OPNFV

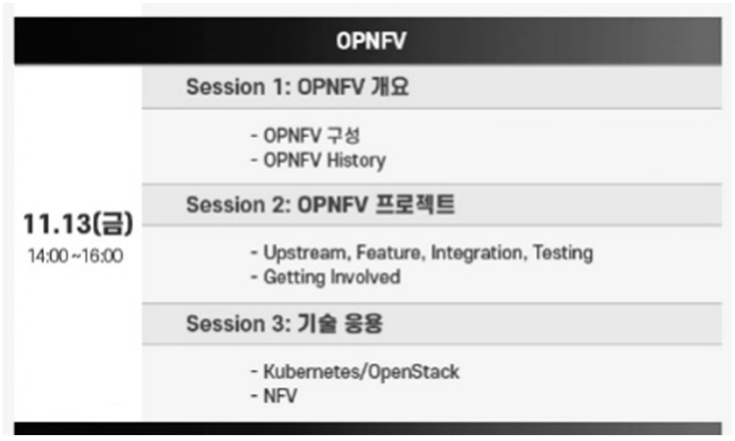
(**O**pen **P**latform for **NFV**)

2020. 11.

2021년 5월까지 사용을 권장 합니다.

안 종 석
james@jslab.kr
JS Lab

james@jslab.kr



JS Lab

목차

james@jslab.kr

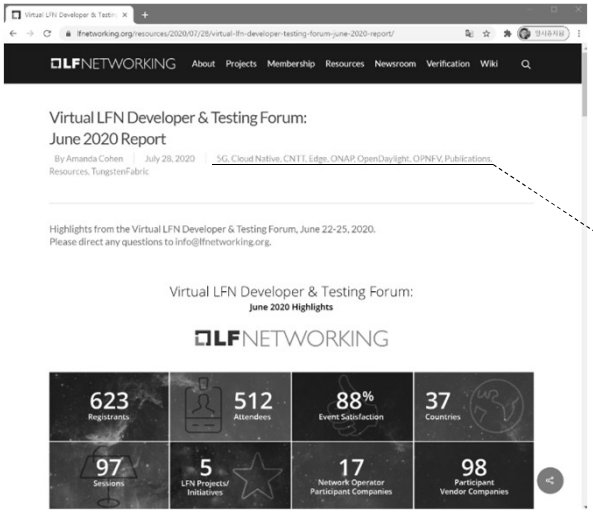
1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

james@jslab.kr

JS Lab

1. 개요

❖ **OPNFV @ LF Networking**



5G,
Cloud Native,
CNTT,
Edge,
ONAP,
OpenDaylight,
OPNFV,
Publications,

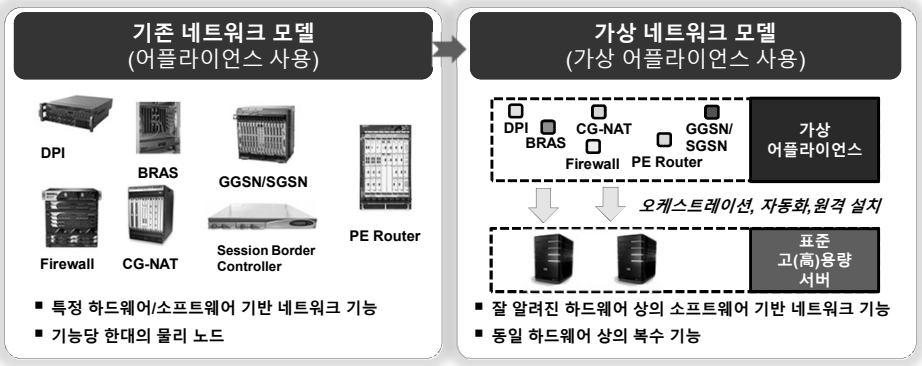
https://www.lfnetworking.org/resources/2020/07/28/virtual-lfn-developer-testing-forum-june-2020-report/

james@jslab.kr

1. 개요

❖ **NFV 개념: 하드웨어 의존성을 최소화 하는 네트워크 가상화로 독립적이며 유연하고 단순한 네트워크를 만드는 도구 제공**

❖ **NFV(Network Functions Virtualization) 표준을 주도하고 있는 ETSI NFV ISG는 2012년 설립 이후 매 분기별로 총회 및 기술 분과 회의를 진행**

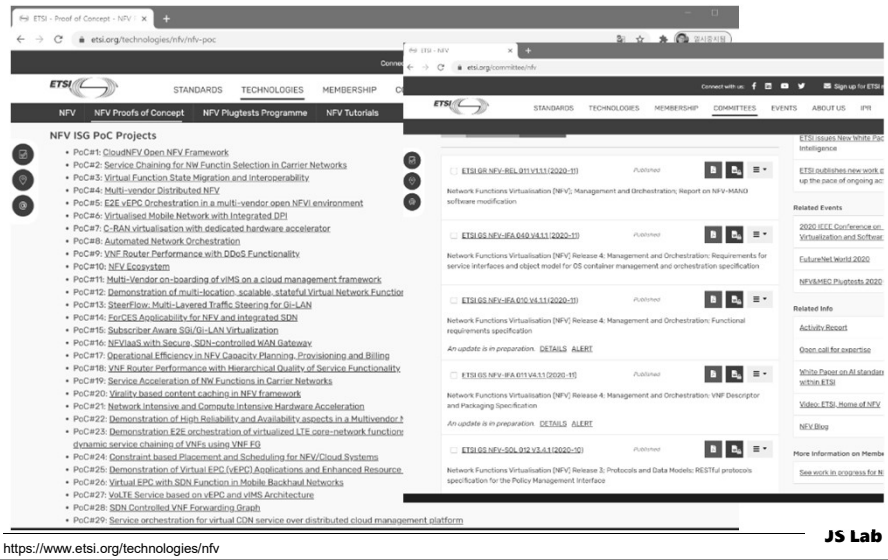


https://www.etsi.org/technologies/nfv

james@jslab.kr

1. 개요

❖ ETSI의 NFV 활동: PoC, 기술 문서, Plugtests Programme 등

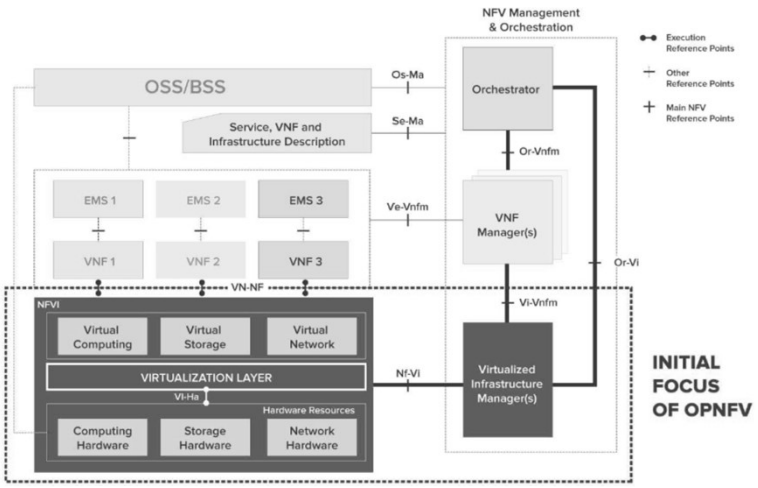


https://www.etsi.org/technologies/nfv

james@jslab.kr

1. 개요

❖ NFV reference architecture framework



ETSI GS NFV 002 v1.2.1 (2014-12)

james@jslab.kr

1. 개요

❖ **OPNFV** (**O**pen **P**latform for **NFV**)

- **통신 사업자 주도로 2014년 10월 출범**
- **NFV (Network Function Virtualization)를 사용하는 신제품이나 서비스를 빠르게 도입 할 수 있도록 하기 위한 캐리어급 통합 오픈 소스 플랫폼** (Carrier-Grade, Integrated, Open Source Platform)
- **Linux 재단의 협업 프로젝트** (Linux Foundation Collaborative Project)

THE LINUX FOUNDATION PROJECTS

james@jslab.kr

https://www.opnfv.org/

1. 개요

❖ **OPNFV Mission**

- **다양한 오픈 소스 생태계 전반의 NFV 구성 요소 개발 및 발전을 촉진**
- **시스템 레벨 통합, 배포 및 테스트를 통해 엔터프라이즈 및 서비스 사업자 네트워크의 변환을 가속화하는 NFV 플랫폼 레퍼런스를 생성**

james@jslab.kr

https://www.opnfv.org/about/mission

1. 개요

❖ **Goals for OPNFV**

- **오픈 소스 플랫폼 개발:** NFV 기능 구축에 사용할 수 있는 통합적이고 검증 된 오픈 소스 플랫폼 개발 (새로운 제품 및 서비스의 도입 가속화)
- **사용자의 참여:** OPNFV가 사용자 커뮤니티의 요구 사항을 충족하는지 검증하기 위해 선도적 인 최종 사용자의 참여 포함
- **운용성 확보:** OPNFV 플랫폼에서 활용 될 관련 오픈 소스 프로젝트 구성 요소 간의 일관성, 성능 및 상호 운용성 확보
- **생태계 구축:** 최종 사용자의 요구를 충족시키기 위한 개방형 표준 및 소프트웨어에 기반한 NFV 솔루션을 위한 생태계 구축
- **홍보:** 오픈 소스 NFV를 위한 선호 플랫폼 및 커뮤니티로서 OPNFV 홍보

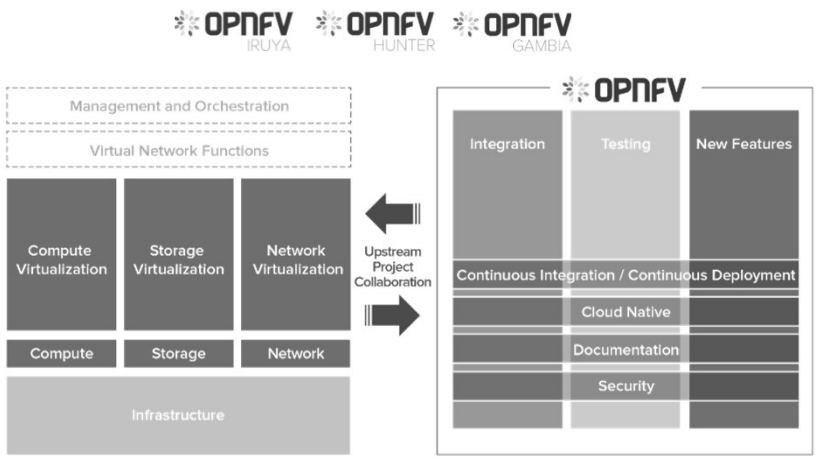
james@jslab.kr

https://www.opnfv.org/about/mission

1. 개요

❖ **OPNFV Current Release: Iruya 9.0**

- **Architectural view**



james@jslab.kr

https://www.opnfv.org/software

1. 개요

❖ **Release History**

❖ **OPNFV provides consumable releases every six months**

Release: Hunter
Version: 8.1
Release Date: 7/09/2019
View Release Details
Release: Hunter
Version: 8.0
Release Date: 5/14/2019
View Release Details

Release: Gambia
Version: 7.2
Release Date: 1/28/2019
View Release Details
Release: Gambia
Version: 7.1
Release Date: 12/14/2018
View Release Details
Release: Gambia
Version: 7.0
Release Date: 11/13/2018
View Release Details

Release: Fraser
Version: 6.2
Release Date: 6/29/2018
View Release Details
Release: Fraser
Version: 6.1
Release Date: 5/30/2018
View Release Details
Release: Fraser
Version: 6.0
Release Date: 5/1/2018
View Release Details

Release: Euphrates
Version: 5.1
Release Date: Monday, 12/18/2017
View Release Details
Release: Euphrates
Version: 5.0
Release Date: Tuesday, 10/24/2017
View Release Details

Release: Danube
Version: 3.0
Release Date: 7/17/2017
View Release Details
Release: Danube
Version: 2.0
Release Date: 5/5/2017
View Release Details
Release: Danube
Version: 1.0
Release Date: 4/4/2017
View Release Details

Release: Colorado
Version: 3.0
Release Date: 11/14/2016
View Release Details
Release: Colorado
Version: 2.0
Release Date: 11/14/2016
View Release Details
Release: Colorado
Version: 1.0
Release Date: 09/26/2016
View Release Details

Release: Brahmaputra
Version: 3.0
Release Date: 04/29/2016
View Release Details
Release: Brahmaputra
Version: 2.0
Release Date: 04/21/2016
View Release Details
Release: Brahmaputra
Version: 1.0
Release Date: 03/01/2016
View Release Details

Release: Arno
Version: SR1
Release Date: 01/01/2015
View Release Details
Release: Arno
Version: 1.0
Release Date: 06/04/2015
View Release Details

james@jslab.kr

1. 개요

❖ **OPNFV의 프로젝트 구분**

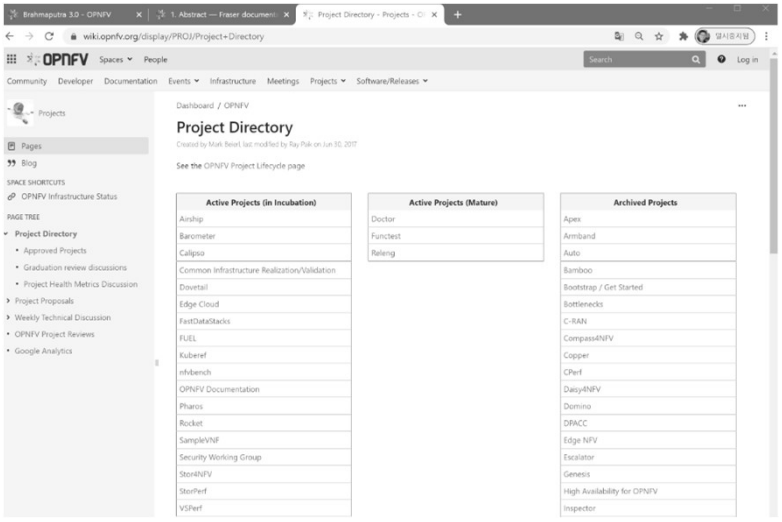
- **요구사항 프로젝트 (Requirements)**
  - ✓ NFV Reference Platform 개발을 위해 필요한 요구사항에 대해 수집 및 문서화를 위한 프로젝트
  - ✓ 정의된 요구사항은 OPNFV Community 나 Upstream Project 에서 구현
- **통합 및 검증 프로젝트 (Integration & Testing)**
  - ✓ NFV Reference Platform 구축에 사용되는 다양한 Open Source Project (e.g. K8s, OpenStack, OpenDaylight)에 대한 통합 및 검증 수행을 위한 프로젝트
- **협력개발 프로젝트 (Collaborative Development)**
  - ✓ NFV Reference Platform 개발을 위해 다른 Open Source Project 그룹 및 표준화 단체등과 협력 개발을 위한 프로젝트
- **문서화 프로젝트 (Documentation)**
  - ✓ OPNFV 와 관련된 문서화 및 문서 작성 도구 개발을 위한 프로젝트

https://wiki.opnfv.org/display/PROJ

james@jslab.kr

1. 개요

❖ **OPNFV의 프로젝트 디렉토리**



https://wiki.opnfv.org/display/PROJ

1. 개요

❖ **OPNFV Iruya: Paving the Way for 5G via Testing Tools, Scripts, and Automation** (By Heather Kirksey, February 4, 2020)

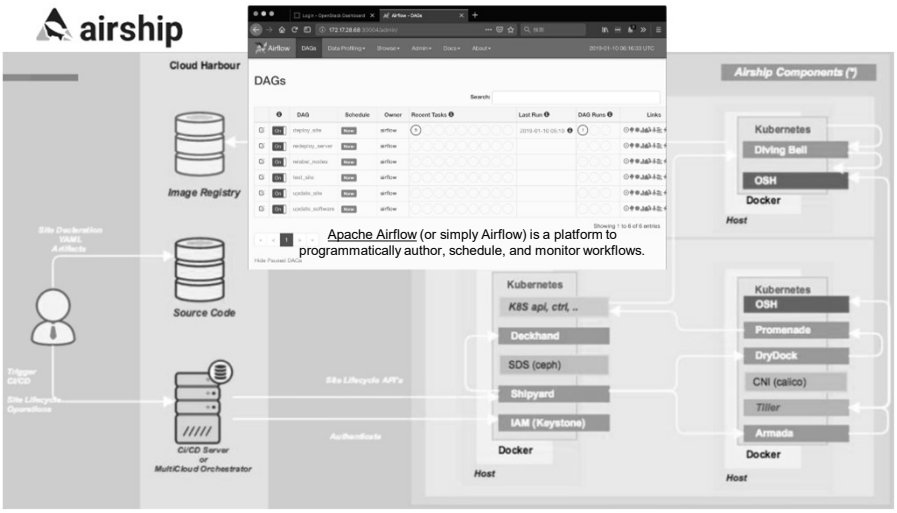
Current Release: Iruya 9.0

#Airship
#easyCLA
#CNTT

https://www.opnfv.org/blog/2020/02/04/opnfv-iruya-paving-the-way-for-5g-via-testing-tools-scripts-and-automation

1. 개요

❖ **Airship**

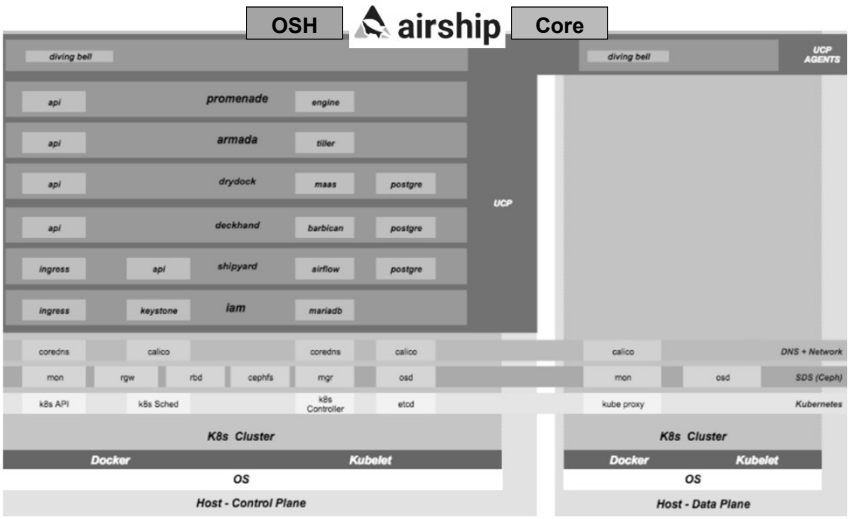


https://www.airshipit.org/

james@jslab.kr

1. 개요

❖ **구성 'Airship Integration Documentation'** (2020년 10월)

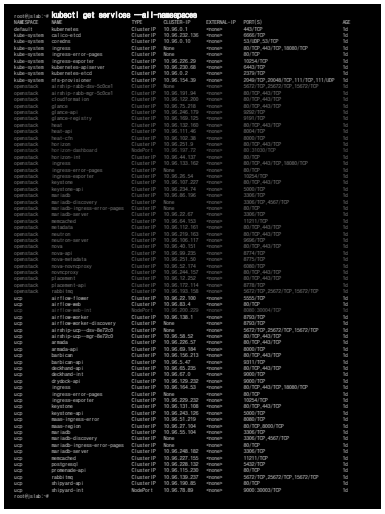


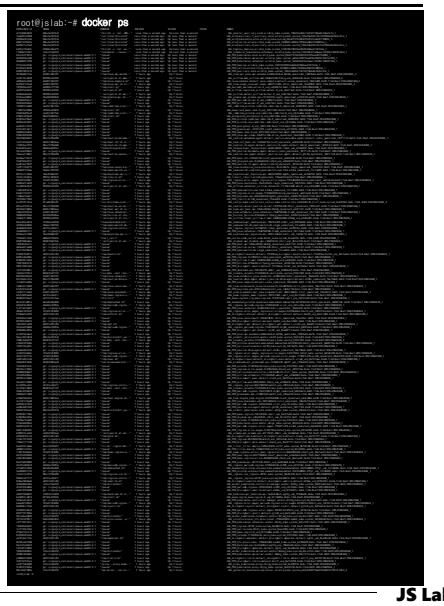
https://readthedocs.org/projects/airship-treasuremap/downloads/pdf/latest/

james@jslab.kr

1. 개요

❖ Airship in a Bottle





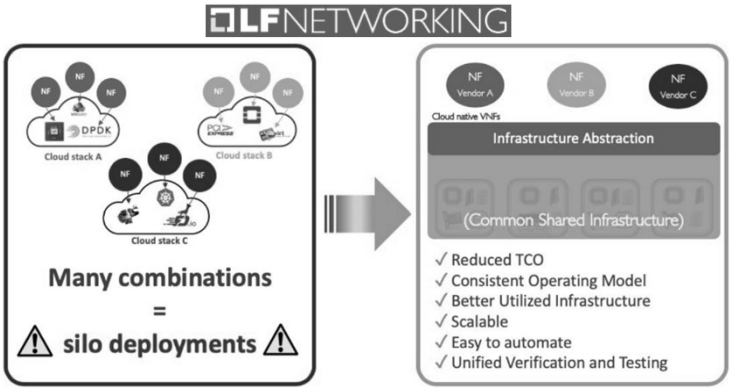
https://github.com/airshipit/airship-in-a-bottle

james@jslab.kr

1. 개요

❖ Cloud iNfrastructure Telco Taskforce (CNTT)

- **Cloud iNfrastructure Telco Taskforce (CNTT) 은 2019년 설립**
- **partnership between LF Networking and GSMA:** provides standardized infrastructures for both virtual machine-based and containerized network functions, enabling providers to shorten onboarding from weeks and months to hours and days, reducing costs, and accelerating digital transformation.

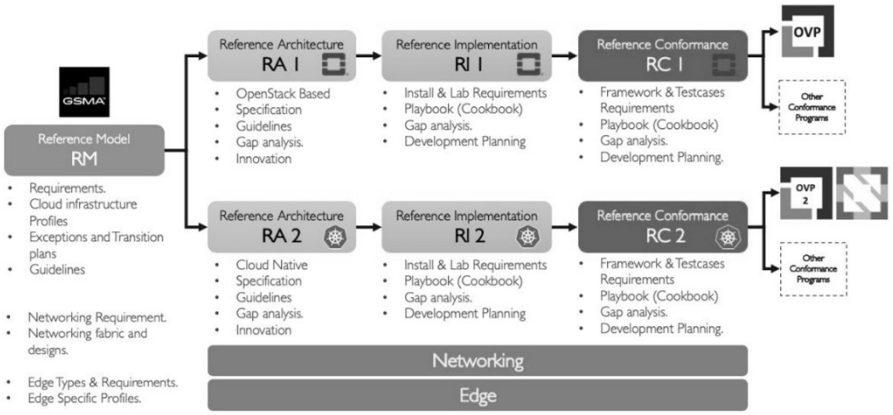


#5G, #Cloud Native, #CNTT, #Edge, #OPNFV, #Publications, #Resources

james@jslab.kr

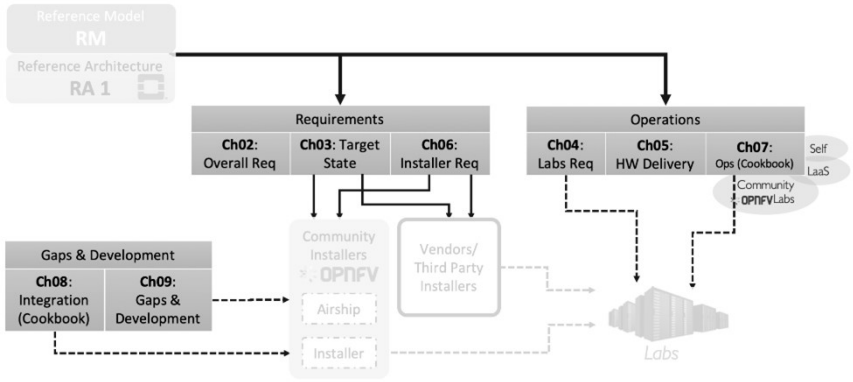
1. 개요

❖ **CNTT Reference framework, Workstreams and Focus Groups**



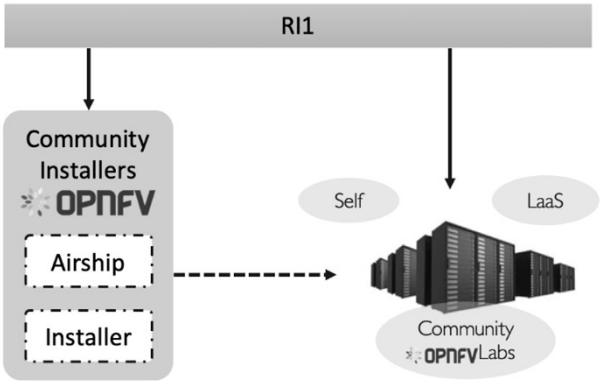
1. 개요

❖ **CNTT Reference Architecture**



1. 개요

❖ **CNTT Reference Implementation**



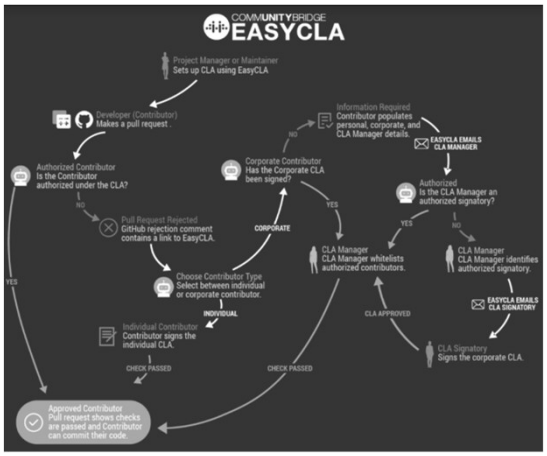
https://cntt-n.github.io/CNTT/doc/ref_impl/cntt-ri/chapters/chapter01.html

james@jslab.kr

1. 개요

❖ **EasyCLA is an Automated, Efficient, and Flexible Tool for Managing Contributor Agreements and Repository Access**

Open Networking Summit (ONS) North America, the Linux Foundation announced Community Bridge, "… a new platform created to empower open source developers.." Community Bridge is a suite of tools that enable individuals and organizations to engage with open source projects.

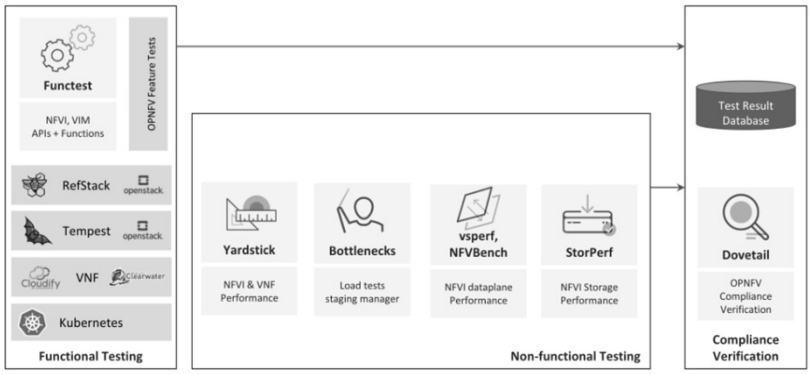


https://github.com/communitybridge/easycla/tree/master/getting-started

james@jslab.kr

1. 개요

❖ **The Open Platform for NFV (OPNFV) community spent about four years developing testing tools that can also help other open-source projects.**

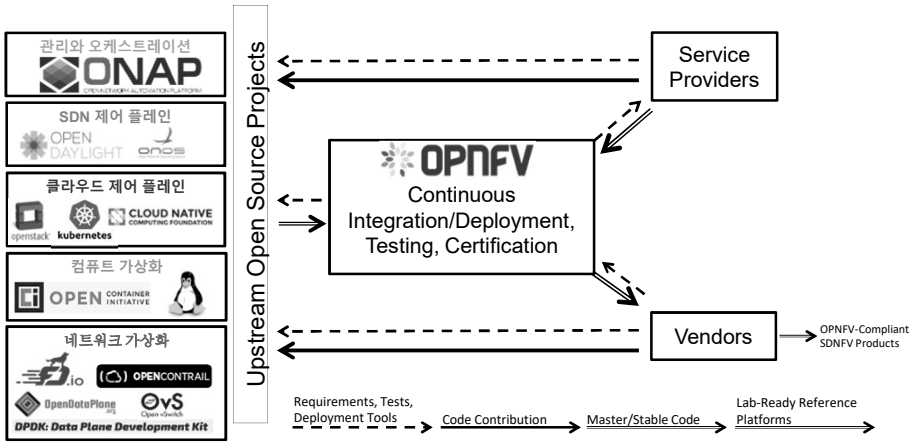


https://superuser.openstack.org/articles/opnfv-testing-tools/

james@jslab.kr

1. 개요

❖ **OPNFV Interaction with Upstream Communities and Downstream Projects/Users**



james@jslab.kr

1. 개요

❖ Developer Tools

1. 개요

❖ Current Release: Iruya 9.0

- Cross-community Collaboration (CNTT & OVP)
- Test Projects
  - ✓ Functest
  - ✓ Yardstick
  - ✓ VSPerf
  - ✓ Bottlenecks
  - ✓ NFVBench
- Edge Computing
  - ✓ Rocket
  - ✓ Edge Cloud
  - ✓ Container4NFV
- Additional Enhancements
  - ✓ Barometer
  - ✓ Doctor
  - ✓ LaaS
- Community Bridge Progress

1. 개요

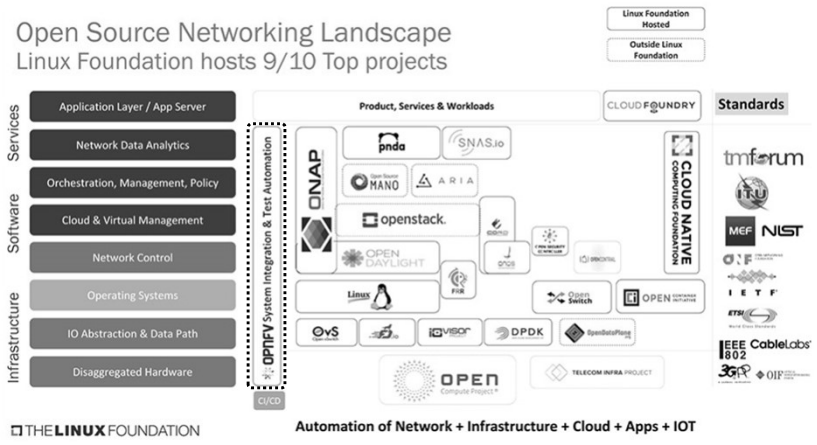
❖ **Last Release: Hunter 8.0**

- **Integration and Test Projects**
  - ✓ **XCI:** Cross-Community Continuous Integration
  - ✓ **Xtesting:** OPNFV now also offers a test framework
  - ✓ **LaaS:** Lab as a Service
  - ✓ **Pharos project:** continues to make progress on specifying an edge POD
  - ✓ **Functest:** functional testing, (OpenStack Rocky / k8s v1.13.5)
  - ✓ **Yardstick:** performance testing, includes additional support for k8s testing
  - ✓ **Bottlenecks:** used for stress and longevity testing
  - ✓ **VSPerf:** used for virtual switch performance characterization
  - ✓ **NFVBench:** used for NFVI data plane performance testing
  - ✓ **SampleVNF:** that provides open source VNF approximations
- **Cross-community Collaboration**
  - ✓ **C-RAN:** A new project relating to performance measurement for Cloud Radio Access Network (C-RAN) VNF
  - ✓ **Rocket:** Also a new project, Rocket aims at improving virtual Evolved Packet Core (vEPC) VNF performance
  - ✓ **Other Enhancements:** container-based undercloud services, IPv6, OpenStack, NFVI telemetry (ONAP)

james@jslab.kr

1. 개요

❖ **Open Source Networking and OPNFV**



james@jslab.kr

james@jslab.kr

1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

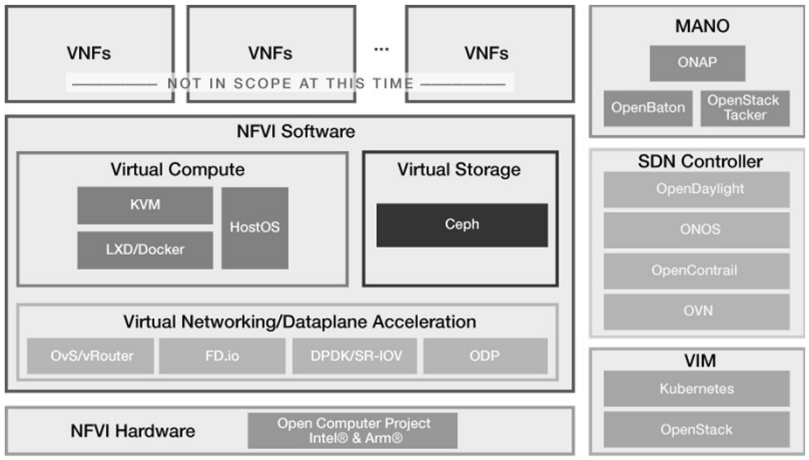
❖ 업스트림(Upstream) OPNFV 커뮤니티(Community)

- Do not look to fork upstream projects or create OPNFV specific versions
- Analysis of NFV requirements or testing activities, we discover gaps, desired features, or bugs, we will participate in the appropriate community process of the upstream organization to incorporate blueprints, patches, and other changes.

https://www.opnfv.org/community/upstream-projects

2. 업스트림(Upstream) 프로젝트 통합

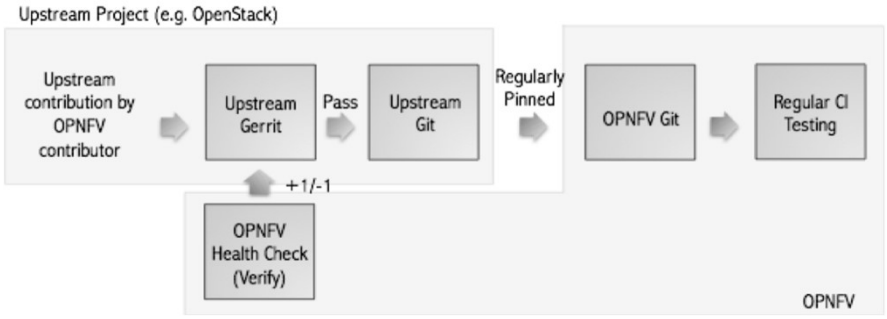
❖ **Upstream Software Projects Integrated by OPNFV**



https://www.opnfv.org/community/upstream-projects https://wiki.opnfv.org/display/COM/Upstream+Project+Mapping

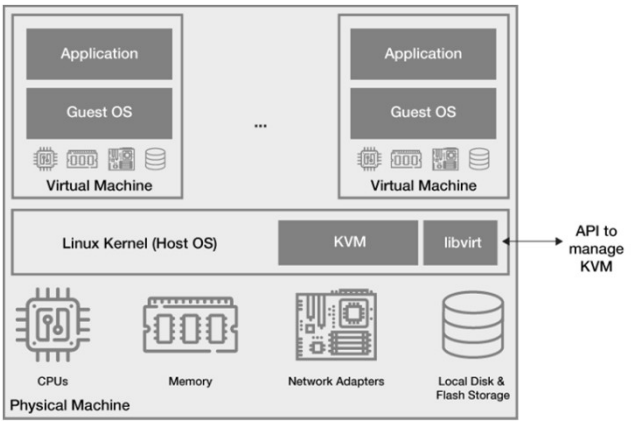
2. 업스트림(Upstream) 프로젝트 통합

❖ **Upstream Project (예: OpenStack)**



2. 업스트림(Upstream) 프로젝트 통합

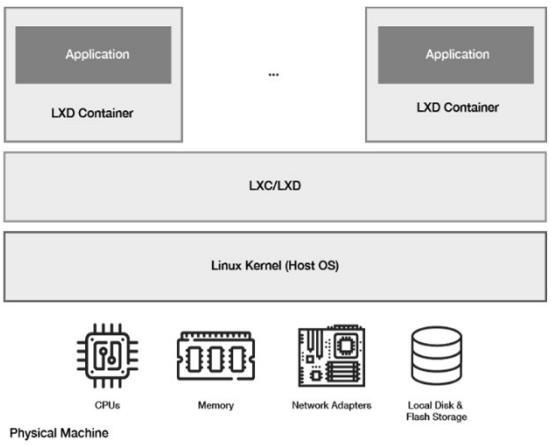
❖ NFVI Software: Virtual Compute: KVM



james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

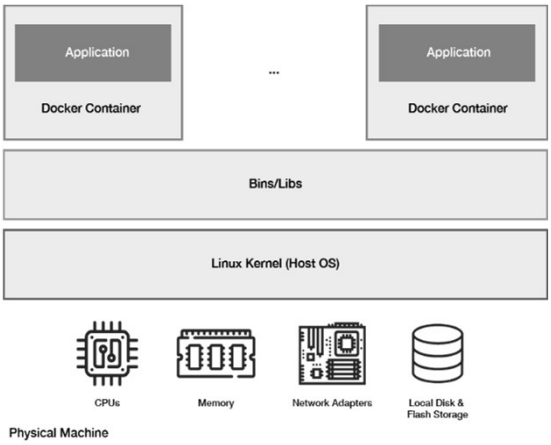
❖ NFVI Software: Virtual Compute: LXD



james@jslab.kr

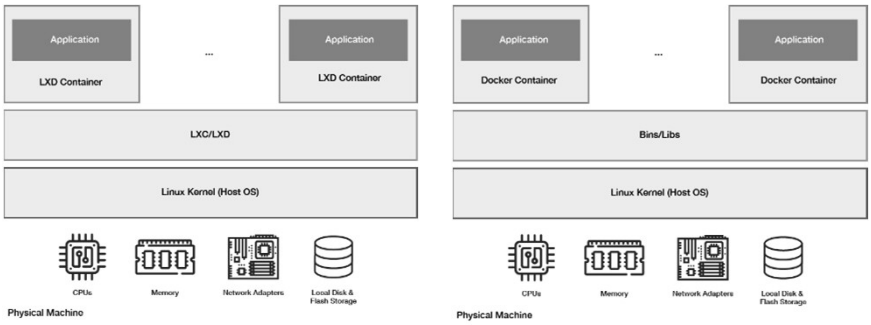
2. 업스트림(Upstream) 프로젝트 통합

❖ NFVI Software: Virtual Compute: Docker



2. 업스트림(Upstream) 프로젝트 통합

❖ LXD Container Architecture / Docker Container Architecture



LinuxFoundationX: LFS164x

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Virtual Storage**

- three types of virtual storage solutions: **block, file, and object.**
- However, for **NFV, block storage is generally sufficient**. For this reason, the only virtual storage solution integrated is a project called **Ceph** for block storage.
- Ceph's foundation is the **Reliable Autonomic Distributed Object Store (RADOS)**, which provides your applications with object, block, and file system storage in a single unified storage cluster

james@jslab.kr

https://ceph.com/ceph-storage/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Virtual Storage: Ceph**

- **SDS(software defined storage) that scales horizontally by adding storage servers**
- **The software was open sourced by Red Hat**
- According to the OpenStack 2016 User Survey, Ceph is the most popular external block storage software used in OpenStack deployments, For this reason, **Ceph is also integrated with OPNFV**.
- **Ceph's official website**:
  - ✓ "Ceph is a unified, distributed storage system designed for excellent performance, reliability, and scalability. Ceph's foundation is the Reliable Autonomic Distributed Object Store (RADOS), which provides your applications with object, block, and file system storage in a single unified storage cluster — making Ceph flexible, highly reliable and easy for you to manage".

james@jslab.kr

https://ceph.com/

2. 업스트림(Upstream) 프로젝트 통합

❖ NFVI Software: Virtual Network

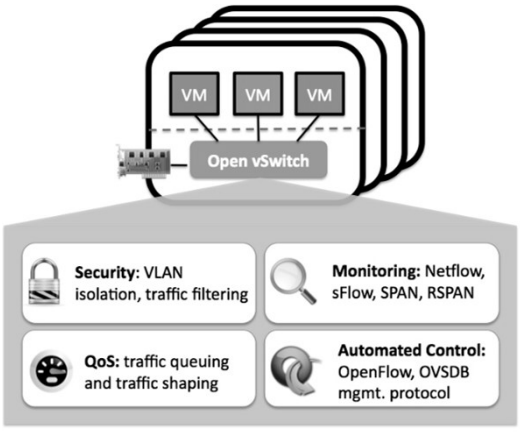
- Open vSwitch (OVS)
- FD.io (The Fast Data Project)
- Tungsten Fabric vRouter

james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

❖ NFVI Software: Virtual Network: OVS

- OVS Architecture

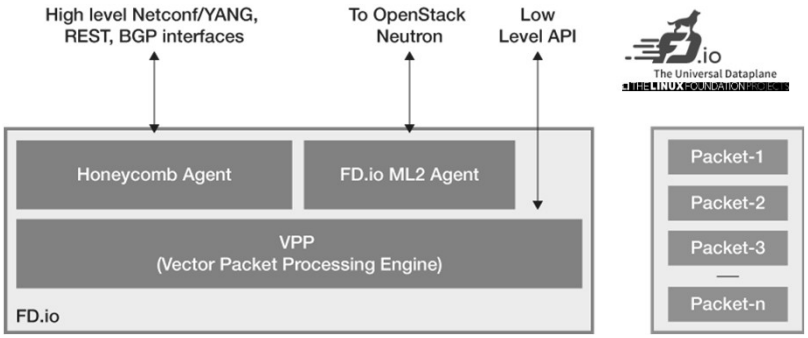


http://openvswitch.org/

james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

- ❖ **NFVI Software: Virtual Network: FD.io**
- ❖ **A high-performance alternative to OVS, the core engine of The Fast Data Project, is a Vector Packet Processing (VPP) engine that came from Cisco.**



james@jslab.kr

https://wiki.fd.io/view/Main_Page

2. 업스트림(Upstream) 프로젝트 통합

- ❖ **NFVI Software: Virtual Network: FD.io**
- ▪ **FD.io Project**
  - ✓ Honeycomb
  - ✓ CSIT
  - ✓ NSH SFC
  - ✓ ONE
  - ✓ VPP Sandbox
  - ✓ TLDK
  - ✓ Deb dpdk
  - ✓ Ci-management
  - ✓ Rpm dpdk
  - ✓ Puppet-fdio
  - ✓ Hc2vpp
  - ✓ TRex
  - ✓ Cicn
  - ✓ GoVPP
  - ✓ ODP4VPP
  - ✓ P4vpp
  - ✓ Pma tools
  - ✓ DMM
  - ✓ Sweetcomb
  - ✓ HICN
  - ✓ JVPP

james@jslab.kr

https://wiki.fd.io/view/Main_Page

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration**

- **When moving network functions from physical to virtual, there is generally a packet performance loss as additional components (e.g. hostOS, hypervisor, OVS) get inserted in the packet-handling path.**
- **Data acceleration technologies optimize packet processing to compensate for this performance loss. OPNFV integrates two dataplane acceleration technologies:**
  - ✓ Data Plane Development Kit (DPDK) and related functionality
  - ✓ OpenDataPlane.

james@jslab.kr

http://openvswitch.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: DPDK**

- **The Data Plane Development Kit (DPDK), The Linux Foundation project, is a set of libraries that bypass the kernel and provide polling mechanisms, instead of interrupt-based operations, to speed up packet processing.**
- **In the 2015 study by Intel: "Using Open vSwitch with DPDK for Inter-VM NFV Applications", OVS with DPDK showed a 75% improvement in throughput over plain OVS.**
- **Other related technologies are huge pages, Non-Uniform Memory Access (NUMA) pinning, Single Root I/O Virtualization (SR-IOV), and cache optimization technologies.**
- **Huge pages improve VNF efficiency by reducing page lookups; NUMA pinning ensures that the workload uses memory local to the processor; SR-IOV enables network traffic to bypass the hypervisor and go directly to the virtual machine; and cache optimizations improve overall efficiency.**

james@jslab.kr

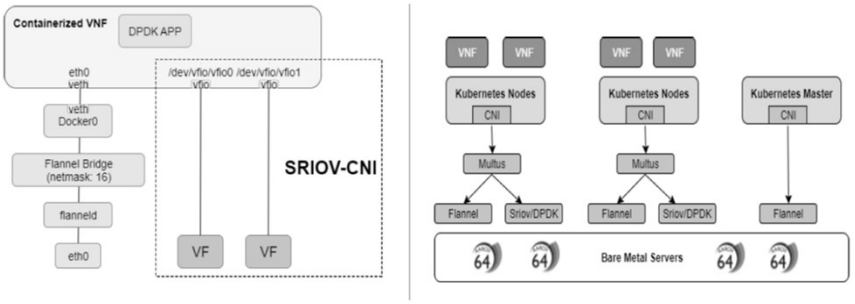
http://www.dpdk.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: DPDK**

- **SRIOV Passthrough**

james@jslab.kr



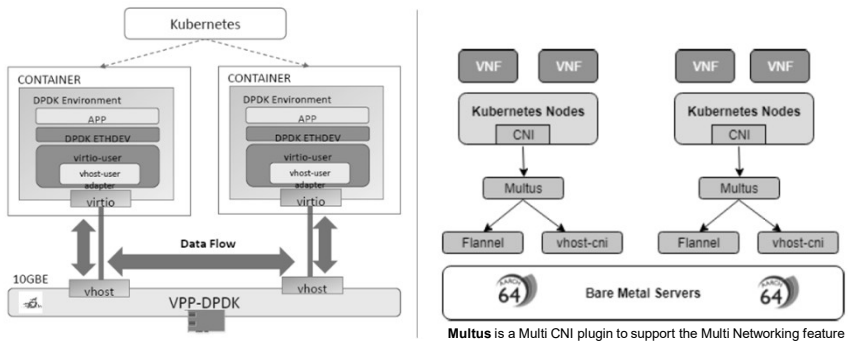
http://openvswitch.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: DPDK**

- **VPP: open-source Vector Packet Processing (VPP) platform**
  - ✓ Features: IPv4/IPv6 L2, MPLS, VLAN, Bridge, NAT, ACL, VxLAN, GRE 등
  - ✓ Integrated: OpenStack/ODL(netconf/Yang), K8s/Flanel (Python API),

james@jslab.kr



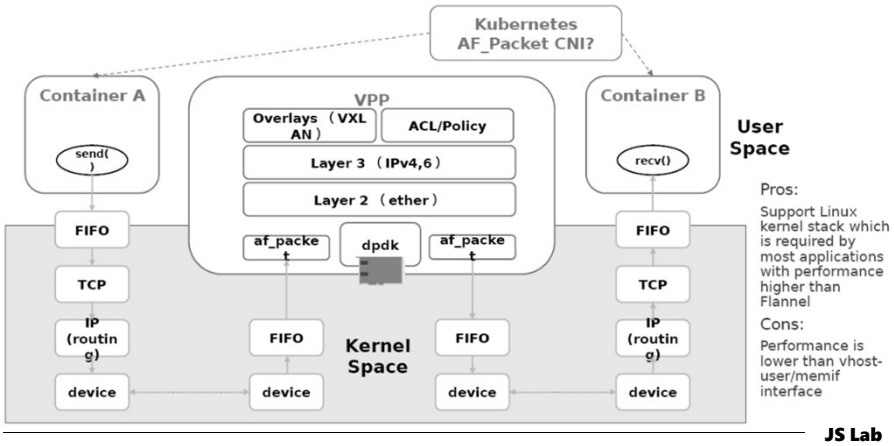
Multus is a Multi CNI plugin to support the Multi Networking feature

https://wiki.fd.io/view/VPP https://wiki.fd.io/view/VPP/Features

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: DPDK**

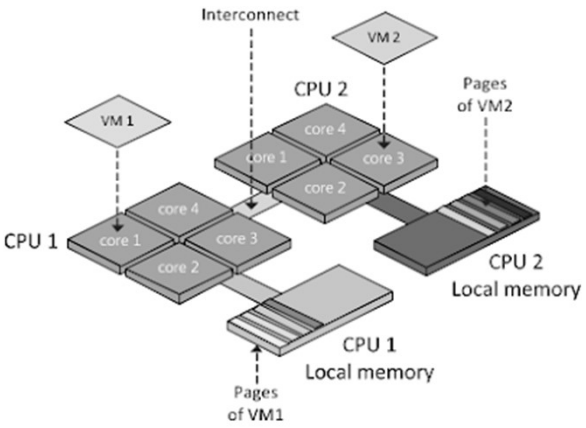
- **VPP for Container Networking**

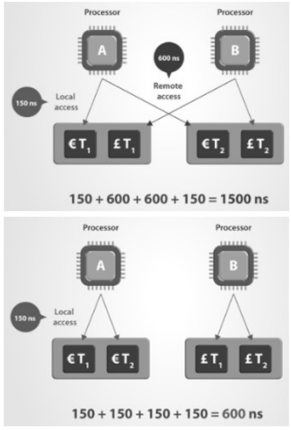


2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: DPDK**

- **NUMA** (**N**on-**U**niform **M**emory **A**ccess) **pinning or aware**

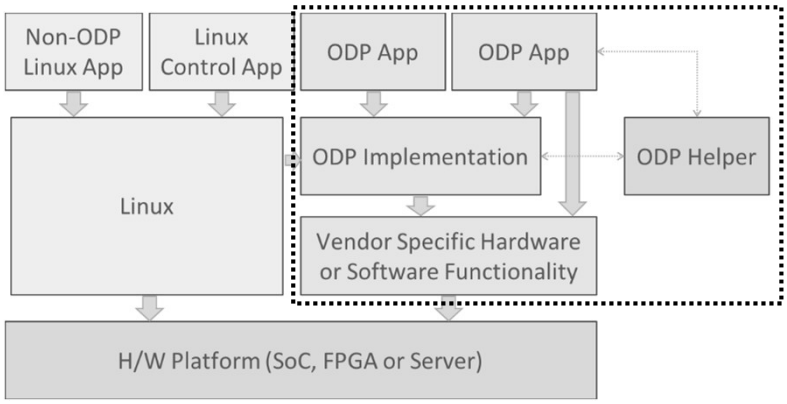




http://openstack-in-production.blogspot.com/2015/08/numa-and-cpu-pinning-in-high-throughput.html

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Dataplane Acceleration: ODP** (**O**pen**D**ata**P**lane)



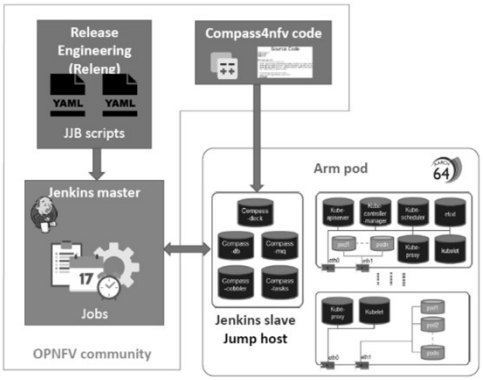
james@jslab.kr

https://www.opendataplane.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Hardware**

- **OPNFV tests its software against industry standard Intel® and ARM® servers.**
- **Most of these servers are proprietary, though one notable open source hardware project is the Open Compute Project (OCP).**



james@jslab.kr

http://www.opencompute.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ Virtualized Infrastructure Manager (VIM)

- Virtualizing hardware resources is the first step, but the infrastructure then needs to be "cloudified" to provide elastic, on-demand, API-driven, self-service infrastructure.
- The VIM is responsible for creating and managing infrastructure resources, such as virtual compute and virtual storage.
- The VIM can work with an SDN controller (more details will be provided later, in the SDN Controller section) to create virtual networks. (Strictly speaking, SDN controllers are part of the VIM, but we are breaking them out for the sake of clarity.)
- OPNFV utilizes OpenStack more so than any other VIM, though Kubernetes is an emerging alternative.

james@jslab.kr

http://www.opencompute.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ VIM: OpenStack

- OpenStack is an open source software for creating private and public clouds.
- OpenStack software controls large pools of compute, storage, and networking resources throughout a datacenter, managed through a dashboard or via the OpenStack API.
- This is one of the largest open source projects, and the Pike release had 1,825 contributors. In a recent Heavy Reading survey, 86% of global telecoms stated that OpenStack is either important or essential to their success, and over 60% are also using or testing OpenStack for NFV.
- Perhaps more interesting, over 21% of those surveyed plan to install OpenStack as part of OPNFV!
- In terms of compute resources, while OpenStack supports bare-metal servers, virtual machines, and containers, it is most often used with virtual machines.

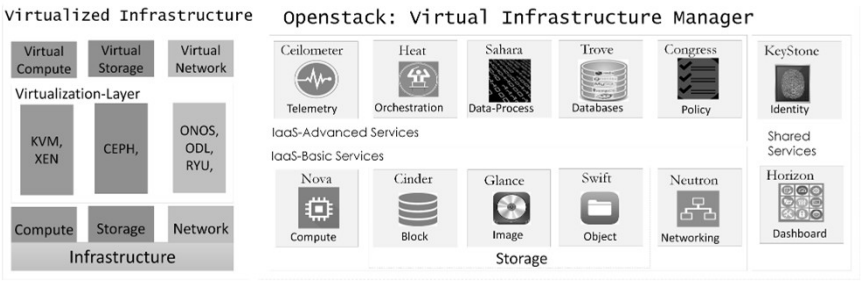
james@jslab.kr

http://www.opencompute.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ VIM: OpenStack

- OPNFV's initial focus is on virtualized infrastructure. Considering the use of OpenStack as the virtual infrastructure manager, the below figure depicts the initial focus on OPNFV.



https://thenewstack.io/opnfv-operationalizes-network-functions-virtualization/

2. 업스트림(Upstream) 프로젝트 통합

❖ VIM: Kubernetes

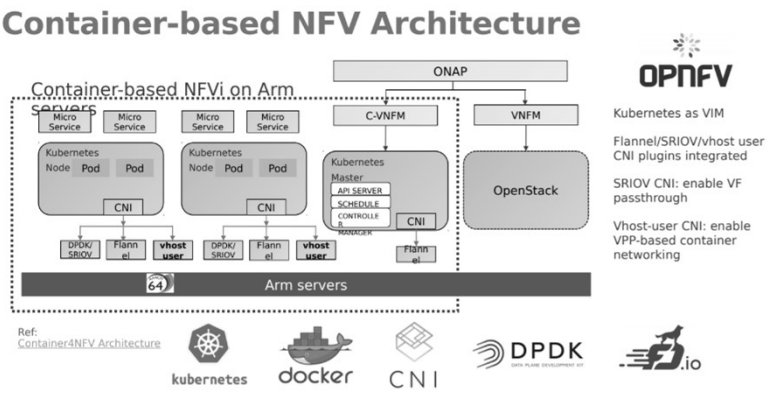
- Kubernetes is an open source system for automating deployment, scaling, and managing containerized applications.
- It groups containers that make up an application into logical units for easy management and discovery.
- Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.
- In July 2015, Google donated Kubernetes to the Cloud Native Computing Foundation, which is part of The Linux Foundation'

https://kubernetes.io/

2. 업스트림(Upstream) 프로젝트 통합

❖ **VIM: Kubernetes**

- **K8s as VIM @ Container-based NFV Architecture**



https://wiki.opnfv.org/login.action?os_destination=%2Fpages%2Fviewpage.action%3FspaceKey%3DOpenRetriever%26title%3DContainer%2527s%2BArchitecture%2Bfor%2BCloud%2BNative%2BNFV&permissionViolation=true

james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

❖ **Software-Defined Networking (SDN) Controller**

- **SDN controllers offer numerous benefits:**
  - ✓ **The entire control plane is centralized for ease-of-management; it can still be distributed for HA and scaling.**
  - ✓ **Policies are centralized, so conflicts are eliminated.**
  - ✓ **Rapid deployment of network-related changes through the centralized control plane is manageable using scripting and other programmatic means.**
  - ✓ **Event-based reconfiguration of networking elements is possible.**
- **OPNFV integrates the following SDN controllers:**
  - ✓ **OpenDaylight**
  - ✓ **OpenContrail**
  - ✓ **ONOS**
  - ✓ **OVN**

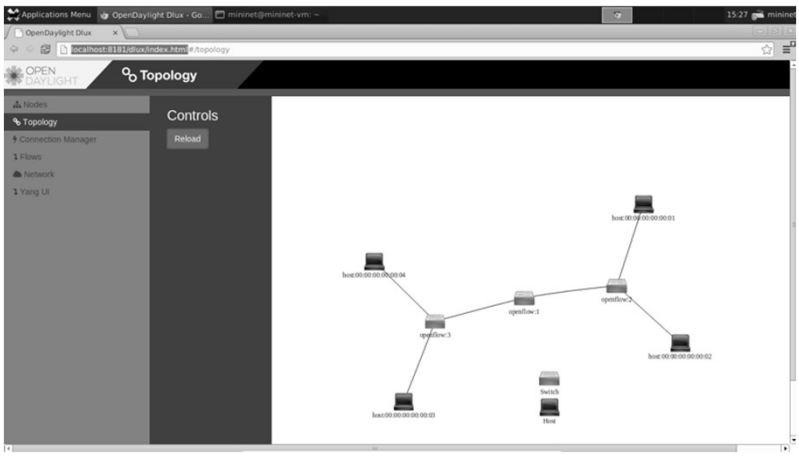
https://kubernetes.io/

james@jslab.kr

2. 업스트림(Upstream) 프로젝트 통합

❖ **SDN Controller: OpenDaylight:**

- **The OpenDaylight Foundation was founded in 2013, and is a founding member of The Linux Foundation Networking Fund (LFN).**

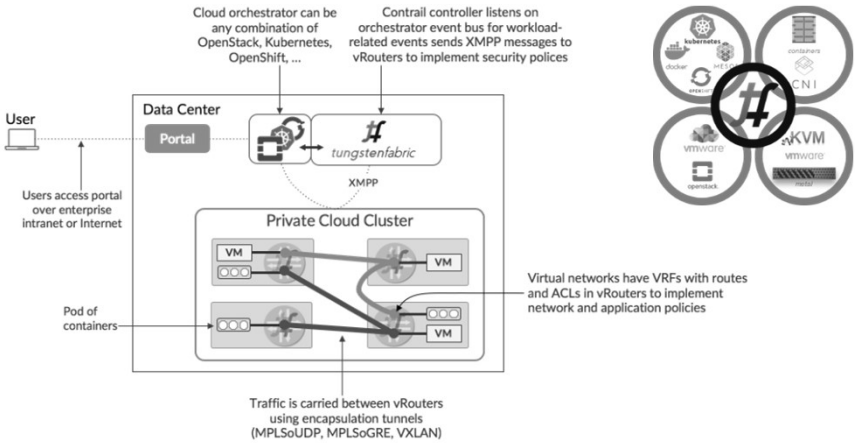


https://www.opendaylight.org/

2. 업스트림(Upstream) 프로젝트 통합

❖ **NFVI Software: Virtual Network: Tungsten Fabric**

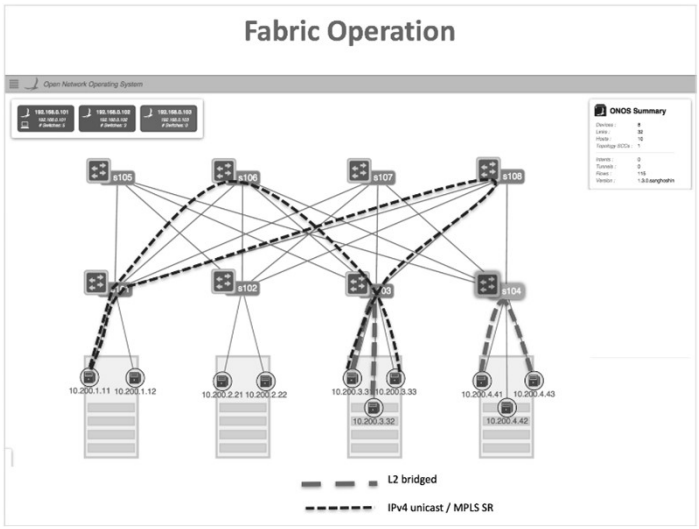
- **vRouter**
- **Tungsten Fabric: Multicloud Multistack SDN**



https://tungsten.io/

2. 업스트림(Upstream) 프로젝트 통합

❖ **SDN Controller: ONOS**



james@jslab.kr

https://kubernetes.io/

2. 업스트림(Upstream) 프로젝트 통합

❖ **Management and Orchestration (MANO)**

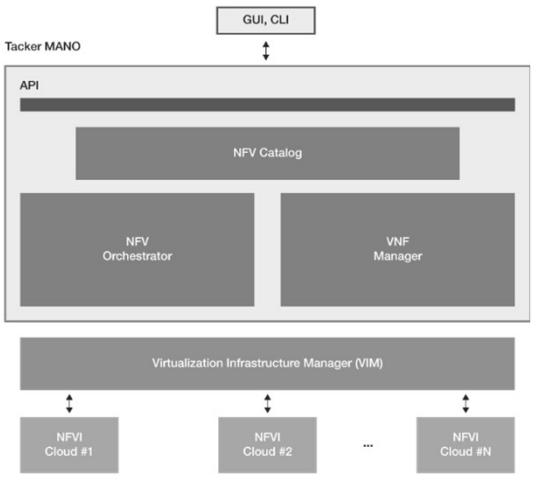
- **NFVO:** Network service (NS - a network service is built by chaining one or more VNFs together) lifecycle management, NS onboarding, NS performance management, NS fault management, VNF forwarding graph (service chaining) management
- **VNFM:** VNF lifecycle management, VNF onboarding, VNF performance management, VNF fault management, VNF image management
- **Design:** Network service design, closed-loop automation application design, policy design, analytic application onboarding, and so on
- **Service Assurance:** Data (events, alarms, logs, metrics) collection, data movement/storage, analytics applications, policy enforcement, alarm correlation

james@jslab.kr

https://kubernetes.io/

2. 업스트림(Upstream) 프로젝트 통합

❖ **MANO: OpenStack Tacker** (멀티클라우드 프로젝트)

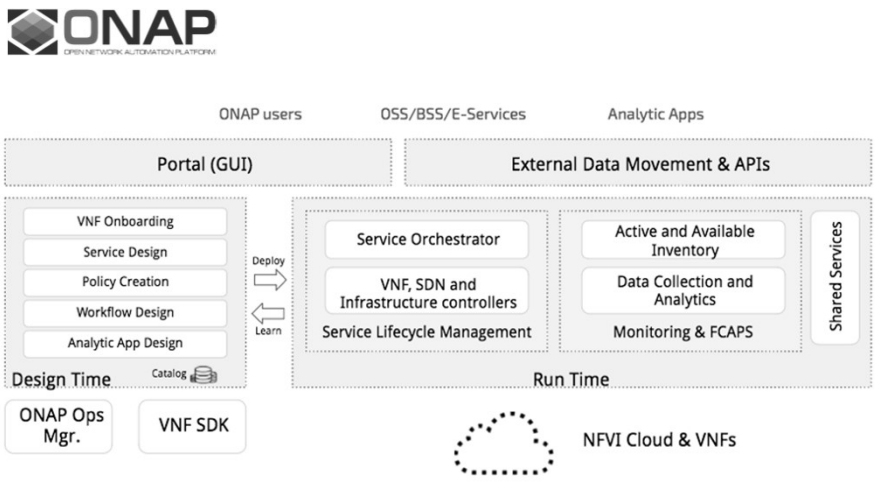


james@jslab.kr

https://docs.openstack.org/tacker/latest/

2. 업스트림(Upstream) 프로젝트 통합

❖ **MANO: ONAP** (**O**pen **N**etwork **A**utomation **P**latform)

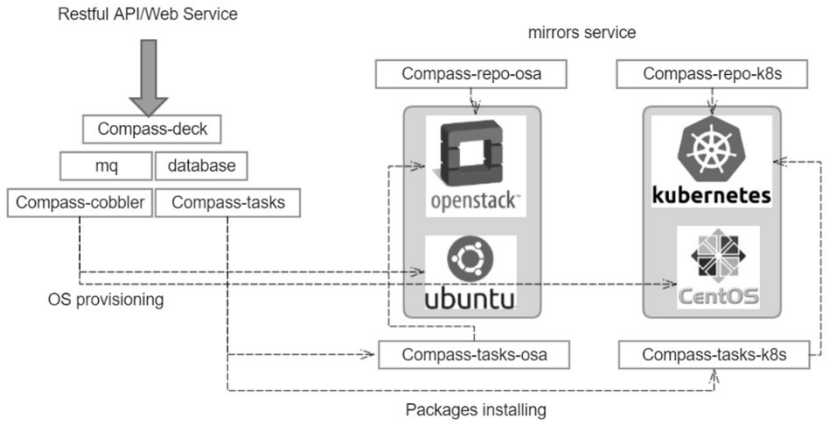


james@jslab.kr

https://www.onap.org/

2. 업스트림(Upstream) 프로젝트 통합

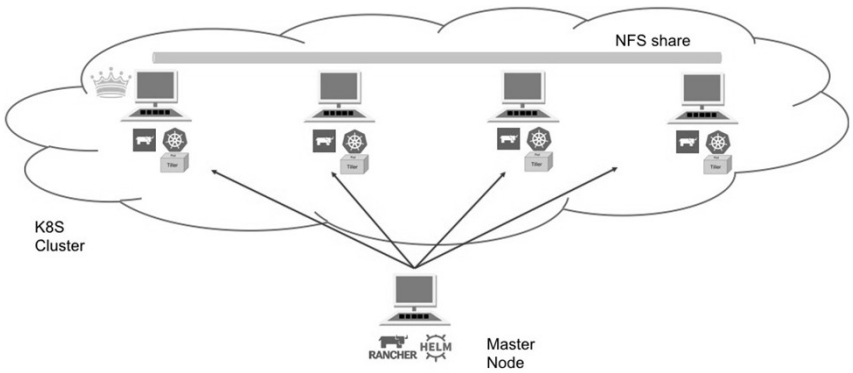
- ❖ Compass was containerized from Euphrates release
- ❖ It will be easy to deploy on any JumpServer. Containerized Compass uses five compass containers instead of a single VM.



https://wiki.opnfv.org/pages/viewpage.action?pageId=10295489

2. 업스트림(Upstream) 프로젝트 통합

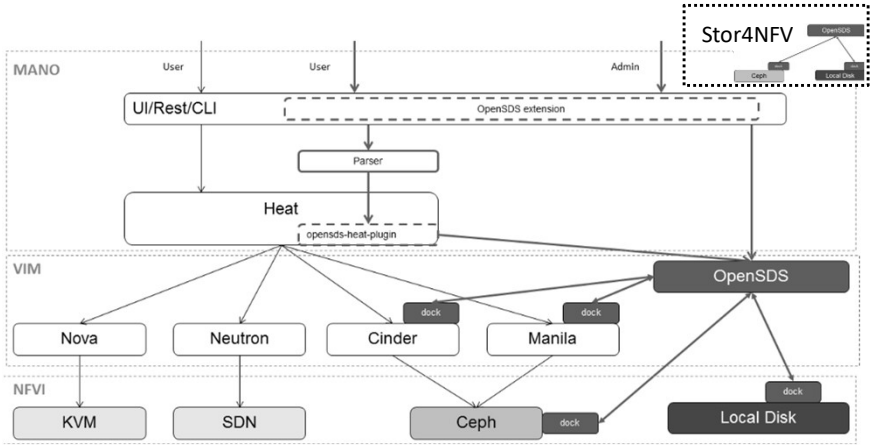
- ❖ ONAP on Kubernetes with Rancher



https://docs.onap.org/en/casablanca/submodules/oom.git/docs/oom_setup_kubernetes_rancher.html

2. 업스트림(Upstream) 프로젝트 통합

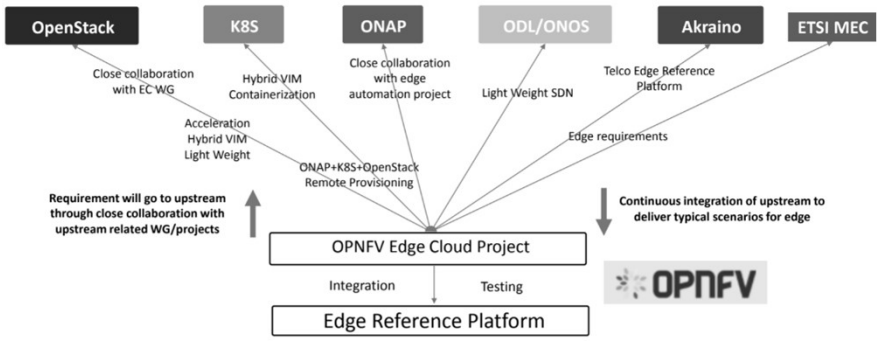
❖ **Stor4NFV can be independently used in any NFV cases without OpenStack or K8s.**



https://wiki.opnfv.org/display/STOR/Stor4NFV+Architecture

2. 업스트림(Upstream) 프로젝트 통합

❖ **Upstream Collaboration for Edge Cloud**



1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

3. 기능(Feature) 프로젝트

❖ OPNFV feature projects can roughly be categorized into three:

- FCAPS (fault, configuration, accounting, performance, security)
- Integration of upstream projects
- Deployment and lifecycle management.

https://docs.openstack.org/tacker/latest/

3. 기능(Feature) 프로젝트

❖ FCAPS (fault, configuration, accounting, performance, security)

Project	Details
Barometer	There is no shortage of open source data collection monitoring tools. Barometer takes an NFV-centric approach to this problem by capturing statistics and events from the NFVI layer in order to detect faults and enforce service level agreements (SLAs). Barometer passes this data on to higher-level fault management systems. The project uses **collectd** for this purpose, and has an extensive set of plugins. These plugins range all the way from IPMI, BIOS, OVS, and DPDK to platform monitoring. Information gathered can be reported to higher-level tools using a variety of interfaces, including standard telco interfaces, such as Simple Network Management Protocol (SNMP).
Calipso	The Calipso project provides real-time discovery, monitoring and visualization of virtual networks. Virtual networks can become large and complex. Moreover, the visibility into these networks is limited, and the relationships between physical and virtual network elements are hard to track. For these reasons, network failures in virtual environments are difficult to troubleshoot. The Calipso project discovers, analyzes, monitors and visualizes virtual networks and their relationships to physical networks, thus simplifying trouble shooting, root-cause analysis and corrective actions.
Doctor	The goal of OPNFV Doctor is that once an infrastructure failure is detected, it should be mapped to the affected virtual resources, and take failover or any other corrective action in a sub-1 second timeframe. The challenges are numerous: • Defining what an infrastructure failure is • Mapping it to affected virtual resources • Marking the status of affected resources appropriately • Generating alarms to a higher-level fault management system. Doctor is an OPNFV success story as it drove requirements into multiple upstream OpenStack projects, including Nova, Neutron, Cinder, and Congress.
High Availability (HA)	This project creates APIs and requirements for high availability (HA) in carrier-grade NFV scenarios. The project addresses HA at three different layers: hardware HA, virtual infrastructure HA, and service HA. As an example of the project's success, during the Colorado release of OPNFV, the HA project identified a number of gaps that were successfully closed in OpenStack's Mitaka release, while some others were deferred to the Newton release.
Moon	The Moon project works with the upstream OpenDaylight (ODL) and OpenStack Keystone and Congress projects to improve the monitoring of authentication, authorization and implementation of Policy Enforcement Points (PEPs). The project does so by identifying gaps in OpenStack and ODL and contributing features to upstream projects around authorization, logging, network enforcement, storage enforcement, and so on.

3. 기능(Feature) 프로젝트

❖ Integration Feature Projects

Project	Details
Bamboo	PNDA.io is the scalable, open source big data analytics platform for networks and services. This project integrates the PNDA.io into OPNFV to enable analytics, closed loop automation, and service assurance.
Container4NFV	The project integrates Kubernetes into OPNFV and has introduced basic performance tests.
FastDataStacks	The project integrates FD.io into OPNFV.
NetReady	Given that NFV is first and foremost a network-centric workload, the NetReady project performs the important role of understanding the gaps in current OpenStack networking models and APIs, as they pertain to carrier-grade needs. Requirements such as L3 only, WAN connectivity and support for legacy networks for specific NFV use cases mean that the current OpenStack networking architecture needs to be evolved. In addition to OpenStack Neutron, NetReady works with the new OpenStack Gluon (Model-Driven, Extensible Framework for NFV Networking Service) project.
KVM for NFV	The KVM for NFV project focuses on the real-time KVM hypervisor in the NFVI, and develops requirements and collaborates with the upstream Linux community to achieve this integration. By using real-time KVM, the community has shown a 10x improvement in small packet performance.
Orchestra	Orchestra integrates the OpenBaton MANO project into OPNFV.
ONOSFW	ONOSFW project integrates the ONOS SDN controller into OPNFV.
OpenContrail for OPNFV	This project integrates the OpenContrail SDN controller into OPNFV.
OVN4NFV	OVN4NFV integrates OVN into OPNFV.
SDNVPN	This project develops requirements for OpenStack BGP VPN. SDNVPN enables the integration of Layer 3 networking services with wide area networks (WAN).
SFC	The SFC project develops requirements, documentation and infrastructure to integrate the upstream ODL SFC implementation project in OPNFV by using OpenStack Tacker.

3. 기능(Feature) 프로젝트

❖ Deployment and Lifecycle Management Projects

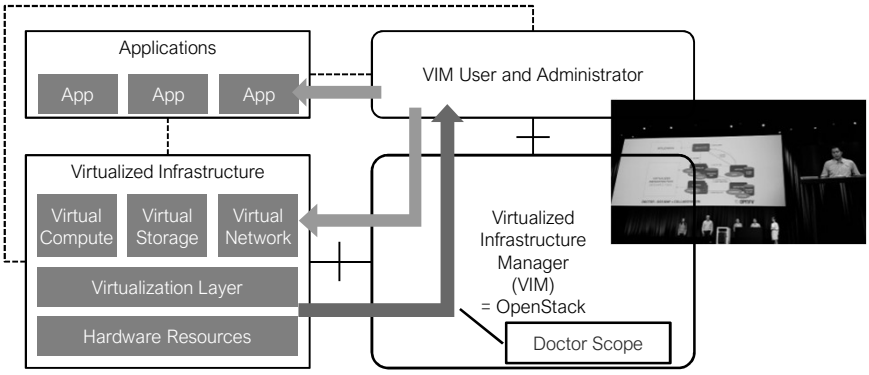
Project	Details
Domino	Centralized orchestration of VNFs and the underlying infrastructure may not always be possible. For example, carrier networks may span geographies, or operators may undergo mergers and acquisitions, resulting in heterogeneous orchestration tools. These situations require a top-down layer that can take a template describing service models and policies, and partition that template into specific templates for each local orchestration and controller tools. This is the role of the Domino project. Domino converts policies to TOSCA and distributes the respective templates using a pub/sub system while taking dependencies into account. The scope of the project includes defining functionality, APIs, test/integration and debugging/tracing.
Parser	The VNFM layer and VNF descriptors provide NFVI requirements (such as vCPU, memory, storage, dataplane acceleration, and so on) and specify post-deployment records (such as utilization, performance report, and so on) in formats such as TOSCA. But these templates often need to be translated to another format such as a Heat template for the VIM layer to act upon them. This project solves the translation problem through four translator projects:**tosca2heat**, **yang2tosca**, **policy2tosca**, **tosca2kube**.
Promise	There are situations when the MANO layer might want to reserve compute, storage, and network resources for a certain duration of time. Before this project, OpenStack had no such mechanism. The OPNFV Promise project solved this gap by helping create the Blazar (reservation-as-a-service) project in OpenStack.

james@jslab.kr

3. 기능(Feature) 프로젝트

❖ Doctor Project

- **Feature project to build fault management and maintenance framework**



james@jslab.kr

3. 기능(Feature) 프로젝트

❖ **Features for Edges**

- **Different from core, edge cloud, especially edge cloud located in access and county levels, are highly distributed.**
- **Space and power are quite limited in edge cloud, leading to constraints on devices of edge cloud**

james@jslab.kr

Light weight control

- Taking consideration of limited space and power, there could be a dozen to less than a hundred nodes in one edge, where it is unnecessary to deploy Orchestrator and VNFM
- VIM and SDN should be deployed in a light weight manner, especially in CO with very hash environment(say less than 10 nodes a stack)

Remote provisioning

- Edge cloud is rather distributed, while the operation environment is considerably hash. Remote provisioning is necessary
- Only hardware maintenance in Access and County level edge, with virtual resource operator sitting at the city level edge for overall virtual resource maintenance
- A unified Orchestrator together with OSS/BSS, EMS, and VNFM are used in regional level to support overall service orchestration and maintenance
- Multi region OpenStack could be considered as one of the solution

Resource Heterogeneity

- With various applications running on edge, heterogeneous resources, including VM, containers, and bare-metal could co-exist in edge
- Edge cloud should be capable of managing the heterogeneous resource pool
- MANO workflow should take these different resources into consideration

Hardware/software acceleration

- Low latency, high bandwidth, large computing service requires various acceleration technology, including DPDK, SR-IOV, GPU, Smart NIC, FPGA and etc.
- OpenStack should fully expose these acceleration capabilities
- Unified API is necessary to fully decouple the VNF services with acceleration resources

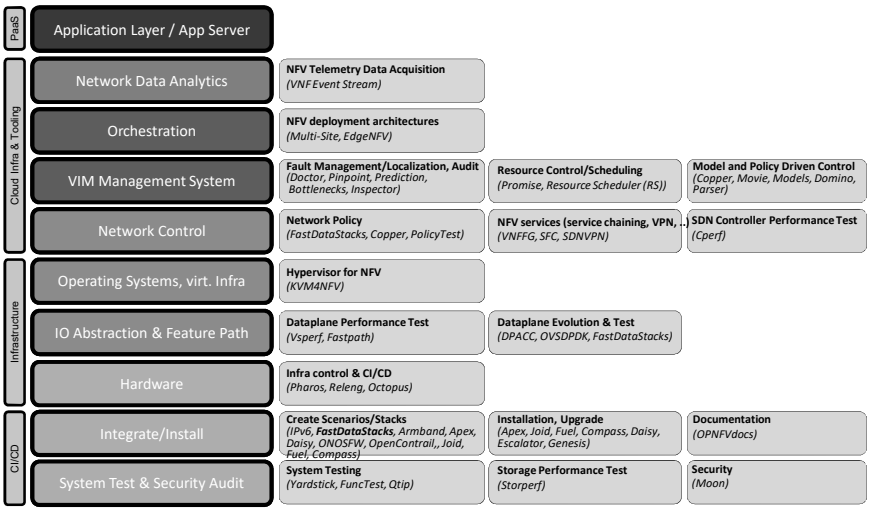
OPNFV

JS Lab

3. 기능(Feature) 프로젝트

❖ **요약**

james@jslab.kr



JS Lab

1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

4. 통합(Integration) 프로젝트

- One of the core pillars of OPNFV is the integration of NFV-related upstream open source projects to create reference architectures/integrated stacks.
- OPNFV integrates hardware and software from the following categories: NFVI, VIM, SDN controllers, and MANO.
- OPNFV allows for multiple options in each category, enabling users to make choices that fit their needs.

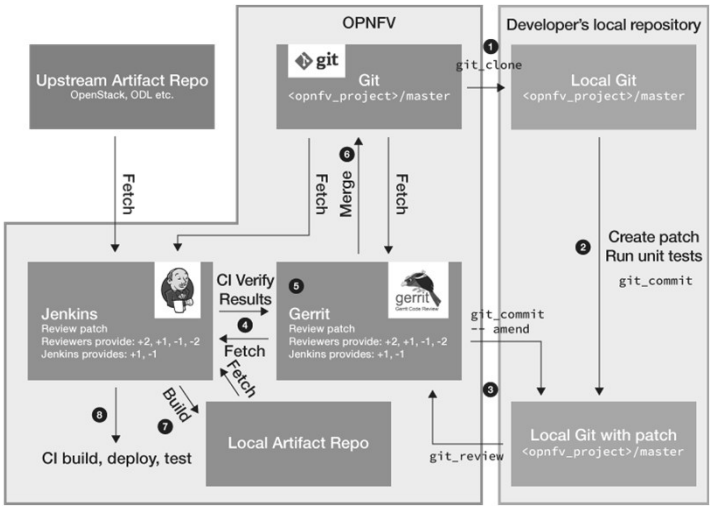
4. 통합(Integration) 프로젝트

❖ **RelEng Project:**

- **Collaboration:** JIRA/Confluence https://jira.opnfv.org/secure/Dashboard.jspa
- **Source code management and code review**:
  - **Git:** https://git.opnfv.org/
  - **Gerrit**: https://gerrit.opnfv.org/gerrit/#/q/status:open
  - **GitHub**
- **CI/software automation:** Jenkins - https://build.opnfv.org/ci/
- **Artifact repository:**
  - **Docker Hub:** http://hub.dicker.com/
  - **Google Cloud Storage:** http://artifacts.opnfv.org/
- **OPNFV Wiki:** https://wiki.opnfv.org/
- **Gerrit:** Code Reviews: Gerrit https://gerrit.opnfv.org/ gerrit.opnfv.org

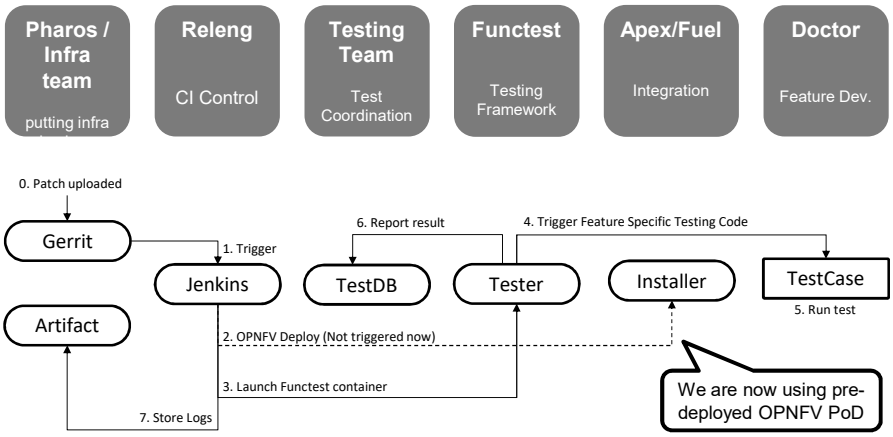
4. 통합(Integration) 프로젝트

❖ **OPNFV CI Pipeline:**



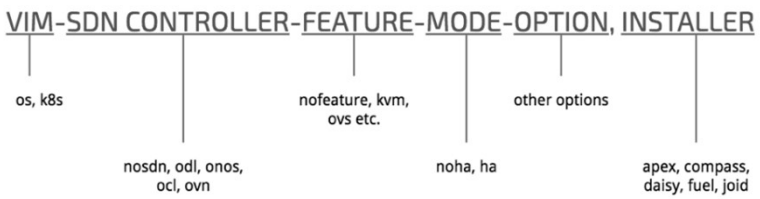
4. 통합(Integration) 프로젝트

❖ **Doctor CI Pipeline:**



4. 통합(Integration) 프로젝트

❖ **Decoding OPNFV Scenarios**



4. 통합(Integration) 프로젝트

❖ OPNFV Installers

Installer	Features
Apex	Uses OpenStack Triple-O to deploy OpenStack-on-OpenStack.
Compass	It is a purpose-built OpenStack installer by Huawei that uses GUI/API/CLI with wizards to assist in the installation process.
Daisy	It uses containerized OpenStack through the Kolla project to ease day-2 lifecycle management tasks.
Fuel	It is based on Mirantis Cloud Platform (MCP) that uses SaltStack, MaaS and Reclass.
JOID	It is based on Canonical MaaS and Juju.

james@jslab.kr

4. 통합(Integration) 프로젝트

❖ OPNFV Installers: Related Projects

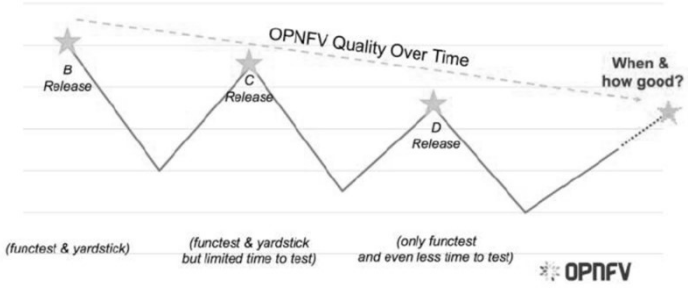
Project	Details
IPv6	Deploys IPv6; also, identifies and fills gaps in upstream projects wrt. IPv6.
ARMBand	Runs OPNFV and the entire CI pipeline on ARM.
FastDataStacks (FDS)	Deploys FD.io scenarios.

james@jslab.kr

4. 통합(Integration) 프로젝트

❖ **XCI(Cross Community) for OPNFV Problem Statement**

- Late integration: 기능 완성에 수개월 이상 소요
- Slow feedback: 기능 동작 확인에 수개월 필요
- Lack of visibility: 현재 상태를 알기에 쉽지 않음
- Too fragmented: 같은 것을 많은 방법으로 시도



james@jslab.kr

https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html#how-to-use

4. 통합(Integration) 프로젝트

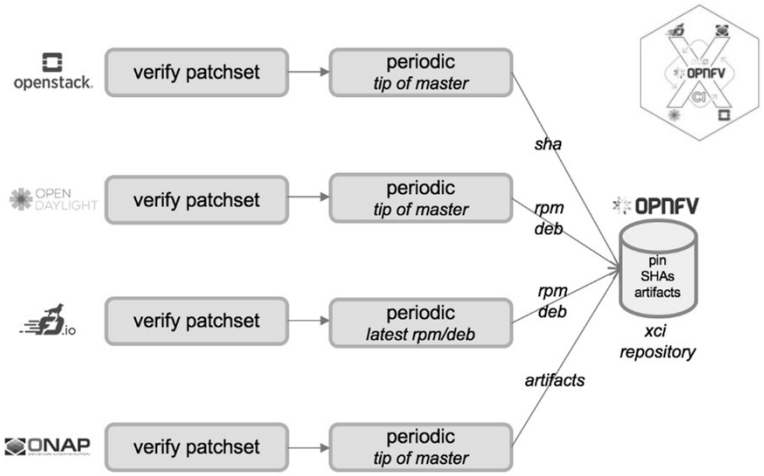
❖ **XCI(Cross Community) Sandbox**

james@jslab.kr

https://opnfv-releng-xci.readthedocs.io/en/latest/xci-overview.html#focus-areas

4. 통합(Integration) 프로젝트

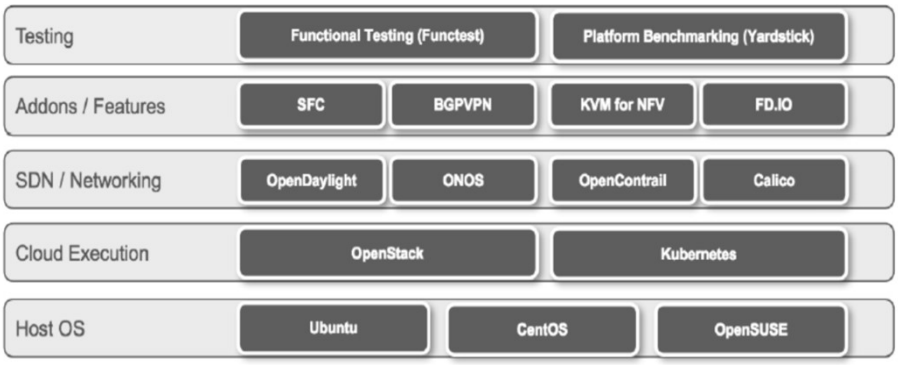
❖ **XCI(Cross Community) repository**



https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html#how-to-use

4. 통합(Integration) 프로젝트

❖ **XCI(Cross Community) 구성** (소프트웨어)



https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html#how-to-use

4. 통합(Integration) 프로젝트

❖ **XCI(Cross Community) 구성** (하드웨어)

Flavor	Number of VM Nodes	VM Specs Per Node	Time Estimates Openstack	Time Estimates Kubernetes
Mini	• 3 VM Nodes • 1 deployment node • 1 controller node • 1 compute node	• vCPUs: 6 • RAM: 12GB • Disk: 80GB • NICs: 1	• Provisioning: 12 mins • Deployment: 65 mins • Total: 77 mins	• Provisioning: 12 mins • Deployment: 35 mins • Total: 47 mins
No HA	• 4 VM Nodes • 1 deployment node • 1 controller node • 2 compute nodes	• vCPUs: 6 • RAM: 12GB • Disk: 80GB • NICs: 1	• Provisioning: 12 mins • Deployment: 70 mins • Total: 82 mins	• Provisioning: 12 mins • Deployment: 35 mins • Total: 47 mins
HA	• 6 VM Nodes • 1 deployment node • 3 controller nodes • 2 compute nodes	• vCPUs: 6 • RAM: 12GB • Disk: 80GB • NICs: 1	• Provisioning: 15 mins • Deployment: 105 mins • Total: 120 mins	• Provisioning: 15 mins • Deployment: 40 mins • Total: 55 mins

https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html

4. 통합(Integration) 프로젝트

❖ **Cross-community Collaboration**

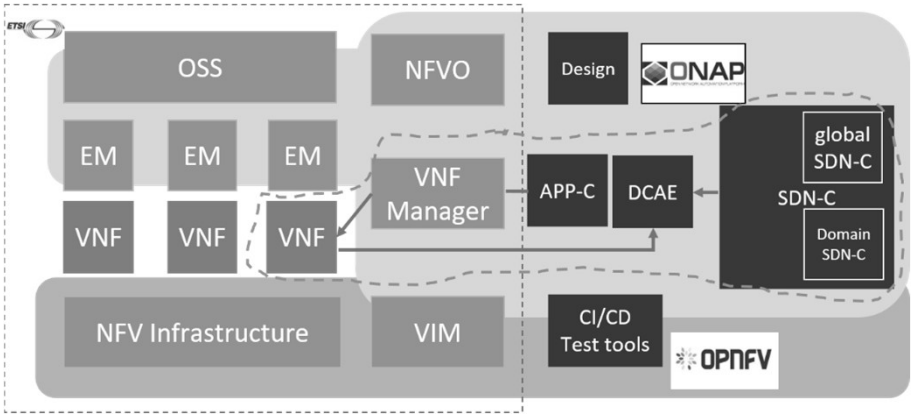
- **C-RAN:** A new project relating to performance measurement for Cloud Radio Access Network (C-RAN) VNF was added in Hunter release.
- **Rocket:** A new project, Rocket aims at improving virtual Evolved Packet Core (vEPC) VNF performance through hardware acceleration.
- **Other Enhancements:** The Hunter release includes container-based undercloud services in the Apex installer and improvements in **IPv6** support and **service function chaining (SFC)**. In the journey towards cloud native, the **Container4NFV** project added support for **Kata Containers and container security for the edge**. The root cause analysis project, Doctor has been working on non-disruptive infrastructure upgrades and faster alarm generation (27x faster than that of Nova reset server state API). The Doctor team works closely with the OpenStack Felix project. Finally, the Barometer project improved user experience with updates to one-click installer and continues to provide valuable NFVI telemetry to projects such as ONAP.

'부록 d. 4G/5G' 참고

https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html#how-to-use

4. 통합(Integration) 프로젝트

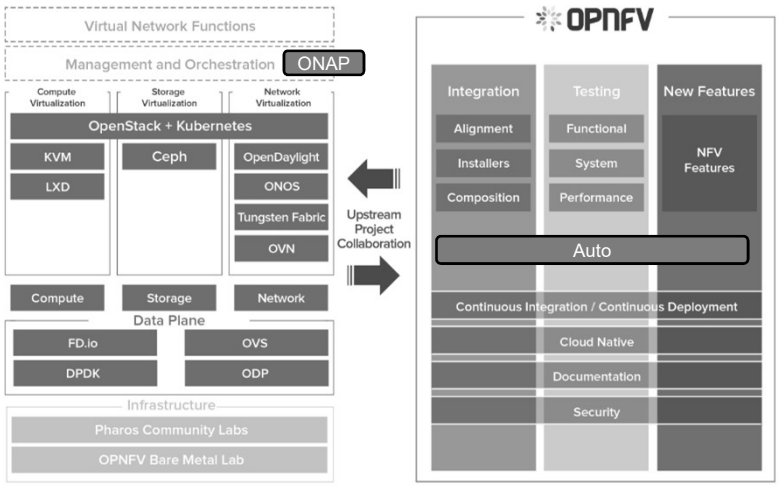
❖ Auto: ONAP-Automated OPNFV
❖ repository: auto



https://wiki.opnfv.org/pages/viewpage.action?pageId=12387216 •최근 변경 : Joe Kidder - 3월 14, 2018

4. 통합(Integration) 프로젝트

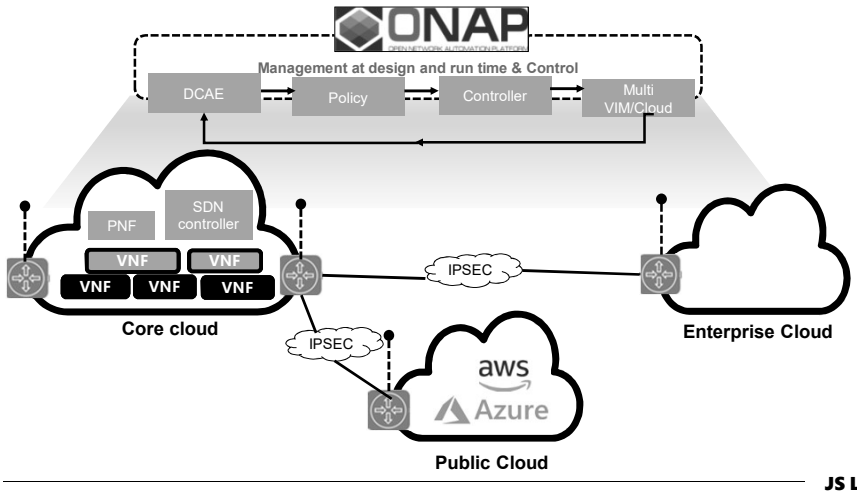
❖ Auto: 아키텍처
❖ OPNFV upstream projects



4. 통합(Integration) 프로젝트

❖ Auto: 확장

❖ DR with ONAP and Multi Cloud



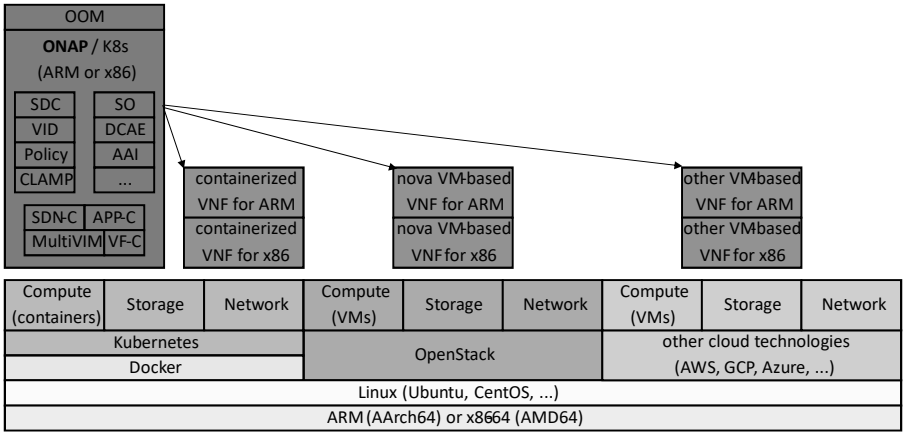
james@jslab.kr

JS Lab

4. 통합(Integration) 프로젝트

❖ Auto: 기능 (feature)

- quantitative test cases
- multiple scenarios

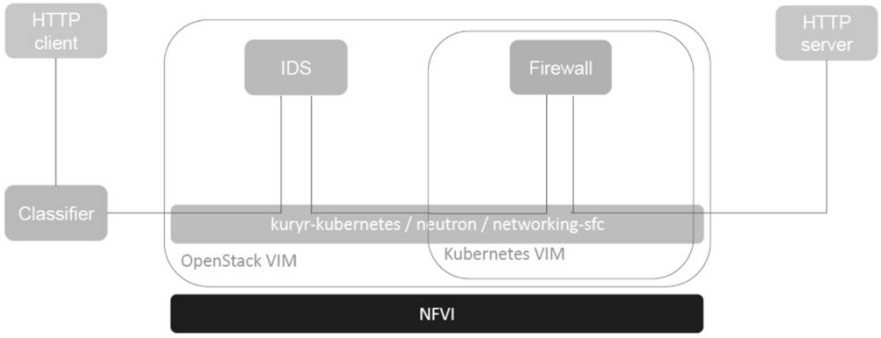


james@jslab.kr

JS Lab

4. 통합(Integration) 프로젝트

❖ **Container based VNFs**

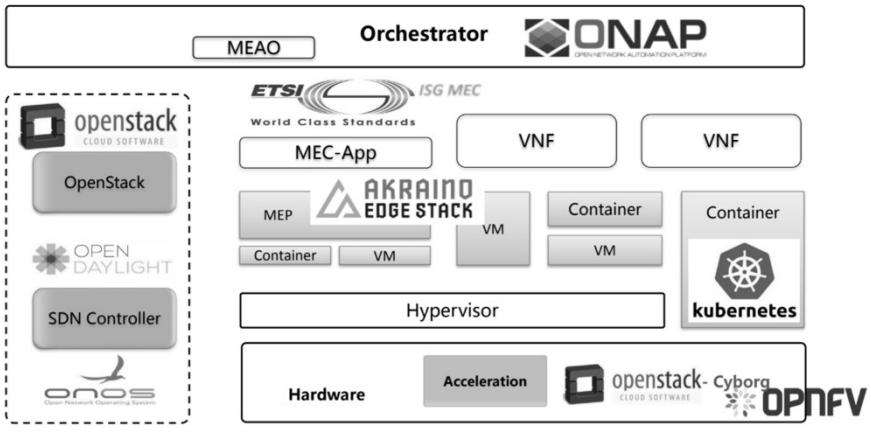


james@jslab.kr

4. 통합(Integration) 프로젝트

❖ **Communities for Edge cloud project**

❖ **OPNFV의 에지 클라우드를 위한 레퍼런스 플랫폼 설계 프로젝트**



james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

4. 통합(Integration) 프로젝트

❖ **Edge cloud project**

- **OPNFV의 에지 클라우드를 위한 레퍼런스 플랫폼 설계 프로젝트**
- **repository 이름: edgecloud**
- **Carrier-scale edge cloud has the following features:**
  - ✓ Limited physical DC environment: limited space and power resource
  - ✓ Remote & scattered location: closer to the users, unattended O&M
  - ✓ Large amount: tens of thousands of edge cloud sites scattered all over the country, while the number of core sites are less than a hundred
  - ✓ Small scale: less than 10 in AP possible
  - ✓ Special traffic-forwarding/processing services require acceleration: low-delay, high-bandwidth services such as MEC, CDN, 5G-UPF
  - ✓ Resource heterogeneity: various infrastructures including VM, container and bare metal

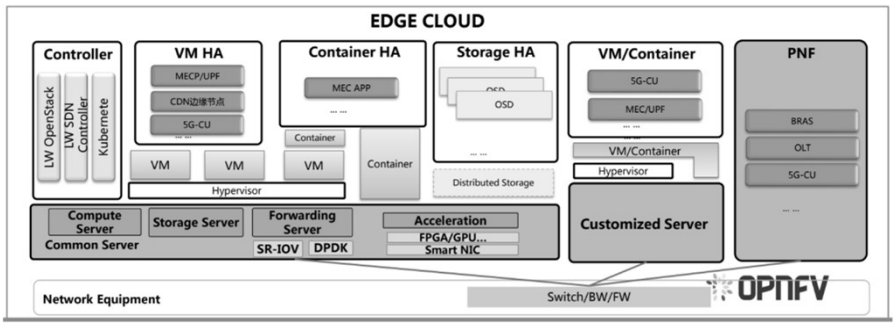
james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

4. 통합(Integration) 프로젝트

❖ **Edge cloud project**

- **Distributed edge clouds bear various applications, resulting in a quite heterogeneous architecture**
- **Edge Cloud Architecture**



james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

4. 통합(Integration) 프로젝트

❖ **Edge cloud: Objectives**

- **Requirements analysis.**
  - ✓ Analyze and conclude the requirements of multiple edge services (MEC, CRAN, vCPE, vOLT, vCDN, etc.)
  - ✓ Translate edge-service requirements into deployment requirements including NFV/SDN component requirements (NFVO, VNFM, VIM, Hypervisor, VSW, HW, etc.)
  - ✓ Guide evolution of Pharos specification suitable for edge cloud scenarios.
- **Scenario design**
  - ✓ Define and release a limited number of scenarios for edge cloud taking consideration of the analyzed requirements
- **Upstream collaboration**
  - ✓ Collaborate with OpenStack, ONAP, Akraino, ODL and other related community for requirements analysis and scenario design
  - ✓ Output detailed requirements for components to relevant project/s.
- **Testing strategy**
  - ✓ Define and develop test strategy and test cases for edge specific testing (e.g. low latency requirement of the stack in edge, performance in specific edge scenarios, and etc.)
  - ✓ Tests developed may also be used by Dovetail if OVP is extended to include edge platforms.

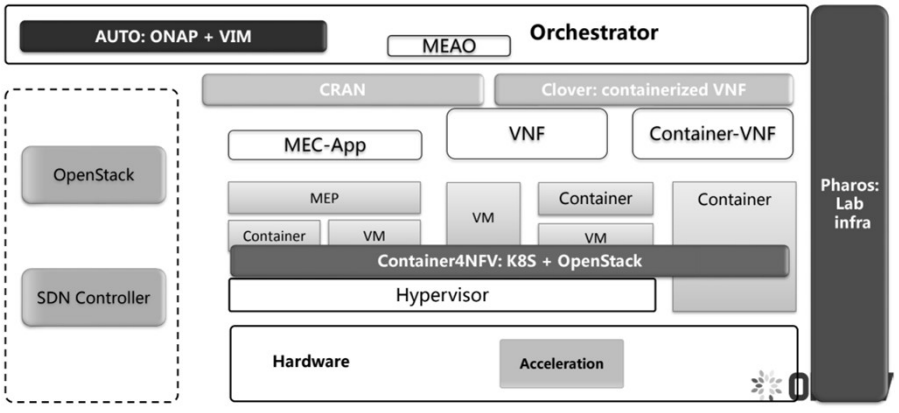
james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

4. 통합(Integration) 프로젝트

❖ **Edge cloud: Related OPNFV projects**

❖ **a new edge-focused project to combine them and cover those uncovered requirements for edge.**

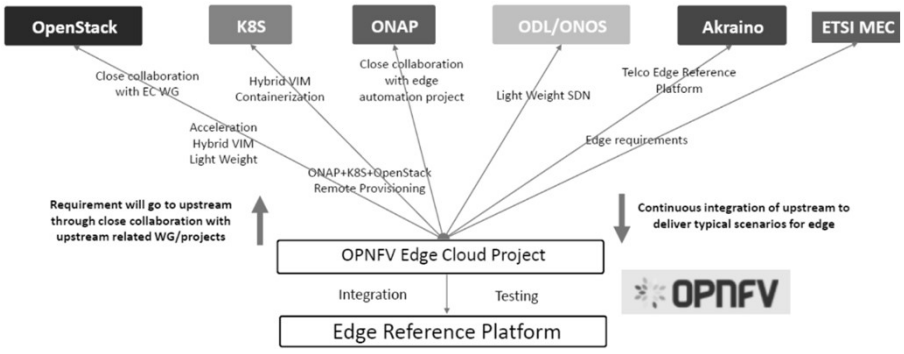


james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

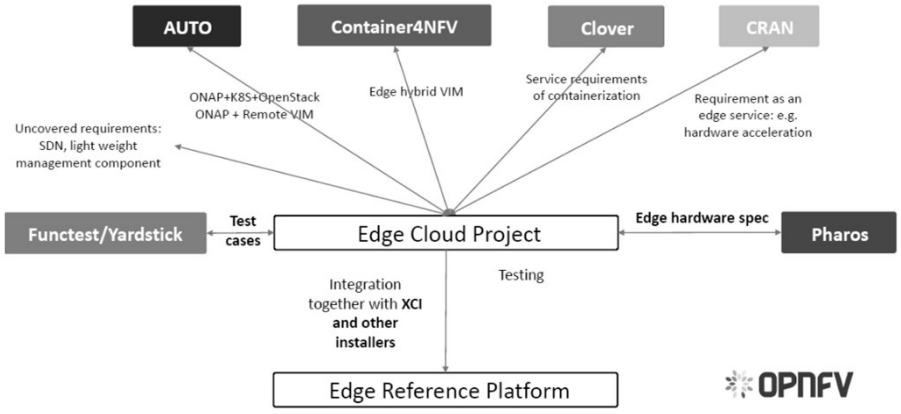
4. 통합(Integration) 프로젝트

❖ Edge cloud: Upstream Collaboration



4. 통합(Integration) 프로젝트

❖ Edge cloud: Projects Cooperation



4. 통합(Integration) 프로젝트

❖ **Edge cloud: Related OPNFV projects**

- **DPACC:** Cooperate with DPACC project in edge cloud use cases. Integrate the acceleration architecture deigned DPACC into edge scenarios. Test the specific scenarios to see if the DPACC acceleration architecture can meet the edge cloud requirement
- **Clover:** Cooperate with Clover project on edge VNF containerization. It can be foreseen future edge will include app deployed as containers. It is necessary to work with clover project to work out the detailed requirements and architecture for containerized app deployed in edge.
- **Container4NFV:** Collaborate with the container4NFV project on edge scenarios. Making sure the edge cloud will coordinate with both K8S and Openstack.
- **Auto:** Collaborate on the edge scenarios. Integrated with ONAP to realize the edge cloud orchestration. Work out the orchestration schema of ONAP with both VM and container in scope

james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

4. 통합(Integration) 프로젝트

❖ **Edge cloud: Dependencies**

- **Related OPNFV projects**
  - ✓ **DPACC**
  - ✓ **Clover**
  - ✓ **Container4NFV**
  - ✓ **Auto**
- **OpenStack:** Cooperate with OpenStack community especially the edge computing WG to verify and explore more on remote management like multi-region, cell and so on. Cooperate with Cyborg project to help the acceleration in edge cloud.
- **ONAP:** Cooperate with ONAP to realize the orchestration of edge scenarios and global orchestration.
- **Akraino:** Cooperate with Akraino with edge cloud reference platform. Output platform design and integrated scenarios to Akraino and reduce vendor-lock in for edge cloud.

james@jslab.kr

https://wiki.opnfv.org/display/PROJ/Edge+cloud

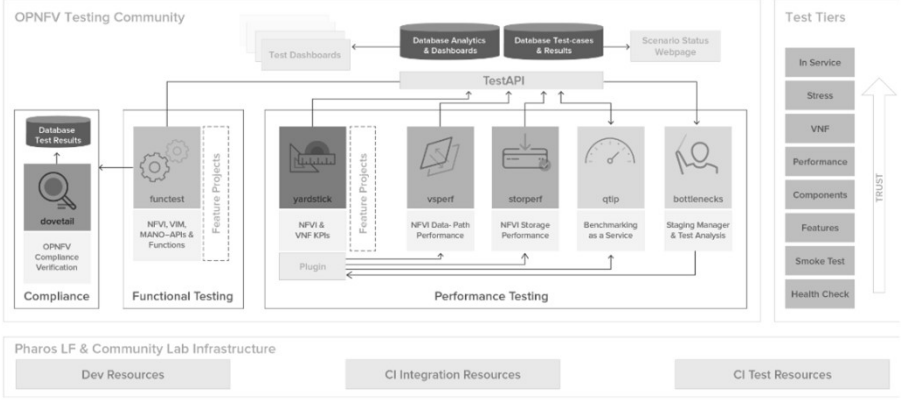
james@jslab.kr

1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Testing Projects:** Functest, Yardstick, Bottlenecks, NFVBench, QTIP, Storperf, VSPERF, Dovetail



JS Lab

5. 테스팅(Testing) 프로젝트

❖ **Testing Project Objectives**

Scope	What Is Tested?
Primary	NFVI/VIM/SDN Controller
Secondary	VNFs

Scope	How Is It Tested?
Primary	• Functional • Performance • Compliance
Secondary	• Stress • Longevity • Scale • Security

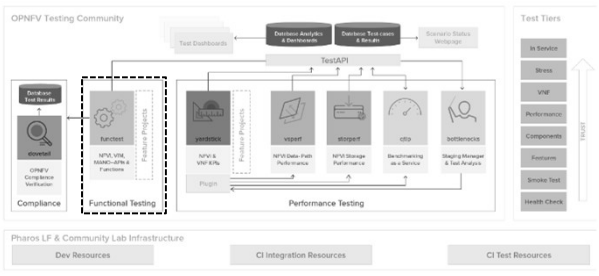
james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Testing Project: Functest**

- **Functest used for functional testing, includes support for OpenStack Rocky and K8s v1.13.5, parallelization of multiple test case execution resulting in faster runs, and the ability to execute Functest on constrained platforms, e.g. Raspberry PI. (@Hunter)**

✓ Functest deals with the functional testing of the VIM and NFVI. It leverages several upstream test suites (OpenStack, ODL, ONOS, etc.) and can be extended by feature projects to add their tests as well. The project is used for scenario validation.



james@jslab.kr

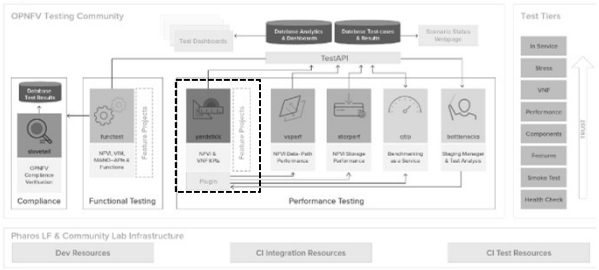
JS Lab

5. 테스팅(Testing) 프로젝트

❖ Testing Project: Yardstick

- **Yardstick, used for performance testing, includes additional support for k8s testing, easy-to-use reports, and expanded support for test tools, e.g. TRex, PktGen, and IxNextGen. Yardstick also includes expanded test cases that improve the testing of L3FWD for OVS-DPDK, SRIOV, vBNG PPPoE, numerous functions of VPP IPSec, and others. (@Hunter)**

✓ The goal of the Yardstick project is to verify infrastructure compliance when running VNF applications with a focus on performance. NFV use cases described in ETSI GS NFV 001 show a large variety of applications, each defining specific requirements and complex configuration on the underlying infrastructure and test tools. The Yardstick concept decomposes typical VNF workload performance metrics into a number of characteristics/performance vectors, and each of them can be represented by distinct test cases.

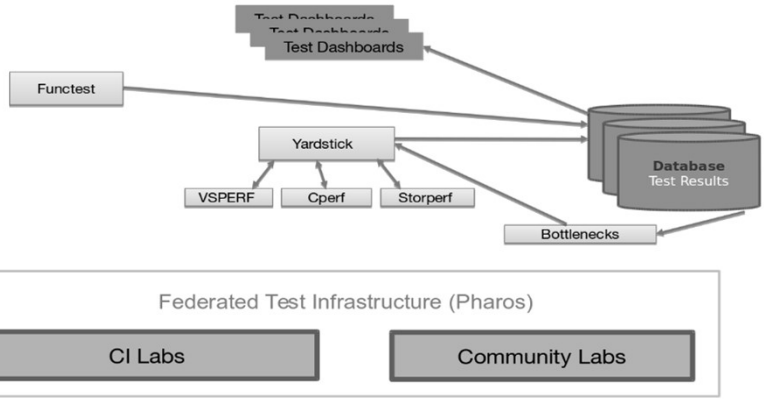


james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: Yardstick

- **Umbrella project for performance testing**
- **Yardstick in Testing Community**



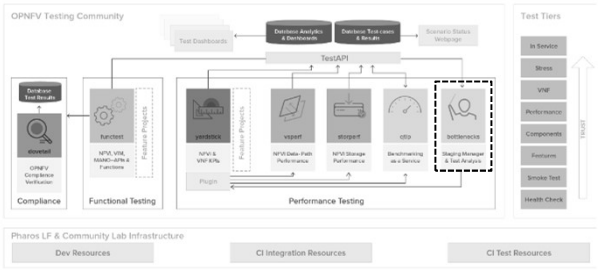
james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: Bottlenecks

- **Bottlenecks, used for stress and longevity testing, has added AI-based historical test results analysis to predict failures in subsequent test runs. Bottlenecks enhancements also include monitoring while testing is in progress. (@Hunter)**

✓ The Bottlenecks project aims to find system bottlenecks by testing and verifying OPNFV infrastructure in a staging environment before committing it to a production environment. Instead of debugging a deployment in a production environment, an automatic method for executing benchmarks is adopted that validates the deployment during staging. This project has more recently been expanded to include stress testing. The stress testing aspect now includes additional tests, such as VNF scale-up/scale-out and lifecycle management event throughput measurements. These tests also include additional monitoring via Prometheus, collectd and Node.js, and visualization through Grafana.

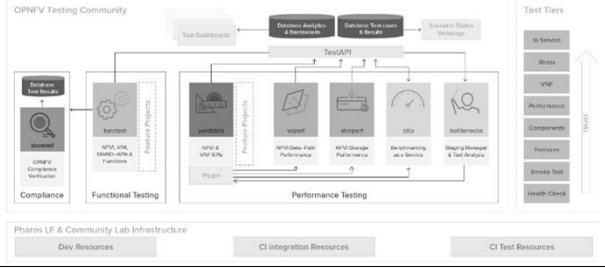


5. 테스팅(Testing) 프로젝트

❖ Testing Project: NFVBench

- **NFVBench, used for NFVI data plane performance testing, includes support for VXLAN-based OpenStack deployments, upgrade to TRex v2.56, along with bug fixes. Starting with this release, NFVBench can run with non-admin credentials as well. (@Hunter)**

✓ The NFVBench project is a compact, self-contained dataplane performance measurement tool for OpenStack-based NFVI platforms. It is agnostic of the NFVI distribution, Neutron networking implementation, and hardware. It runs on any Linux server with a DPDK-compliant NIC connected to the NFVI platform, and bundles a highly efficient software traffic generator. It provides a fully automated measurement of most common packet paths at any level of scale and load using RFC-2544. It is easy to use, as it takes care of most of the guesswork generally associated to dataplane benchmarking, and it can run in any lab or production environments.

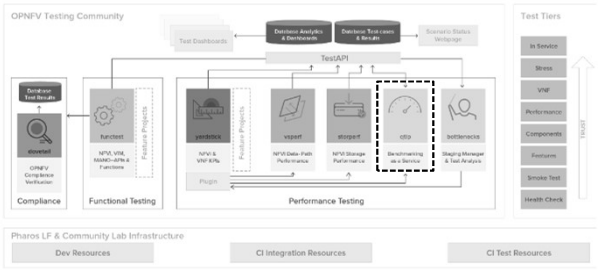


james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: QTIP

- As the project for "Platform Performance Benchmarking" in OPNFV, QTIP aims to provide users a simple indicator for performance, supported by comprehensive testing data and a transparent calculation formula. The current focus is on compute and storage benchmarking of the NFVI layer.



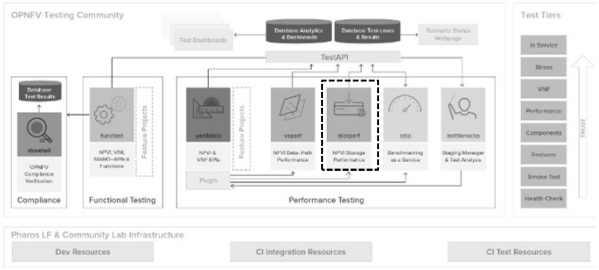
https://wiki.opnfv.org/display/qtip/Platform+Performance+Benchmarking

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: Storperf

- The purpose of the Storperf project is to provide a tool to measure block and object storage performance in an NFVI. When complemented with a characterization of typical VF storage performance requirements, it can provide pass/fail thresholds for testing, staging, and production NFVI environments.



https://wiki.opnfv.org/display/storperf/Storperf

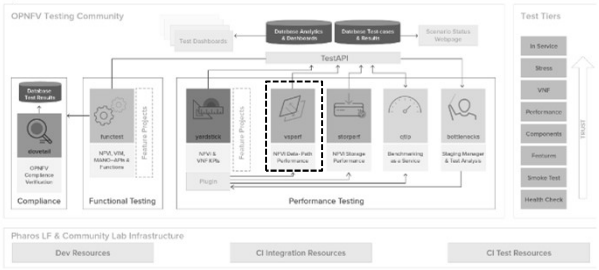
james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: VSPerf

- **VSPerf, used for virtual switch performance characterization, incorporated new test cases and test tools to support causation analysis, and visibility into live metrics during test runs. The release also supports analysis automation and expanded test collection metrics including OVS DPDK core mapping and interrupt latencies and logging during test runs. (@Hunter)**

✓ VSPERF provides an automated test framework and comprehensive test suite based on Industry Test Specifications for measuring NFVI dataplane performance. The datapath includes switching technologies with physical and virtual network interfaces. The VSPERF architecture is switch and traffic generator-agnostic and test cases can be easily customized. Software versions and configurations, including the vSwitch (OVS or FD.io VPP), as well as the network topology, are controlled by VSPERF (independent of the VIM). VSPERF is used as a development tool for optimizing switching technologies, qualification of packet processing components and for pre-deployment evaluation of the NFV platform datapath.

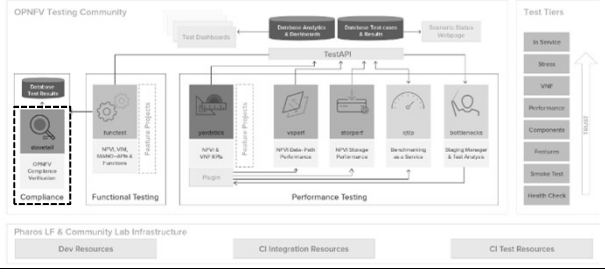


james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ Testing Project: Dovetail

- **As CSPs start using commercial NFVI/VIM offerings in production, it is important for them to ascertain whether these products meet the baseline OPNFV functionality. The Dovetail project intends to define and provide a set of OPNFV-related validation criteria/tests that will provide input for the OPNFV Verification Program. The project intends to incrementally define qualification criteria that establish the foundations of how one is able to measure the ability to utilize the OPNFV platform, how the platform itself should behave, and how applications may be deployed on the platform.**



james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Testing Working Group Resources**

- **Since the different test projects have very similar needs, there is a common set of resources for them to tap into. The common testing working group resources consist of:**
  - ✓ Common APIs
  - ✓ Common database (tests, results, etc.)
  - ✓ Common dashboards.
- **The database is used to store items such as:**
  - ✓ Pods: the list of pods used for production CI
  - ✓ Projects: the list of projects providing test cases
  - ✓ Test cases: the test cases related to a given project
  - ✓ Results: the results of the test cases
  - ✓ Scenarios: the OPNFV scenarios tested in CI..

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Test Results: Results for various OPNFV test projects are available on an easy-to-use dashboard.**

- **Review the OPNFV test results dashboard and answer the following questions:**
  - ✓ For the master branch, did the Storperf test project run against any scenario in the last 10 days? If yes, how many scenarios did it run against? (Hint: Click on each installer link individually to see all scenarios).
  - ✓ For the master branch, did Functest run against the os-odl-sfc-ha scenario in the last 10 days? (Hint: Check the status page).
  - ✓ Find a scenario in which Functest was run against the master branch in the last 10 days. Click on the test and view the console output. Can you identify how many test cases passed, how many failed, and how many were skipped?

james@jslab.kr

http://testresults.opnfv.org/reporting/

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Plugfests**

- **OPNFV conducts plugfests after each release. Vendors and community members get together and collaborate on interoperability testing and different test projects. The plugfest serves two purposes:**
  - ✓ First, it accelerates projects significantly by having a burst of intense activity with the ability to work on issues and solve problems with other contributors face-to-face rather than remotely.
  - ✓ Second, vendors with proprietary products that cannot be included in the OPNFV CI pipeline can test their wares across different scenarios and test cases.
- **These plugfests are held concurrently with hackfests.**

james@jslab.kr

http://testresults.opnfv.org/reporting/

5. 테스팅(Testing) 프로젝트

❖ **Apex-Based Basic Install**

- **3-node OPNFV install using Intel NUC i7 nodes**
- **16GB RAM, 250MB SSD, and 1 TB HDD**
- **The install uses one NUC for the Jumphost, and one each for the controller (OpenStack + ODL) and the compute nodes.**

james@jslab.kr

https://wiki.opnfv.org/display/copper/DevStack+in+a+VM+Notes

5. 테스팅(Testing) 프로젝트

❖ **Hardware Requirements for virtual Deploys:** The following minimum hardware requirements must be met for the virtual installation of Gambia using Fuel

- **1 Jumpserver:** A physical node (also called Foundation Node) that will host a Salt Master container and each of the VM nodes in the virtual deploy
- **CPU:** Minimum 1 socket with Virtualization support
- **RAM:** Minimum 32GB/server (Depending on VNF work load)
- **Disk:** Minimum 100GB (SSD or 15krpm SCSI highly recommended)

james@jslab.kr

https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

JS Lab

5. 테스팅(Testing) 프로젝트

❖ **Hardware Requirements for baremetal Deploys:** The following minimum hardware requirements must be met for the baremetal installation of Gambia using Fuel

- **1 Jumpserver:** A physical node (also called Foundation Node) that hosts the Salt Master container and MaaS VM
- **# of nodes:** Minimum 5
  - ✓ 3 KVM servers which will run all the controller services
  - ✓ 2 Compute nodes
- **CPU:** Minimum 1 socket with Virtualization support
- **RAM:** Minimum 16GB/server (Depending on VNF work load)
- **Disk:** Minimum 256GB 10kRPM spinning disks
- **Networks:** Mininum 4
  - ✓ 3 VLANs (public, mgmt, private) - can be a mix of tagged/native
  - ✓ 1 Un-Tagged VLAN for PXE Boot - PXE/admin Network
- **Power mgmt.:** All targets need to have power management tools that allow rebooting the hardware (e.g. IPMI)

james@jslab.kr

https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

JS Lab

5. 테스팅(Testing) 프로젝트

❖ **Hardware Requirements for hybrid (baremetal + virtual) Deploys:** The following minimum hardware requirements must be met for the hybrid installation of Gambia using Fuel

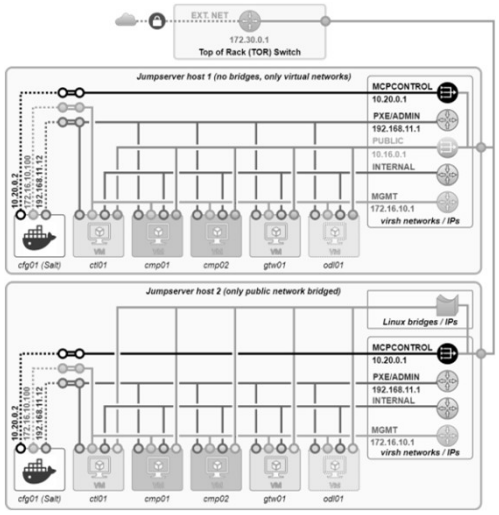
- **1 Jumpserver:** A physical node (also called Foundation Node) that hosts the Salt Master container, MaaS VM and each of the virtual nodes defined in PDF
- **# of nodes:** Minimum 5
  - ✓ If the control plane is virtualized, minimum baremetal requirements are:
    - ➢ 2 Compute nodes
  - ✓ If the computes are virtualized, minimum baremetal requirements are:
    - ➢ 3 KVM servers which will run all the controller services
- **CPU:** Minimum 1 socket with Virtualization support
- **RAM:** Minimum 16GB/server (Depending on VNF work load)
- **Disk:** Minimum 256GB 10kRPM spinning disks
- **Networks:** Same as for baremetal deployments
- **Power mgmt.:** Same as for baremetal deployments

james@jslab.kr

https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Fuel Virtual noHA POD Network Layout Examples**



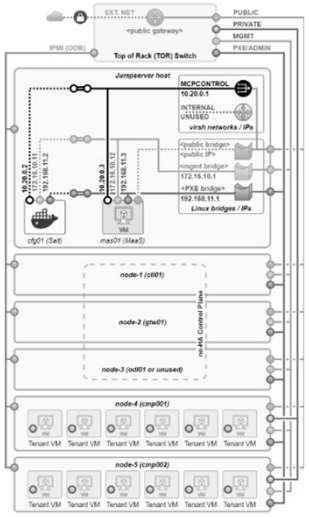
james@jslab.kr

JS Lab

https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

5. 테스팅(Testing) 프로젝트

❖ OPNFV Fuel Baremetal noHA POD Network Layout Example

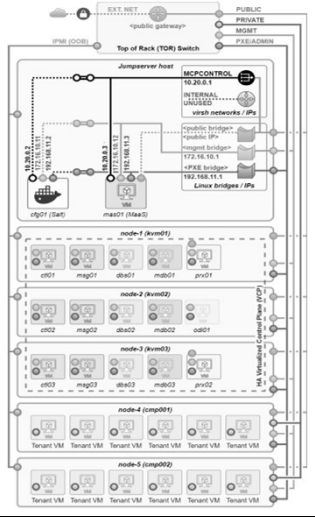


https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ OPNFV Fuel Baremetal HA POD Network Layout Example

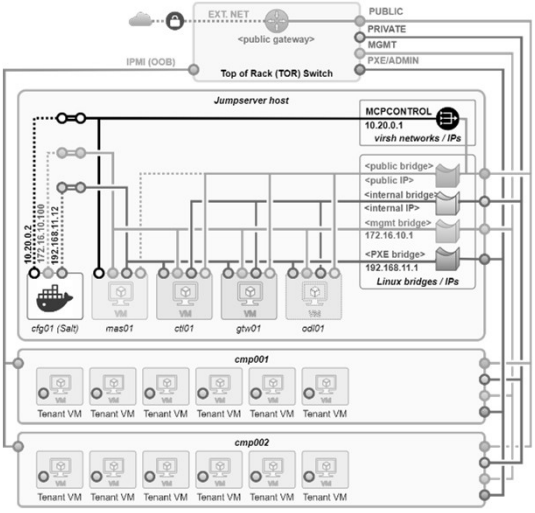


https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Fuel Hybrid noHA POD Network Layout Examples**



https://opnfv-fuel.readthedocs.io/en/stable-gambia/release/installation/installation.instruction.html#opnfv-software-prerequisites

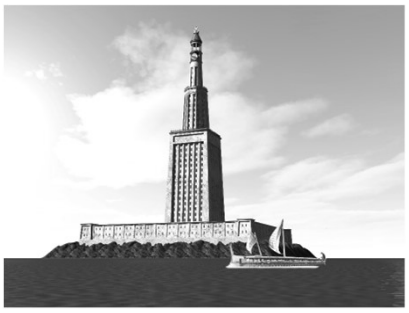
JS Lab

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

- **OPNFV Pharos Lab-as-a-Service**
- **OPNFV XCI Sandbox**
- **Edge Pharos Effort:** With edge computing around the corner, the original Pharos footprint is proving to be a bit too heavy. To address this, the Edge Cloud project is working on a reduced Pharos specification specifically for edge environments.



https://wiki.opnfv.org/display/pharos/Pharos+Specification

JS Lab

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

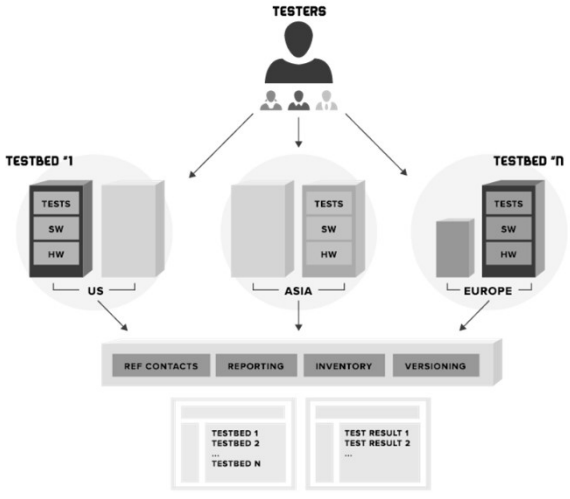
- **Community Labs: Active** (May 2019)

Organization	Location	POD	vPOD	Note
Linux Foundation	Portland, Oregon	lf-pod1, lf-pod2, lf-pod3, lf-pod4, lf-pod5	lf-virtual1, lf-virtual2, lf-virtual3	
China Mobile	Beijing, China	cm-pod1		1 Full Bare Metal Pod
Enea	Kista, Sweden			CI POD operational, second POD for test/development work available upon request
Ericsson	Rosersberg, Sweden	ericsson-pod1, ericsson-pod2	ericsson-virtual1 ericsson-virtual2 ericsson-virtual3ericsson-virtual4 ericsson-virtual5 ericsson-build3, ericsson-build4	
Huawei	Lang Fang, China	huawei-pod7,huawei-pod12	huawei-virtual5	Huawei Hosting
Huawei	Munich, Germany	huawei-pod5,huawei-pod22		Huawei Hosting
Huawei	Shang Hai, China	huawei-pod1,huawei-pod2,huawei-pod3,huawei-pod4	huawei-virtual1,huawei-virtual2,huawei-virtual3,huawei-virtual4	Huawei Hosting
Intel	Portland, Oregon	Intel Hosting	intel-virtual7, intel-virtual8, intel-virtual9	
Orange	Lannion, France			
Orange	Paris, France			
CENGN	Ottawa, Canada	cengn-pod1, cengn-pod2		JOID, FDS, Internship Program
ZTE	Shang hai, China	zte-pod1, zte-pod2, zte-pod3, zte-pod4, zte-pod8	zte-virtual1, zte-virtual2, zte-virtual3, zte-virtual4	Used for CI (performance related)
Nokia	Espoo, Finland	nokia-pod1		Doctor and Functest testing
OOL	Okinawa, Japan		ool-virtual1, ool-virtual2, ool-virtual3	Used for Doctor CI
BII	Beijing, China	BII-pod1		Using the lastet Danbue3.0 BII Hosting
Flex	Milpitas, California	flex-pod1, flex-pod2		Used for Apex, Yarstick and Fuel deployment
ITRI	Hsinchu, Taiwan	itri-pod1		Industrial Technology Research Institute
CAICT	Beijing, China	caict-pod1		Used for Fuel deployment

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**



https://www.opnfv.org/community/projects/pharos

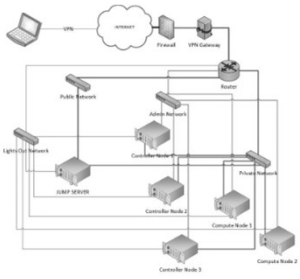
james@jslab.kr

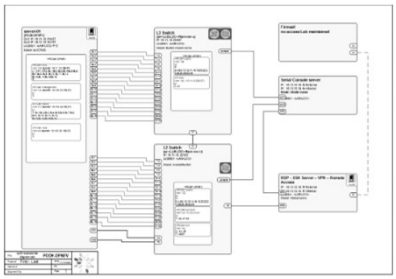
5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

- **One CentOS/Ubuntu jump server on which the virtualized Openstack/OPNFV installer runs. For an ARM POD, the jump server should also be an ARM server**
- **3 controller nodes**
- **2 compute nodes**
- **A configured network topology allowing for LOM, Admin, Public, Private, and Storage Networks**
- **Remote access as defined by the Jenkins slave configuration guide**

(http://artifacts.opnfv.org/octopus/brahmaputra/docs/octopus_docs/opnfv-jenkins-slave-connection.html#jenkins-slaves)



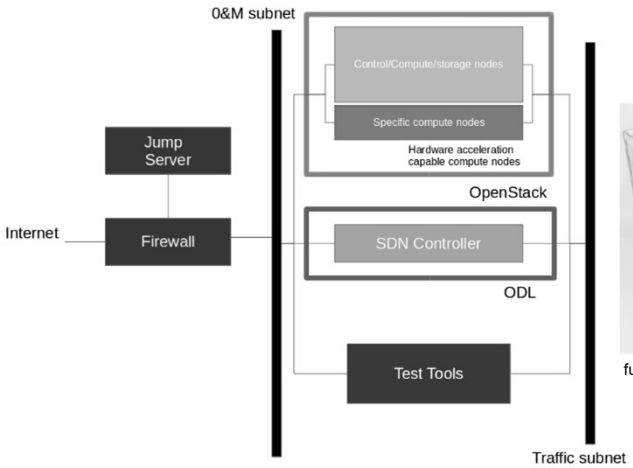


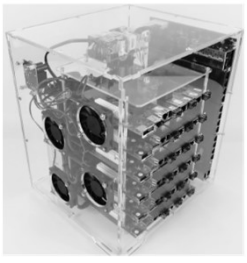
https://wiki.opnfv.org/display/pharos/Pharos+Specification

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**





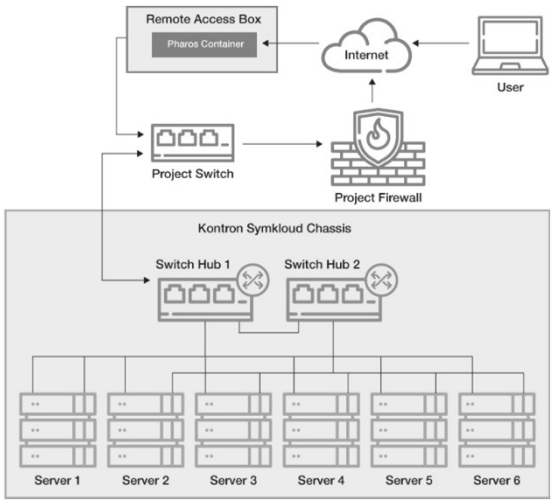
fully compliant with the OPNFV Pharos specification.

https://wiki.opnfv.org/display/pharos/Pharos+Specification

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**



james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

- **Intel PHAROS POD-14**
  - ✓ https://wiki.opnfv.org/display/pharos/Intel+POD14
- **Intel PHAROS POD-20**
  - ✓ https://wiki.opnfv.org/display/pharos/Intel+POD20
- **PHAROS VPN Access:**
  - ✓ https://wiki.opnfv.org/display/INF/Infra+Lab+Support

Intel POD 20 is for Akraino Testing during Q2-Q3 2018

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

- **Intel POD20**
- **Intel POD 20 is for Akraino Testing during Q2-Q3 2018**

Host Name	CPU	Memory	Storage	IPMI	1G NIC	10G NIC	
pod20-jump	2x Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz	64GB	480GB (SSD)	IF0: 10.10.200.10 A4:BF:01:00:03:D1 U/P: root/root N/A	IF0: 10.10.200.20 F16, A4:BF:01:00:03:CF, VLAN 200 (DMZ) IF1: 10.10.200.30-99, A4:BF:01:00:03:D0, VLAN 200 (DMZ north-south data)	IF2: **PVID 201**, 00:1E:67:FD:EE:30, VLAN 201 (Private - MGMT, support 9216 MTU) 192.168.204.0/28 IF3: 00:1E:67:FD:EE:30, VLANs 1000-1099 (east-west data)	
pod20-node1	2xE5-2699	64GB	480GB (SSD)	IF0: 10.10.200.11 A4:BF:01:00:06:15 U/P: root/root http://10.10.200.21	IF0: 10.10.200.21 A4:BF:01:00:06:13, VLAN 200 (DMZ - OAM) IF1: 10.10.200.30-99, A4:BF:01:00:06:14, VLAN 200 (DMZ north-south data)	IF2: **PVID 201**, 00:1E:67:FD:EF:14, VLAN 201 (Private - MGMT, support 9216 MTU) 192.168.204.0/28 IF3: 00:1E:67:FD:EF:15, VLANs 1000-1099 (east-west data	Edge SX Controller-0
pod20-node2	2x Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz	64GB	480GB (SSD)	IF0: 10.10.200.12 A4:BF:01:00:09:7B U/P: root/root http://10.10.200.22 Uses node 3 (controller-1) Node 4 (compute-0) node 5 (compute-1)	IF0: 10.10.200.22 [eno1 X722 1G], A4:BF:01:00:09:79, VLAN 200 (DMZ - OAM) IF1: 10.10.200.100-176 [eno2 X722 1G], A4:BF:01:00:09:7A, VLAN 200 (DMZ north-south data)	IF2 [enp24s0f0 XXV710 25GbE]: **PVID 202**, 00:1E:67:FD:F6:B8, VLAN 202 (Private - MGMT, support 9216 MTU) 192.168.204.16/28 IF3: [enp24s0f1 XXV710 25GbE] 00:1E:67:FD:F6:B9, VLANs 1100-1199 (east-west data)	Edge DX-0 Controller-0
pod20-node3	2x Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz	64GB	480GB (SSD)	IF0: 10.10.200.13 A4:BF:01:00:02:4B U/P: root/root http://10.10.200.	IF0: 10.10.200.23 [eno1 X722 1G], A4:BF:01:00:02:49, VLAN 200 (DMZ) IF1: 10.10.200.100-176 [eno2 X722 1G] , A4:BF:01:00:02:4A, VLAN 200 (DMZ north-south data)	IF2 [enp24s0f0 XXV710 25GbE]: **PVID 202**, 00:1E:67:F8:57:78, VLAN 202 (Private - MGMT, support 9216 MTU) 192.168.204.16/28 IF3 [enp24s0f1 XXV710 25GbE]: 00:1E:67:F8:57:79, VLANs 1100-1199 (east-west data)	Edge DX-0 Controller-1
pod20-node4	2xE5-2699	64GB	480GB (SSD)	IF0: 10.10.200.14 A4:BF:01:00:03:5E U/P: root/root http://10.10.200.	IF0: 10.10.200.24 A4:BF:01:00:03:5C, VLAN 200 (DMZ) IF1: 10.10.200.200.177-254, A4:BF:01:00:03:5D, VLAN 200 (DMZ north-south data)	IF2: **PVID 203**, 00:1E:67:FD:F4:8C, VLAN 203 (Private - MGMT, support 9216 MTU) 192.168.204.32/28 IF3: 00:1E:67:FD:F4:8D, VLANs 1200-1299 (east-west data)	Edge DX-1 Controller-0
pod20-node5	2xE5-2699	64GB	480GB (SSD)	IF0: 10.10.200.15 A4:BF:01:00:0B:65 U/P: root/root http://10.10.200.	IF0: 10.10.200.25 FA4:BF:01:00:0B:63, VLAN 200 (DMZ) IF1: 10.10.200.177-254, A4:BF:01:00:0B:64, VLAN 200 (DMZ north-south data)	IF2: **PVID 203**, 00:1E:67:FE:01:C0, VLAN 203 (Private - MGMT, support 9216 MTU)192.168.204.32/28 IF3: 00:1E:67:FE:01:C1, VLANs 1200-1299 (east-west data	Edge DX-1 Controller-1

JS Lab

https://wiki.opnfv.org/display/pharos/Intel+POD20

5. 테스팅(Testing) 프로젝트

❖ **Pharos Project:**

- **Intel POD 20 is for Akraino Testing during Q2-Q3 2018**

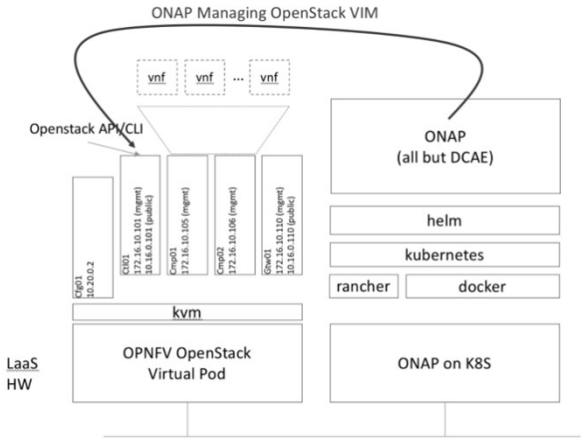
JS Lab

https://wiki.opnfv.org/display/pharos/Intel+POD20

5. 테스팅(Testing) 프로젝트

❖ **Auto: 기능 (feature)**

- **LaaS** (Lab as a Service): The Auto project is using LaaS to work on integration and test of the ONAP/OPNFV integration.



https://wiki.opnfv.org/display/AUTO/Auto+Lab+Deployment

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Verified Program (OVP) currently covers the NFVI/VIM layers**

- **The ONAP community sees value in OPNFV test frameworks such as Dovetail, Functest, and Yardstick.**
- **OPNFV CI Pipeline for ONAP**
- **Cloud Native Collaboration**
- **Edge Computing Collaboration**

Results and lessons from the fourth ONAP Developer Design Forum (DDF) and sixth OPNFV Plugfest, January 8-11, 2019.

5. 테스팅(Testing) 프로젝트

❖ **ONAP DDF ACTIVITIES**

- **Production Deployment**
- **Real World Experience**
- **Edge Computing**
- **Requirements, Architecture, Testing and Planning**
  - ✓ SDO/Open Source Harmonization
  - ✓ Orchestration and Lifecycle Management
  - ✓ Control Loop Automation & Network Data Analytics
  - ✓ ONAP Deployment
  - ✓ Security
  - ✓ NFVI/VIM (NFV Cloud)
- **Blueprints**
  - ✓ 5G
  - ✓ CCVPN
  - ✓ vCPE
  - ✓ Change Management
- **Modeling**
- **Cloud Native**

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Plugfest Lab and Hardware Resources**

- **Nokia, Intel, and Lenovo generously made hardware available to Plugfest attendees**
  - ✓ 1 x AirFrame Open Edge hardware chassis
  - ✓ 5 x AirFrame Open Edge server blades with single socket Intel Xeon SP with 20 cores 256 GB memory, 2 x 25Gb NICs
  - ✓ 1 x AirFrame OR18 switch Z9100ON, 32x 100GbE
- **NOKIA also made a POD available remotely from Espoo, Finland** (https://wiki.opnfv.org/display/pharos/Nokia+Hosting):
  - ✓ 6 x AirFrame OR18 Nodes (3 control, 2 compute, and one jump host)
  - ✓ 1 x AirFrame OR18 switch Z9100ON, 32x 100GbE
  - ✓ 1 x AirFrame OR18 switch S3048ON, 48x 1GbE
  - ✓ 1 x AirFrame OR18 PSU for power shelf
  - ✓ 1 x OR Rack

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Plugfest Lab and Hardware Resources**

- **Intel POD 14 was made available remotely from Hillsboro, Oregon** (https://wiki.opnfv.org/display/pharos/Intel+POD14)**:**
  - ✓ • 5 Servers: 2xE5-2699 CPU, 64-128GB RAM, SSD storage, 2 x 10 GbE, 2 x 1 GbE
  - ✓ • AG07-EXT-1 1GbE switch
  - ✓ • AG07-IZ1-1 10GbE switch
  - ✓ • StarlingX 2018-11 software
- **Lenovo made a POD available remotely from Morrisville, North Carolina** (https://wiki.opnfv.org/display/EVNT/Lenovo+Fraser+Plugfest+Page)**:**
  - ✓ 6 x AirFrame OR18 Nodes (3 control, 2 compute, and one jump host)
  - ✓ 1 x AirFrame OR18 switch Z9100ON, 32x 100GbE
  - ✓ 1 x AirFrame OR18 switch S3048ON, 48x 1GbE
  - ✓ 1 x AirFrame OR18 PSU for power shelf
  - ✓ 1 x OR Rack

james@jslab.kr

5. 테스팅(Testing) 프로젝트

❖ **OPNFV Plugfest Lab and Hardware Resources**

- **Intel POD 14 was made available remotely from Hillsboro, Oregon** (https://wiki.opnfv.org/display/pharos/Intel+POD14)**:**
  - ✓ • 5 Servers: 2xE5-2699 CPU, 64-128GB RAM, SSD storage, 2 x 10 GbE, 2 x 1 GbE
  - ✓ • AG07-EXT-1 1GbE switch
  - ✓ • AG07-IZ1-1 10GbE switch
  - ✓ • StarlingX 2018-11 software
- **Lenovo made a POD available remotely from Morrisville, North Carolina** (https://wiki.opnfv.org/display/EVNT/Lenovo+Fraser+Plugfest+Page)**:**
  - ✓ 6 x AirFrame OR18 Nodes (3 control, 2 compute, and one jump host)
  - ✓ 1 x AirFrame OR18 switch Z9100ON, 32x 100GbE
  - ✓ 1 x AirFrame OR18 switch S3048ON, 48x 1GbE
  - ✓ 1 x AirFrame OR18 PSU for power shelf
  - ✓ 1 x OR Rack

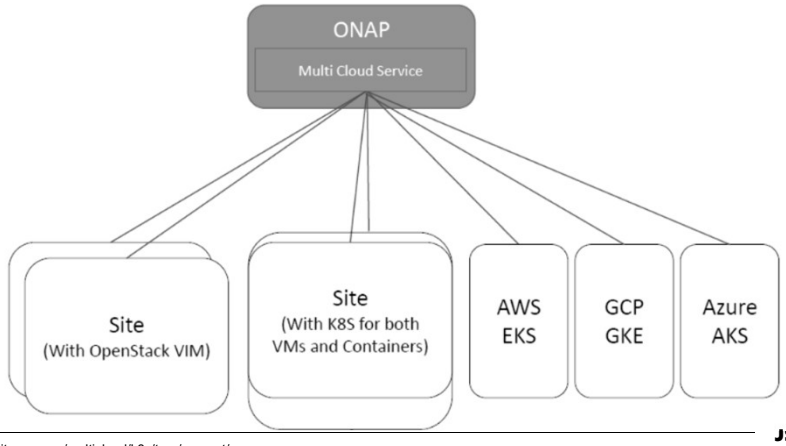
james@jslab.kr

JS Lab
Results and lessons from the fourth ONAP Developer Design Forum (DDF) and sixth OPNFV Plugfest, January 8-11, 2019.

5. 테스팅(Testing) 프로젝트

❖ **ONAP support for K8s based Cloud regions**

- **Kubernetes based Cloud-region support in ONAP to bring up VM and container based VNFs/workloads**
- **Kubernetes Reference Deployment (KRD)**



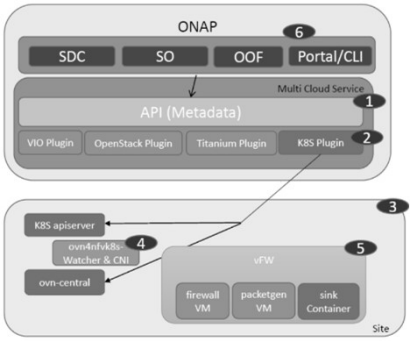
james@jslab.kr

https://git.onap.org/multicloud/k8s/tree/vagrant/

5. 테스팅(Testing) 프로젝트

❖ **Work items categories**

- **Uniform API across cloud technologies**
- **K8s Multi-Cloud Service plugin**
- **Kubernetes Reference Deployment**
- **OVN4NFVK8S**
- **Hybrid vFW ONAP use case**
- **Integration with SO, OOF and SDC**



james@jslab.kr

https://git.onap.org/multicloud/k8s/tree/vagrant/

5. 테스팅(Testing) 프로젝트

- Abstraction of networking tech/features e.g. through NetReady/Gluon
- Blueprint-based VNF deployment (HOT, TOSCA, YANG)
- Application level configuration and lifecycle thru YANG (for any aspects depending upon OPNFV NFVI+VIM components)
- Policy (through DCAE)
- Telemetry (through VES/DCAE)

5. 테스팅(Testing) 프로젝트

- Testing Documentation

Functional Testing

- Functest: Kubernetes: Healthcheck
- Functest: Kubernetes: Smoke
- Functest: Kubernetes: Features
- Functest User Guide
- Functest: Docker: Arm
- Functest Docker: Components
- Functest Docker: Features
- Functest Docker: Healthcheck
- Functest Docker: Smoke
- Functest: Docker: Parser
- Functest Docker: VNF

Infrastructure Compliance

- Yardstick User Guide
- Yardstick Docker
- Yardstick Docker aarch64

System Limitation Testing

- Bottlenecks Testing Guide
- Bottlenecks Docker

Storage Performance Testing

- StorPerf User Guide
- Storperf Docker

Network Performance Testing

- VSPERF Configuration and User Guide
- VSPERF Test Guide

1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

6. 사례(Use Cases)

❖ Functional use cases

- Virtual Central Office - enterprise
- Virtual Central Office - residential.

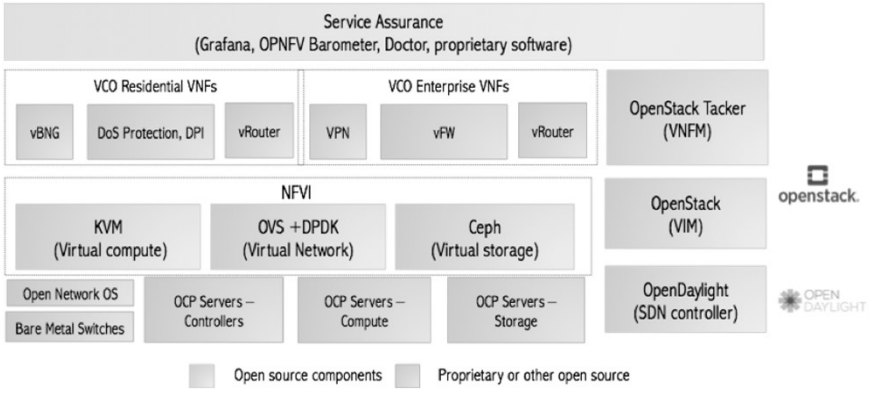
❖ Operational use cases

- VNF Onboarding
- Commercial NFVI/VIM validation
- In-house NFV CI pipeline
- In-house VNF certification platform.

https://opnfv-releng-xci.readthedocs.io/en/latest/xci-user-guide.html#how-to-use

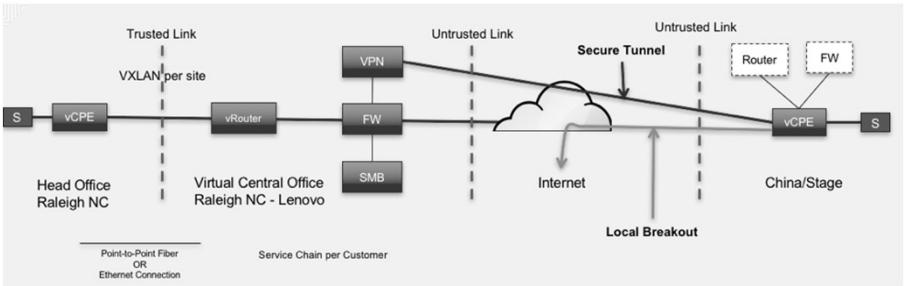
6. 사례(Use Cases)

❖ Virtual Central Office
❖ The Virtual Central Office stack is presented below:



6. 사례(Use Cases)

❖ Virtual Central Office: Enterprise
❖ OPNFV VCO Enterprise Network Service



6. 사례(Use Cases)

- Virtual Central Office: Residential
- OPNFV VCO Residential Network Service

james@jslab.kr

6. 사례(Use Cases)

- Commercial NFVI/VIM Validation
- Given that OPNFV represents a "tuned" vendor-agnostic NFVI/VIM stack, it is important for users to determine whether or not a commercial product (open source or proprietary) meets the baseline functionality available across various OPNFV scenarios. For this reason, the community has created an OPNFV Verified Program (OVP) that vendors can use to verify their products against in terms of defined interfaces, behaviors and key features, while still letting the vendors add value. There is a formal process, and test results have to be submitted to OPNFV to gain an OPNFV Verified badge.
- For more information, review the OVP website.

james@jslab.kr

https://www.opnfv.org/verified

6. 사례(Use Cases)

❖ In-house NFVI CI Pipeline

- With different components for the NFV stack coming from different open source projects and proprietary vendors, the timing of updates and upgrades is impossible to control. For this reason, CSPs need an NFV CI pipeline to continuously integrate and test the various components that make up their stack — very similar to how the OPNFV community approaches this problem. For this reason, users can adapt an OPNFV CI pipeline for in-house use.

james@jslab.kr

https://www.opnfv.org/verified

6. 사례(Use Cases)

❖ In-house VNF Validation Platform

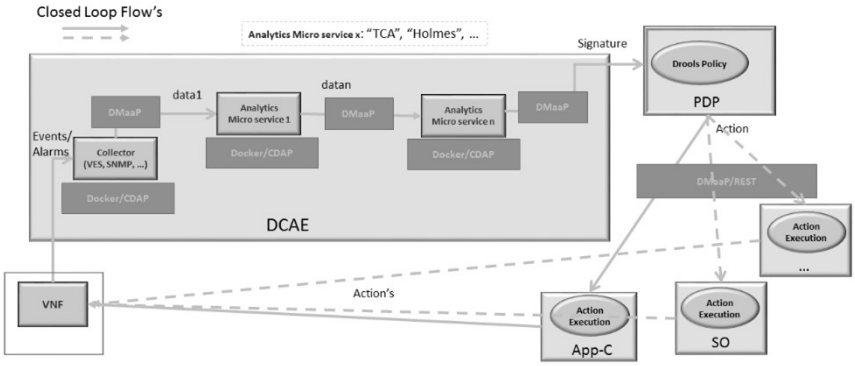
- VNFs are characterized by vendors in their own environment that is often not representative of the operator's environment. Given that OPNFV represents a "tuned" vendor-agnostic software stack to test and characterize VNFs, OPNFV, and test projects such as Yardstick and NFVbench offer a way to onboard, characterize and validate VNFs in a consistent vendor-agnostic manner.

james@jslab.kr

https://www.opnfv.org/verified

6. 사례(Use Cases)

❖ **Auto Edge Cloud Use Case**



https://wiki.opnfv.org/display/AUTO/Auto+Use+Cases#AutoUseCases-UseCase1

6. 사례(Use Cases)

❖ **ONAP support for K8s based Cloud regions**

- **Uniform API across cloud technologies** (HEAT, K8s, Azure etc..)
- **K8s Multi-Cloud Service plugin** (https://github.com/onap/multicloud-k8s/tree/master/src/k8splugin)
  - ✓ Support for deployment and services(K8S Helm charts).
  - ✓ Networking – OVN, flannel and Multus (Create/Delete VNs, Distributed Router, Gateways, SNAT in Gateway)
- **Kubernetes Reference Deployment** (https://github.com/onap/multicloud-k8s/tree/master/vagrant)
  - ✓ Installation of software & configuration to make K8s based sites.
  - ✓ Additional of Virtlet, Multus, OVN, SRIOV and flannel.
- **OVN4NFVK8S** (https://github.com/opnfv/ovn4nfv-k8s-plugin)
  - ✓ Support for multiple virtual networks
  - ✓ Support for dynamic creation/deletion of virtual networks
- **Hybrid vFW ONAP use case** (https://github.com/onap/multicloudk8s/blob/master/vagrant/tests/integration_vcFW.sh)
  - ✓ Firewall and packet generator as Virtual Machines and Sink services containerized.
- **Integration with SO, OOF and SDC**

https://git.onap.org/multicloud/k8s/tree/vagrant/

james@jslab.kr

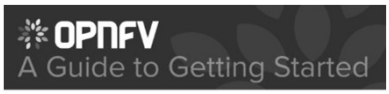
1. 개요
2. 업스트림(Upstream) 프로젝트 통합
3. 기능(Feature) 프로젝트
4. 통합(Integration) 프로젝트
5. 테스팅(Testing) 프로젝트
6. 사례(Use Cases)
7. 참여(Getting Involved)

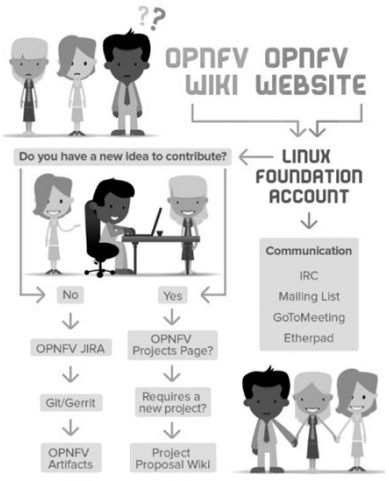
james@jslab.kr

7. 참여(Getting Involved)

❖ Get Involved

- End Users
- Developers
- Membership





https://www.opnfv.org/community/get-involved

7. 참여(Getting Involved)

- Code and documentation are available through git git.starlingx.io
- Apache 2 license
- IRC: #starlingx@Freenode
- Mailing List for daily discussions http://lists.starlingx.io/cgi-bin/mailman/listinfo/starlingx-discuss
- Weekly meetings: Zoom calls https://wiki.openstack.org/wiki/Starlingx/Meetings

7. 참여(Getting Involved)

- Bugs are tracked in Launchpad https://bugs.launchpad.net/starlingx
- New ideas are introduced in the specs repository https://git.openstack.org/cgit/openstack/stx-specs/
- Design and implementation work is tracked in StoryBoard https://storyboard.openstack.org/#!/project_group/86

7. 참여(Getting Involved)

- We cordially invite you to join the StarlingX community
- Please try out the code and read the documents on https://www.starlingx.io/
- Please sign up for the mailing list http://lists.starlingx.io/cgi-bin/mailman/listinfo
- Please attend community meetings https://wiki.openstack.org/wiki/StarlingX#Meetings
- Please consider joining as a member

james@jslab.kr

james@jslab.kr

JS Lab