Cloud Native

OPNFV Cloud Native Working Group Proposal

- **Scope** of the new "*OPNFV Cloud Native Working Group*"
  - Serve as a forum to coordinate among all projects working on cloud native initiatives and drive for common goals and strategies
  - Serve as a common bridge with CNCF, ONAP/LFN, edge and other relevant communities. Represent OPNFV to work with these communities as a liaison and/or on joint initiatives.
  - Provide periodic updates to TSC

- **Work items and objectives** of the new "*OPNFV Cloud Native Working Group*"
  - Initial work item / starting point:
    - Create a "world map of cloud native in OPNFV" (short presentation) - document cloud-native related OPNFV projects and efforts, their current status as well as plans.
      Outline how these projects fit into the larger cloud-native ecosystem / articulate relationships to other projects (e.g. CNCF) / identify opportunities for future work.
  - Establish a cloud native CD toolchain and process
  - Drive commonalities among OPNFV k8s scenarios. Produce common Kubernetes scenarios (e.g. two) with long-live community environment
  - Work with projects (initially select a few) to help them
    - produce cloud native artifacts
    - adopt cloud native tools and practices
    - adapt/expand to support cloud native environments
  - Ensure new projects include cloud native where appropriate
  - Establish robust connection and engagement with upstream CNCF, ONAP/LFN, edge and other communities
  - Support multi-clouds based on k8s: e.g. ONAP, edge, public clouds
  - Drive longer term efforts: e.g. common API, service mesh, 'middle-box' data path, acceleration, edge support, etc.
  - Produce user oriented and developer oriented training material

---

We propose to identify some low hanging fruits for initial goals, and at the same time start to formulate and drive some of the longer term objectives. Meeting time can be shared among the main projects involved to reduce overlap and may choose two alternating time zones to accommodate more participants. May look into existing WGs (Infra, Testing WG) for more best practices.
3) Ask from TSC

- Creation of a WG with the defined scope and proposed work items
- Current task force members to kick start the WG, WG is open to everyone interested in participating.

PS: Recommendation summary deck previously presented to TSC: Cloud native WG recommendations.pdf
==============================

Objectives:

OPNFV started the open source NFV journey largely based on a VM approach, specifically Openstack. This approach means more than just VM as basis of virtual compute unit, but also the implies a "virtual appliance" model. This model influences the overall architecture, software and service design philosophy in many aspects beyond the choice of hypervisors. Supporting cloud native for NFV, therefore, means supporting ALL of the following three aspects (as defined by CNCF and others):

1) Supporting containers,

2) Supporting management and automation of containerized software

3) Supporting micro-service model of software

More background information can be found in this ONS Technical Forum Presentation. (March 2018, Los Angeles, CA)

These three aspects are inter-linked closely in order to fully realize the benefits of cloud native paradigm for NFV. Therefore, our overarching objective is to integrate best of breed cloud native software solutions and best practices for supporting cloud native deployment of NFV solutions for all end users and high priority use cases. Specifically,

1) Container and container management

- Various types of container implementations
- Other light weight compute choices e.g. kata, virtlet, unikernels
- Performance, security, other aspects of container improvement
- Storage support in the cloud native environment
- Orchestration : Kubernetes (auto deployment, scaling, management etc.), Kubernetes + Openstack (including Kubernetes on Openstack, and other variations)
- Integration of these upstream components
- Goal is to move the integration process to XCI style continuous integration with both Openstack and Kubernetes

Currently the integration and testing work happens in several tracks within OPNFV

- container4nfv, working with Compass and Joid, produces a number of artifacts in Fraser.

- other artifacts, as listed in Fraser status page, by installers (Apex, Compass, Joid). Most are baremetal (ie starting with k8s-*), a few are on top of openstack (os-odl-k8s-*). See also this analysis.
- XCI sandbox supports a Kubernetes based deployment (see xci sandbox doc).
- most of current configurations are experimental to add special support/features to kubernetes and are for developers in nature - these are all very important work. However, we also need to support many (probably majority) users who only want a convenient Kubernetes configuration that they can use and has been tested in the LFN community for networking related use cases.

Recommendation #1:

Establish a coordinated effort to produce a common configuration that can be jointly tested and promoted into a high quality system that is consumable to (a) users who seek convenience for NFV use cases (b) developers/users in other projects who can leverage OPNFV's work (e.g. ONAP) (c) other developers who want to build other higher layer software on the top.

Focus on two main configurations: Kubernetes baremetal and Kubernetes on top of Openstack.

Sync with the ongoing efforts of consolidation through XCI and converge on a common set.

2) Container networking

- There are many aspects of container networking, some involve VNF data path, but many don't. (Note: the phrase "data path" is ambiguous).
- For the normal case, we should stay fully faithful to Kubernetes and CNCF, and participate in the ongoing efforts there (upstream first). This should be our baseline in producing a consumable Kubernetes configuration (see Recommendation #1). container4nfv, e.g., supports CNI and plug-ins, multus, SRIOV etc.
- For the "bump-in-the-wire" case, a.k.a "middle box", various acceleration designs have been proposed/developed, OPNFV should work with related projects to test/validate their applicability with respect to solving user's demonstrated requirements. Some of these are based on ongoing data path work within OPNFV and projects within LFN (e.g. ligato/FD.IO in user space), and others in open source community (e.g. Linux BPF in kernel space). See a separate bullet below to distinguish this case. These acceleration methods are critical to many NFV use cases, but we should not mix importance with maturity, the promotion of these features should be gated with a neutral set of test cases. Doing so is not to demote its importance, but to give space and freedom in its development so those projects can move faster.
- We are not here to "choose", but provide a consistent validation process to test features, gauge maturity, and help down-stream to consume these features in an easier way. All solutions demonstrated benefits and maturity should be promoted to the same level.

Recommendation #2:

Encourage testing projects to develop consistent test methodologies for container networking. Build these test methods into CI/CD/XCI test and promotion pipeline (see Recommendation #1).

Encourage OPNFV integration projects and the upstream projects to integrate and test data path features/acceleration in OPNFV to enhance networking stack for all of the LFN community.

Encourage benchmarking of performances of various networking solutions for OPNFV use cases, e.g. Clover is trying to use tools like vsPerf and others like jMeter to conduct explainable benchmarking.

3) Micro-services support

- Micro-service service mesh
- Micro-service visibility/data collection and analysis, and related tools and automation methods
- Tools for best practice of micro-services
- Software (micro-services) components commonly needed for NFV use cases
- Test methodologies and tools in the micro-service/cloud native approach
- The micro-service framework is applicable to many aspects of NFV software ecosystems, including containerized sample VNFs, test tools, measurement tools, other management/control or applications, e.g. ONAP.

Recommendation #3:

The work here has been started in the Clover project. We encourage more community participation in advancing these goals.

Encourage containerized software, in OPNFV and across LFN, to adopt micro-service architecture supported by a feature rich high performance service mesh.

4) Cloud native continuous deployment (CD)

To disambiguete Continuous Deployment from Contiguous Delivery and Contiguous Integration, please use this Atlassin definition.

- One of the integral part of being cloud native software is its continuous deployment (Note: Continuous Deployment is a distinct process from the general CI/CD discussion in OPNFV.)
- We should extend current CI (strictly speaking XCI is still CI, just for clarification of terminology) to new CD for cloud native software. See this proposal from the ONS Technical Forum: Extending CI/CD to support Cloud Native VNFs and Operations.
- Note that this toolchain works for both containers and VMs - one can adopt this style of CD for VM as well - however, the initial objective should be focused on container first.

- Will require "long lasting computing resources" - this can be current UNH pods being allocated for this purpose, Pharos-pods dedicated for this purpose, virtual public cloud (GCE, AWS, ...) paid for for this purpose, or new resources.
- The Clover project is currently building the software tools and automated tests that implement this CD pipeline. The first step of long running testbed is being prepared for Gambia.

Recommendation #4:

Continuous deployment is an integral part of being cloud native. Encourage participation to Clover project to establish a CD tool and process during Gambia and general use thereafter.

Link XCI and CD together for a complete cloud native CI/CD.

Consider to dedicate a POD in UNH LaaS to be OPNFV's long running testbed.

5) Support for "MiddleBox" Network Functions

While the adoption of technologies like the Micro-Services (number 3 above) are very well suited to services that operate at layer 7 (such as management and control functions), the interface to these service meshes does not provide access to the entire raw packet which is necessary for many data plane functions.

There needs to be support for a different framework to support L3/4 functions, such that packet headers can be inspected and appropriate action taken. Examples would be vCPE or vEPC.

In addition these network functions often need to be able to be connected in a logical sequence. In an OpenStack / VM based environment this would be called a Service Function Chain, but we won't use this term as it may implpy a certain implementation. ETSI uses the term VNF Forwarding Graph, and this kind of logical sequence of functions is required, regardless of how it is implemented.

Recommendation #5:

Review existing open source projects to achieve the needs of a middlebox network function. An example, implemented in user space, would be Ligato, which makes use of fd.io (see https://github.com/ligato). Another example, implemented in kernel space, is eBPF and related project ioVisor or solution like Cilium.

6) Concrete use cases and sample applications

- Use case driven (user/customer driven) is an important way to make OPNFV more relevant to customer needs and be more "consumable".
- Work cross project to identify common priority use cases. Future deployment should be a focus for us as today's commercial deployment has yet to adopt cloud native in large scale.
- One of the outcome of these sample VNFs should be a best practice document helping the community/vendors/customers understand the challenges involved and best methodology of moving to cloud native.
- It will be greatly beneficial is projects within LFN, and other related upstream projects, can share work on use cases and examples. Specifically, we should join forces in creating cloud native use cases.

Recommendation #6:

Work across projects to create a set of common use cases that reflect shared market requirements.

7) Cross project collaborations

- Eat your own dog food. Encourage OPNFV projects consider containerizing their artifacts, supporting testing containerized solutions, adopting CI /CD, and service mesh.
- ONAP also aims to move to cloud native. We need to position OPNFV in helping ONAP achieve their goals without unnecessary replication.
- Auto project is working on running ONAP on OPNFV VIM. It will be beneficial to run ONAP as a cloud native application on OPNFV's cloud native infrastructure.
- FD.IO has goals in high performance data path for cloud native VNFs. We should investigate best way to integrate that as one of the acceleration methods.
- Akraino, a new LF project, is starting to work on edge software stack to support use cases in enterprise, IoT and operators that will have similar needs.
- The CD (continuous deployment) initiative is applicable for all projects within LFN, and related to upstream (CNCF, Openstack), and compliments to the XCI initiative, Lab-as-a-service/Pharos etc.
- Cloud native is a general theme that we should push together in LFN.

Recommendation #7:

Encourage OPNFV projects consider containerizing their artifacts, supporting testing containerized solutions, adopting CI/CD, and service mesh.

OPNFV should take a lead to drive a LFN cloud native initiative.

Related projects in OPNFV:

Projects in OPNFV are currently working towards at least part of the above objectives: (please suggest addition of anything missed out)

- Container4nfv (formerly OpenRetriever): Container4NFV
  - Focuses on containers, kubernetes
- Clover: https://wiki.opnfv.org/display/CLOV
  - Focuses on micro-services management, service mesh, CI/CD, sample VNF micro-services
- XCI: http://docs.opnfv.org/en/latest/infrastructure/xci.html
  - Focuses on cross-project CI
- Auto: ONAP-Automated OPNFV (Auto)
  - Focuses on ONAP on top of OPNFV's NFVI
- Edge cloud: Edge cloud
  - Focuses on carrier scale edge cloud.
- FDS(FD.IO): https://wiki.opnfv.org/display/fds
  - Several fronts in networking data path
- Installer projects that produce Kubernets scenarios: Apex, Compass, Joid
- Test projects/ Tesing Working Group - how to reuse test tools for cloud native
- Infra Working Group: How to allocate/use infra to support long running micro-services
- Doc: in OPNFV and cross LFN projects

==

Raw discussions during zoom meetings are captured in this etherpad: https://etherpad.opnfv.org/p/cnwg

new "OPNFV Cloud Native Working Group"