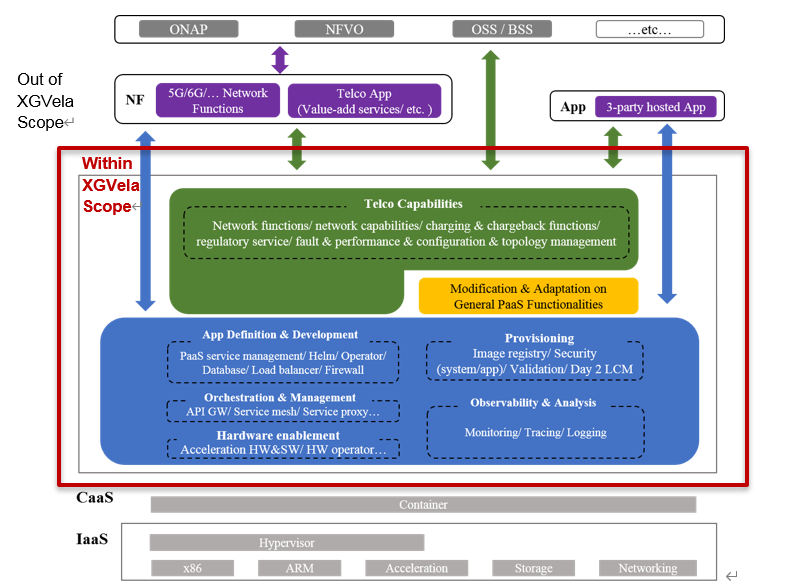
|  |  |  |
| --- | --- | --- |
| Performance requirements | | |
| Requirements | Existing Solution | Gaps w.r.t TelcoPaaS |
| It shall configure the platform parameters for Low latency and jitter | SRIOV-CNI, OVS-DPDK CNI (Virtio-User), Smart NICs CNIs, OVN-Multi-CNI | NMaaS and Network Operators |
| It shall configure the platform parameters High bandwidth and throughput |
| It shall configure the platform and software parameters for performance determinism across CNFs |

|  |  |  |
| --- | --- | --- |
| Security requirements | | |
| Requirements | Existing Solution | Gaps w.r.t TelcoPaaS |
| It Must support the Multi-tenancy across multiple CNFs | TPM/TXT? SGx, DDP | SMaaS? (CNI Security Management as a Service?) |
| It Must enable the ability to authenticate and verify of underlaying infrastructure |
| It Must enable the ability to Network traffic isolation |

|  |  |  |
| --- | --- | --- |
| General requirements | | |
| Requirements | Existing Solution | Gaps w.r.t TelcoPaaS |
| It shall support APIs for orchestrating Network Service on Telco PaaS | Ansible, NFD, Multus, istio, Telemetry (collectd, prometheus), Ceph, CMK, FPGA etc… |  |
| It Shall support Multi-residency support |
| It shall support app level and platform level network Telemetry & Monitoring |

# **High Level Architecture**

XGVela is a telecom cloud native PaaS platform designed to provide PaaS capabilities needed in the upper application software, network element development, operation and maintenance. Below is the High-level architecture diagram of XGVela.



The Green Box is the Telco PaaS layer that abstracts the telco specific capabilities for easy deployment and management of network function using underlaying General PaaS. The Networking requirements of a network function are addressed on both Telco PaaS and General PaaS.

General PaaS layer will be responsible for Provisioning the physical NIC, providing multiple interfaces, telemetry, policy apply, enhanced platform features etc. While the Telco PaaS microservice (Network-Management-as-a-Service (NMaaS)) will be responsible for abstracting the APIs that will be consumed by the NFVO/Orchestration layer.

**Proposed Architecture - high level for Networking**



Traditionally, multiple network interfaces are employed by network functions to provide separation of control, management and data/user network planes. They are also used to support different protocols or software stacks and different tuning and configuration requirements. Typically, in Kubernetes each pod only has one network interface (apart from a loopback) which is not enough for a production ready telco network function. Multus Container Network Interface (CNI) is the device plugin that can be used to create multiple network interfaces for pods in Kubernetes.

NmaaS: Network Management as a Service:

* Exposes NB APIs to Orchestration to configure the Infrastructure NIC, add/delete interface to NF at runtime, SRIOV configuration etc.
* Talks to Network Operator to setup the required underlaying driver/software etc needed based on the Network function definition

Multus: A container network interface (CNI) plugin for Kubernetes that enables attaching multiple network interfaces to pods

Userspace CNI: A Container Network Interface (CNI) plugin designed to implement userspace networking (as opposed to kernel space networking). An example is any DPDK based applications. It is designed to run with either OVS-DPDK or VPP along with the Multus CNI plugin in Kubernetes deployments

OVN-Multi-CNI: A Container Network Interface (CNI) plugin to manage network traffic flows, network policy support, including ingress and egress rule etc.

SRIOV- NIC CNI: A Container Network Interface (CNI) plugin enables the configuration and usage of SR-IOV VF networks in containers and orchestrators like Kubernetes.

**Deployment Models of PaaS Networking.**

## Make Before Break principle

# Proposed Architecture

## Existing Solution:

This section lists the open source projects which satisfy the platform networking enabling requirements for xGVela.

**OpenNESS**[7] (Open Network Edge Services Software) is an edge computing software toolkit that enables highly optimized and performant edge platforms to onboard and manage applications and network functions with cloud-like agility across any type of network. OpenNESS Enhanced K8s extension makes it suitable to deploy Telco workload (CNFS) including management, control and data plane along with traditional cloud applications. This Features can be base line for the Telco PaaS. Some of the network funcation specific features like CNFs, multi interface, acceleration support, SFC, telemetry etc that OpenNESS enables will be discussed in this section

Below is the list of the Opensource projects that satisfy the requirements of networking to run the CNFs.

**Networking requirements**

**Requirements**: <add explanation for every requirements>

**Current Opensource components address the requirements**: e.g, ovn-multi CNI, Multus, etc.

<I will expand this further>

**Gaps:**

**Performance Requirements:**

**Requirements**: <add explanation for every requirement>

**Current Opensource components address the requirements**: e.g, ovn-multi SRIOV CNI etc, ovs-dpdk CNI, smart NICs etc

<I will expand this further>

**Gaps:**

**Generic requirements**

**Requirements**: <add explanation for every requirement>

**Current Opensource components address the requirements**: e.g, deployment, ONAP, EMCO, kubespray, telemetry etc.

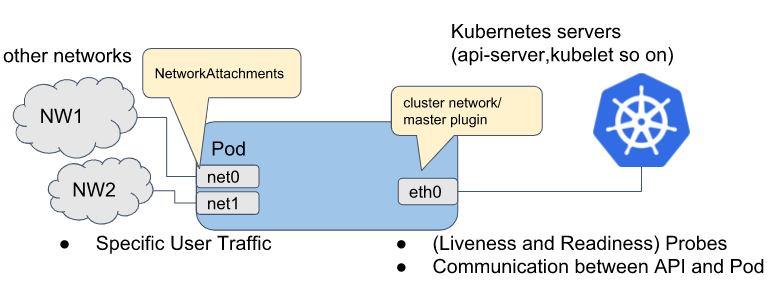
<I will expand this further>

**Gaps:**

List of CNIs that will enhance the dynamic configuration and multi network performant network for CNFs

**Multus-cni:**

Multus CNI is a container network interface (CNI) plugin for Kubernetes that enables attaching multiple network interfaces to pods. Typically, in Kubernetes each pod only has one network interface (apart from a loopback) -- with Multus you can create a multi-homed pod that has multiple interfaces. This is accomplished by Multus acting as a "meta-plugin", a CNI plugin that can call multiple other CNI plugins.



<https://github.com/intel/multus-cni>

**Ovn4nfv-k8s**

OVN Multi CNI and associated network watcher (for K8S) Enables multiple OVN based networks and enables PODs/VNFs to sit on multiple virtual networks. Founded by Intel in OPNFV: <https://gerrit.opnfv.org/gerrit/gitweb?p=ovn4nfv-k8s-plugin.git;a=summary>

**Kube-OVN**

Kube-OVN integrates the OVN-based Network Virtualization with Kubernetes. It offers an advanced Container Network Fabric for Enterprises with the most functions and the easiest operation.

**ligato.io**

An Open Source Go Framework for Building Applications to Control and Manage Cloud Native Network Functions (CNF)

**SRIOV-NIC CNI**

The SR-IOV CNI plug-in plumbs VF interfaces allocated from the SR-IOV device plug-in directly into a Pod

<https://github.com/intel/sriov-cni>

**OVS-DPDK CNI**

The Userspace CNI is a Container Network Interface (CNI) plugin designed to implement userspace networking (as opposed to kernel space networking). An example is any DPDK based applications. It is designed to run with either OVS-DPDK or VPP along with the Multus CNI plugin in Kubernetes deployments. It enhances high performance container Networking solution and Data Plane Acceleration for containers. <https://github.com/intel/userspace-cni-network-plugin>

## Telco PaaS Specific Components:

* **NmaaS: Network Management as a Service:**

<Will add low level details of these new components>

* **Network Operator:**

<Will add low level details of these new components>

## Analysis - Gaps, Pros and Cons

# Gaps

List the gaps in the current architecture

Analysis of where these gaps can be addressed

# Summary

# References

[1]<https://www.openshift.com/blog/telco-revolution-or-evolution-depends-on-your-perspective-but-your-network-is-changing>

[2]<https://www.intel.in/content/www/in/en/communications/why-containers-and-cloud-native-functions-paper.html>

[3 <https://www.redhat.com/en/blog/cnf-and-vnf-certification-red-hat-and-intel>

[4]<https://wiki.onap.org/download/attachments/79203136/VNF-CNF%20OVP%20-%202%20key%20slides_V3.pptx?version=2&modificationDate=1582822482000&api=v2>

[5] <https://github.com/XGVela/XGVela/wiki/Architecture-Document>

[6]<https://events19.linuxfoundation.org/wp-content/uploads/2018/07/ONS2019_Cloud_Native_NFV.pdf>

[7] <https://www.openvswitch.org/support/ovscon2019/day1/1133-OVNForK8sNetworkFunctions.pdf>

[8]<https://events19.linuxfoundation.org/wp-content/uploads/2017/11/2018-OSSNA-CNF-Journey-in-Telecom-Seminar.pdf>

[9] <https://events19.linuxfoundation.org/wp-content/uploads/2018/07/ONS2019_Cloud_Native_NFV.pdf>