OPNFV

XCI Solution Brief

Cross Community Continuous Integration (XCI) Empowers Innovation by Increasing Collaboration Between OPNFV and Upstream Communities

With XCI, OPNFV regularly integrates the latest from each supported branch of select upstream projects, slashing the time to implement new features and address bugs from months to days.

A Linux Foundation Collaborative Project

The OPNFV project has embraced the DevOps development model by building a sophisticated continuous integration (CI) pipeline. Until now, OPNFV has integrated the latest major release of upstream projects such as OpenStack, OpenDaylight, FD.io and others. Major releases of these projects can be on a multi-month cadence, for instance six months. Waiting this long for integration is sub-optimal since it delays how quickly OPNFV community members can access the latest upstream innovations; and it also delays how quickly upstream communities can get valuable feedback from OPNFV testing.
The XCI initiative solves this problem by regular integration and testing of the latest software versions from several upstream projects.

“XCI is needed to address the complex testing challenges in OPNFV. From stress, to robustness, resiliency, VNF on-boarding and end-to-end testing, XCI is now a part of OPNFV's DNA and allows us to test as closely as possible from upstream to provide realistic ‘Telco grade’ feedback, including orchestration."

- MORGAN RICHOMME, NFV ARCHITECT, ORANGE

THE CHALLENGE

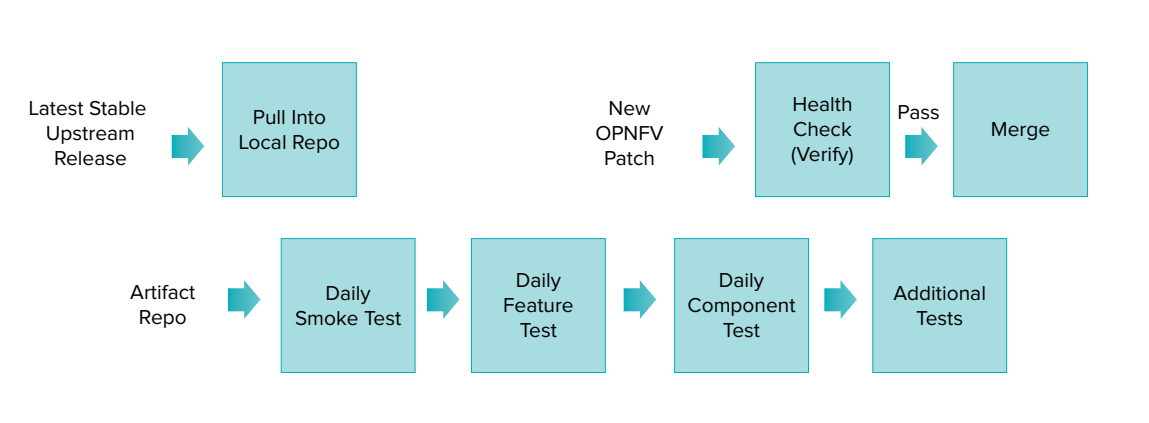
A recent survey of Communications Service Providers (CSPs) indicates that 80% of those surveyed feel that the DevOps software development model (DevOps) is essential or important to NFV success. The top DevOps engagement activities were evaluating DevOps toolchains, and automating and testing infrastructure.

DevOps from the OPNFV perspective means a cultural and mindset change where development, release, and operations teams work together in concert by applying CI/CD principles, automation, and practices to develop, test, deploy and monitor software systems. The process part of DevOps, also known as continuous integration/continuous deployment, or CI/CD, is something OPNFV has been working towards since the inception of the project. OPNFV's CI pipeline allows the community to benefit from a DevOps approach where users get quick access to new features, developers get rapid feedback, and the overall software stack undergoes a large number of incremental changes as opposed to few dramatic changes every few months.

At a high level, the current OPNFV CI pipeline can be summarized as follows:

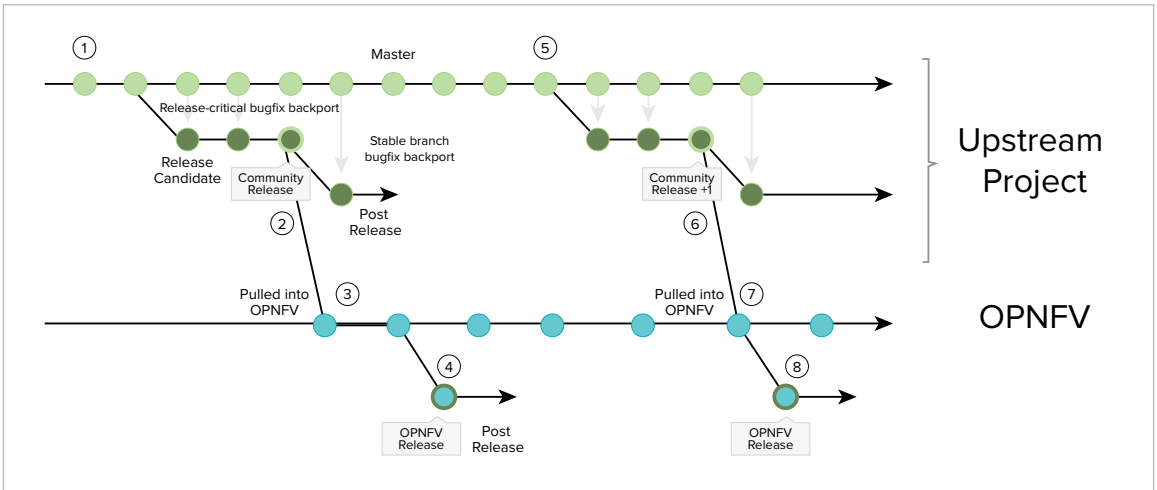
1. The OPNFV project pulls the most recent major releases of upstream projects. For example, in the case of OpenStack, this would be the most recent six-monthly release; e.g., OPNFV Danube integrates the OpenStack Newton release.
2. For patches sent to OPNFV Gerrit for review, there are various verification jobs that run against the patch to provide feedback to submitters and reviewers.
3. Automated deployment occurs in a variety of environments across multiple hardware platforms via the OPNFV Pharos project. The CI pipeline currently integrates and installs (by invoking different installers) different combinations of stack components, projects and configurations, called OPNFV scenarios, on a daily basis and executes a smoke test on each scenario. Next, additional automated tests of increasing complexity are executed against the scenario.

The below diagram summarizes these three flows:



Simplified View of the OPNFV CI Pipeline

While the pipeline is quite advanced, it can always be improved. One major challenge with the existing approach is that the upstream project code is not current. The below diagram shows how upstream projects are integrated into OPNFV today and how it impedes the community's progress.



1. New feature committed upstream	**5.** If the feature needs a bug fix, new patch committed upstream +0 months
2. Feature appears in upstream release +1-6 months	**6.** Fix appears in next upstream release +1-6 months
3. Feature appears in OPNFV repo +2 months	**7.** Fix pulled into OPNFV repo +2 months
4. Feature appears in OPNFV release +4 months	**8.** Fix appears in next OPNFV release +4 months

OPNFV Upstream Project Integration and Impact (Durations as estimates)

Let us walk through a few situations of how the current integration approach is limiting:

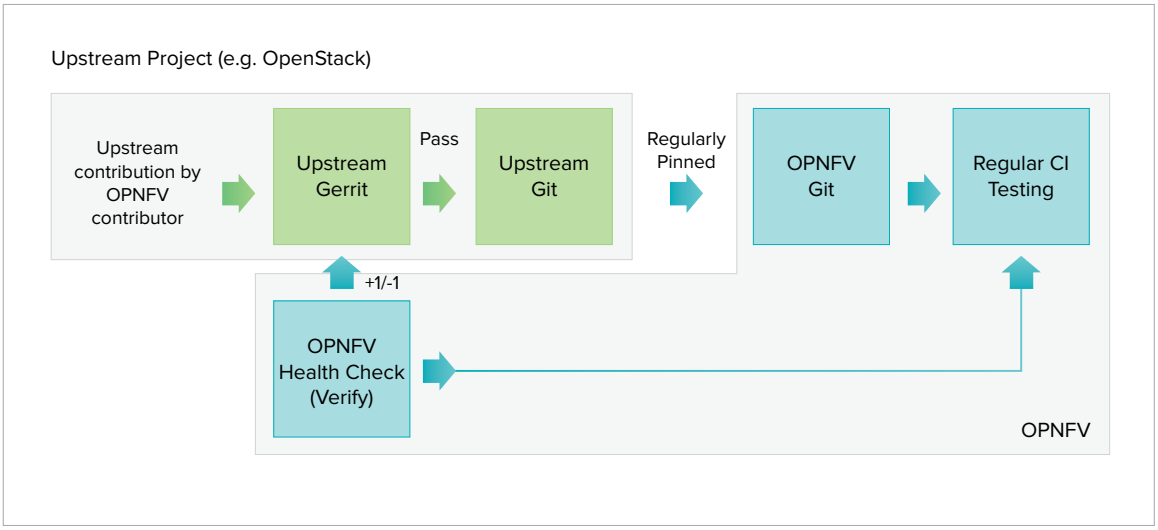
A. **A Proof of Concept (PoC) lab wants to try out a new upstream feature:** Per the above diagram, once a developer commits the code, it could take 3-8 months before the feature appears in the OPNFV repository. It is roughly an additional 4 months for the feature to appear in an OPNFV major release; the total time elapsed being 7-12 months. In the case of OpenStack, hypothetically, a patch merged for the Newton release in late March or early April of 2016 would have shown up in OPNFV Danube on April 4, 2017.

B. **A new upstream feature has a bug, and an OPNFV user wants to submit a fix, and then try out a fixed version:** As you see, this will take 3-8 months for the OPNFV community to try out the new feature and find the bug. Then it might take an additional 5-10 months for the fix to appear in the next upstream release which then propagates to the next OPNFV release over six months. So for a new feature to be developed, debugged, and to finally appear in an OPNFV release could take 14-24 months from initial development.

Clearly, these delays are detrimental to the community's goal of rapid integration, validation, and verification. XCI aims to re-engineer the status quo.

THE SOLUTION

The XCI initiative integrates the latest from all supported branches of select upstream projects on a periodic basis instead of waiting for a major release. The initiative will start with regular integration of OpenStack, OpenDaylight[1] (ODL) SDN controller and the FD.io virtual switch. The below diagram shows how this works:



XCI Integration Tasks

[1] In reality, an XCI approach with ODL has been in the works for some time, but it is now being improved and formalized.

XCI involves two primary integration and testing tasks:

1. For patches submitted upstream by community members, OPNFV XCI will run verification job(s) depending on the project and the patch, providing feedback to the upstream community Gerrit in the form of +1/-1.
2. OPNFV XCI will use the latest from all supported branches on a regular basis (e.g. daily), integrate and install them into scenarios, and perform regular CI testing against those scenarios.

This approach solves key problems identified above:

1. Upstream changes can now be utilized by OPNFV very quickly; e.g. daily.
2. Feedback can now be provided rapidly, again say daily. A feature development or bug fix cycle can now be compressed from months to just days.

XCI UNDER THE HOOD

XCI utilizes most of the current tools from the RelEng and Pharos projects. Additionally, XCI uses two OpenStack infrastructure tools: Bifrost and OpenStack-Ansible (OSA) for the purposes of provisioning nodes and installing different scenarios built from the latest supported branches of OpenStack, ODL and FD.io:

Bifrost: Bifrost is an OpenStack project for bare-metal provisioning of server nodes. It builds on the OpenStack Ironic project, except that it is used independently of other OpenStack services. The project provides a set of Ansible playbooks that deploy a base image onto bare metal hardware in an automated fashion.

OpenStack-Ansible: OSA, also an OpenStack project, uses Ansible to deploy an OpenStack environment on a provisioned node. The project creates Ansible playbooks to deploy core and optional OpenStack services. OSA is used to install different scenarios with ODL and FD.io.

With the above tools, XCI will support three base operating systems: Ubuntu 16.04, CentOS 7 and OpenSUSE 42.2. As a side technical note, the XCI initiative has additional complexity: since Bifrost and OSA are actually OpenStack projects, they have to be pinned and tested on a regular basis too, since an older version may not be able to deploy the latest code. So you have CI of the tooling along with CI of the software it installs.

XCI DEVELOPER SANDBOX

In addition to the automated flows above, developers may want to locally develop and test based on the latest versions of the upstream projects. The XCI Sandbox allows developers to set up a scenario with the latest upstream code or code developed locally by them. Moreover, XCI Sandbox allows developers to do so on a single node, even a laptop. It does not require a full Pharos POD. Specifically, XCI Sandbox:

- Provides an automated way to set up a development and test environment
- Offers different flavors of environments
- Allows different versions of upstream components
- Allows a mechanism to enable or disable additional OpenStack or other upstream project services

Currently, the sandbox has four different flavors available (xci-aio, xci-mini, xci-noha, xci-ha) that consume from 1 VM to 6 VMs with increasing hardware requirements. As an example, the xci-ha flavor is the most resource-hungry and requires 8 vCPUs, 16GB RAM and 80GB of disk per VM, and it takes 2 hours and 10 minutes to install.

In summary, the XCI initiative extends the continuous aspect of the CI pipeline to upstream projects. While the initial upstream projects XCI will integrate are OpenStack, ODL and FD.io, in the future, the efforts will also be extended to other upstream projects such as ONAP.

REFERENCES

XCI wiki page: wiki.opnfv.org/pages/viewpage.action?pageId=8687635

XCI developer sandbox: wiki.opnfv.org/display/INF/XCI+Developer+Sandbox

Ansible information: ansible.com

Pharos wiki page: wiki.opnfv.org/display/pharos/Pharos+Home

OpenStack Bifrost information: docs.openstack.org/developer/bifrost

OpenStack Ansible information: docs.openstack.org/developer/openstack-ansible

XCI YouTube Video: https://www.youtube.com/watch?v=0X8nMVf48DM&t=1s