**OPNFV Releases**

**Summary**

OPNFV provides consumable releases every six months, these are made available around March and September respectively. A release of OPNFV provides an install-able datacenter reference platform, providing a variety of compositions and associated functions applicable for carrier network virtualization.

Users and developers should be provided an integrated, operational data-center build derived from open source projects which has undergone a basic level of testing. This should be able to be deployed on an OPNFV compliant infrastructure for the purpose of hosting applications and delivering services.

**The release process**

OPNFV releases provide a method for community projects to coordinate and release their code and features in a coordinated simultaneous release.

Projects intending to participate in a release need to communicate this intent to the release project during release planning. This process can be found described, along with other processes, on the [Release page](#).

**Stable Releases**

OPNFV releases are supported by a stable branch up according to the stable branch procedures.

Any project participating in an OPNFV release is expected to, and must commit to, provide stable release support of their contributions up until the next release of OPNFV is made available. A period of approximately 6 months.

Each major OPNFV release consists of an initial release (x.0), followed by two point releases (x.1, x.2). The point releases are intended to provide an opportunity to release scenarios that were failing deployment, or were not of sufficient quality, as of the initial release. However, the point releases may not introduce new functionality.

**Automated Deployment**

An OPNFV release will provide an automatic and repeatable deployment and installation to a pharos compliant configuration of servers. Offline deployment methods may also be provided for some deployment tools and deployment to virtual environments for development and evaluation may also be supported.

**OPNFV release projects**

**Iruya**

- [Iruya Release Page](#)

**Hunter**

- [Hunter Release Page](#)
- [Hunter Release Meeting](#)

**Gambia**

- [Gambia Release Page](#)
- [Gambia Release Meeting](#)
- [Gambia Release Meeting (APAC)](#)

**Fraser**

- [Fraser Release Page](#)
- [Fraser Release Meeting](#)
- [Fraser Release Meeting (APAC)](#)

**FAQ**

- How are different features/functions enabled for an installation?
- Several projects require specific components installed. How are their needs being met?
- What is the role of testing and test-documentation for the release?
- What constitutes an OPNFV release?
- How is an OPNFV release be maintained? Will there be specific service releases?
- How is it installed?
FAQ

How are different features/functions enabled for an installation?

A user can choose the features to be installed for a particular installation. Feature selection and the associated configuration of the features is done via a set of configuration files which are common for all installers (“install.yaml” – for general configuration and feature selection, “feature_x.yaml” – for configuration specific to a “feature_x”, like for example a SDN-controller like OpenDaylight). Note that not all features/functions are install-time features. Installers only take care of install-time features. Note that installers only install a particular feature or component, i.e. they don’t take any responsibility for the appropriate operation of a feature, component, or function.

Several projects require specific components installed. How are their needs being met?

Projects which are interested in a particular combination of features and functionality are expected to supply the configuration files (i.e. the specific versions of “install.yaml” and “features.yaml”) they require to the deployment project or projects of preference. In addition, the projects are to supply a set of system tests (typically as part of one of the testing projects like FuncTest, Yardstick, etc.) to verify their functionality, leveraging the particular combination installed – along with a description of what “successful testing” of the feature means. Note that as part of a test, additional features and/or configuration to the installer-supplied setup may need to be added (“post-install”). Using OpenDaylight as an example: OpenDaylight might only come configured with a set of base features by the installation process. Additional features that specific project testing requires would be installed by the test itself, i.e. through feature:install in karaf. In summary, a deployment can be phased into 3 steps - typically followed by testing, which would be the 4th step:

- **Hardware configuration** (so that an installer can execute successfully): Pharos system description and associated hardware configuration scripts supply for this.
- **Base system installation**: One of the deployment tools (“installer”) supplies for this. If a project has a specific feature/component that needs to be installed as part of base system setup, the project needs to engage with the installer projects to get its feature installed/configured. Features and functions of the OPNFV platform fall into this category. See also: Is my feature an install-time or post-install-time feature?.
- **Post base-install feature installation/configuration**: Per the above, several projects might have the need to perform project specific installation and/or configuration (often required by test projects).
What is the role of testing and test-documentation for the release?

Test projects (i.e. Functest, Yardstick, etc.) supply a set of tests for a release. OPNFV testing adds iteratively to the existing test suites available in the OPNFV each release covering new features and capabilities in the platform. Tests which are to be run on an installation are defined by a set of test-configuration scripts. Test configuration scripts define which test framework is used (e.g. tempest, robot, yardstick, ...), which tests are executed, how these tests are configured, and eventually even where the tests are to be executed (e.g. in case a certain hardware dependency is given). Tests are automatically executed as part of the CI/CD pipeline. Test results along with the test-system definition and test descriptions are archived in a test-database which documents all test runs. Projects can add project-specific tests to the existing set of tests supplied by the testing projects. Those tests could be run exclusively for a particular deployment/installation (e.g. specific L3VPN tests would only be run on an installation which has L3VPN features installed and configured). The configuration of these tests adds to the above described test-configuration.

What constitutes on OPNFV release?

A release is constituted through a set of deployments (scenarios and configurations), the associated tests (defined by the test configuration) and the corresponding test results. Note that the set of combinations tested follows interest: For a release not all possible feature/component and deployment configurations will be tested. Testing will follow the interest (and associated invested effort) of the projects participating, rather than try to test any possible combination.

How is an OPNFV release be maintained? Will there be specific service releases?

An OPNFV release is maintained on a single stable release branch.

Typically an x.1 and x.2 version of the release is planned for coordinated improvement and fault correction. Once the planned stable releases are completed the release is maintained in a stable state on the stable branch where the latest stable version is always available.

How is it installed?

OPNFV installation will be facilitated by an install script which is used to install the installer on a "jumphost". Once operational, the installer will automatically install the overall OPNFV system on the hosts chosen for the installation.