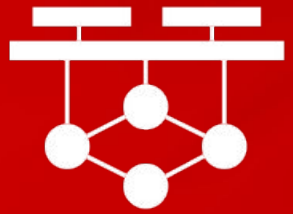


ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

OPENDAYLIGHT TUTORIAL

September 2018

Janki Chhatbar
jchhatba@redhat.com
Senior Software Engineer

Rimma Iontel
riontel@redhat.com
Senior Architect, Red Hat

For Hands-On Instructions

https://tinyurl.com/ons2018odl

More details:
https://docs.openstack.org/tripleo-quickstart/latest/getting-started.html

OPENDAYLIGHT PRIMER

History of OpenDaylight

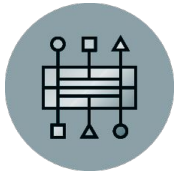
- Open source project hosted by Linux Foundation
- SDN project developed in answer to the industry need for network programmability
  - Result of a collaboration started in 2013
  - Founding members: Arista Networks, Big Switch Networks, Brocade, Cisco, Citrix, Ericsson, HP, IBM, Juniper Networks, Microsoft, NEC, Nuage Networks, PLUMgrid, Red Hat and VMware
- Modular open platform for customizing and automating networks
- Current release: Fluorine (August 2018)
- Base/component for several commercial SDN and virtualized application projects
- Included in open source frameworks for
  - Cloud: OpenStack
  - Orchestration: ONAP
  - NFV: OPNFV

OpenDaylight Use Cases

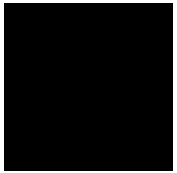
- SDN Integration
  - Aggregate control of different SDNs
  - Centralized administration
  - Global network view
- SDN-as-a-Code Approach
  - Bring Agility and DevOps to network management
  - Include networking into CI/CD pipeline
- Network Function Virtualization
  - Converged control for VNF networking and PNFs
  - Service Function Chaining
  - Unified Orchestration and Operational approach
    - Service instantiation
    - Lifecycle management
    - Service assurance

Why OpenDaylight?

True SDN Platform



Standard-based, Open Approach



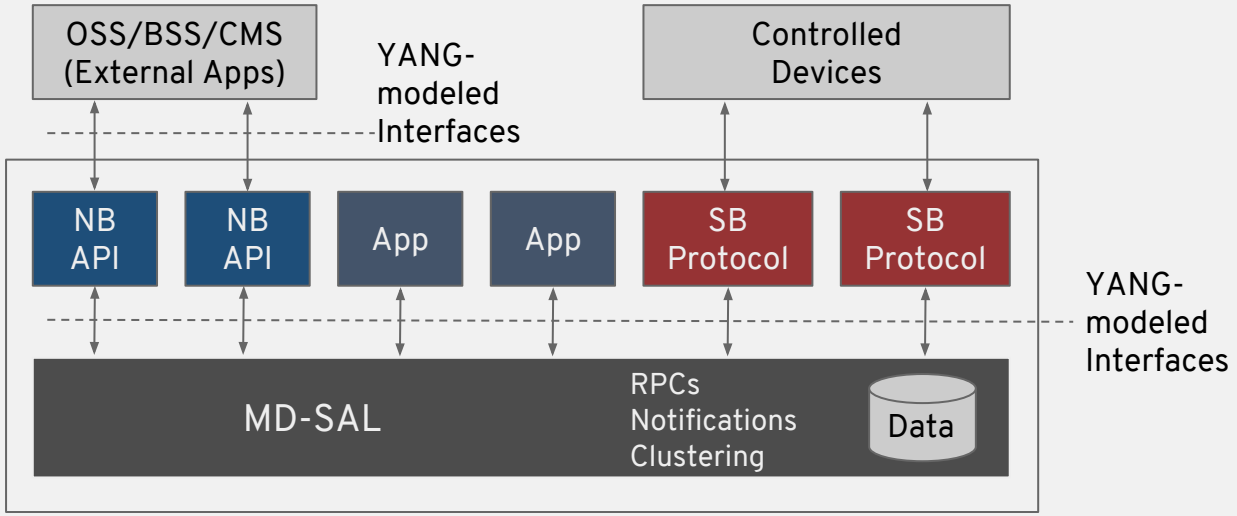
Managing Physical Fabric

Enhanced Cloud Networking

SDN for NFVI

OpenDaylight: a YANG-Based Microservices Platform

- Based on Model-Driven Service Abstraction Layer (MD-SAL)
  - Network devices and network applications as objects
- Creates well-defined APIs
- Java and RESTCONF APIs auto-generated from YANG
- Controller Clustering

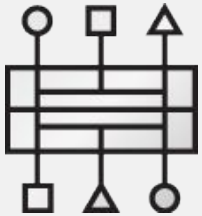


True SDN Platform

- Modular and extensible
  - OSGi framework based
- Multi-protocol
  - OpenFlow, OVSDB, NETCONF, BGP, PCEP, LISP, SNMP
- Large community and ecosystem
  - 3rd party applications utilizing ODL Northbound APIs
  - Commercial SDN controllers
    - Inocybe Open Networking Platform
    - Lumina SDN Controller – Commercial Edition
    - Pantheon ODL Platform
  - Product integration
    - Red Hat OpenStack Platform
- Ready for future innovation

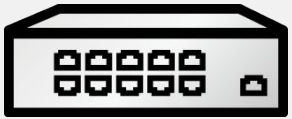
Open Approach Based on Standards

- Open APIs available for 3rd party applications
  - Can coexist with proprietary plugins and services
  - Bidirectional Northbound REST APIs
    - Topology manager, host tracker, flow programmer, static routing…
- Standard interfaces and protocols
- Full “reference stack”
- Avoid vendor lock-in



Managing Physical Fabric

- Multi-vendor infrastructure
- Topology discovery
  - Across physical and virtual domains
- Network automation
- Overlay/underlay correlation
- Service Assurance
  - Monitoring, diagnosis, troubleshooting, and analysis



Enhanced Cloud Networking

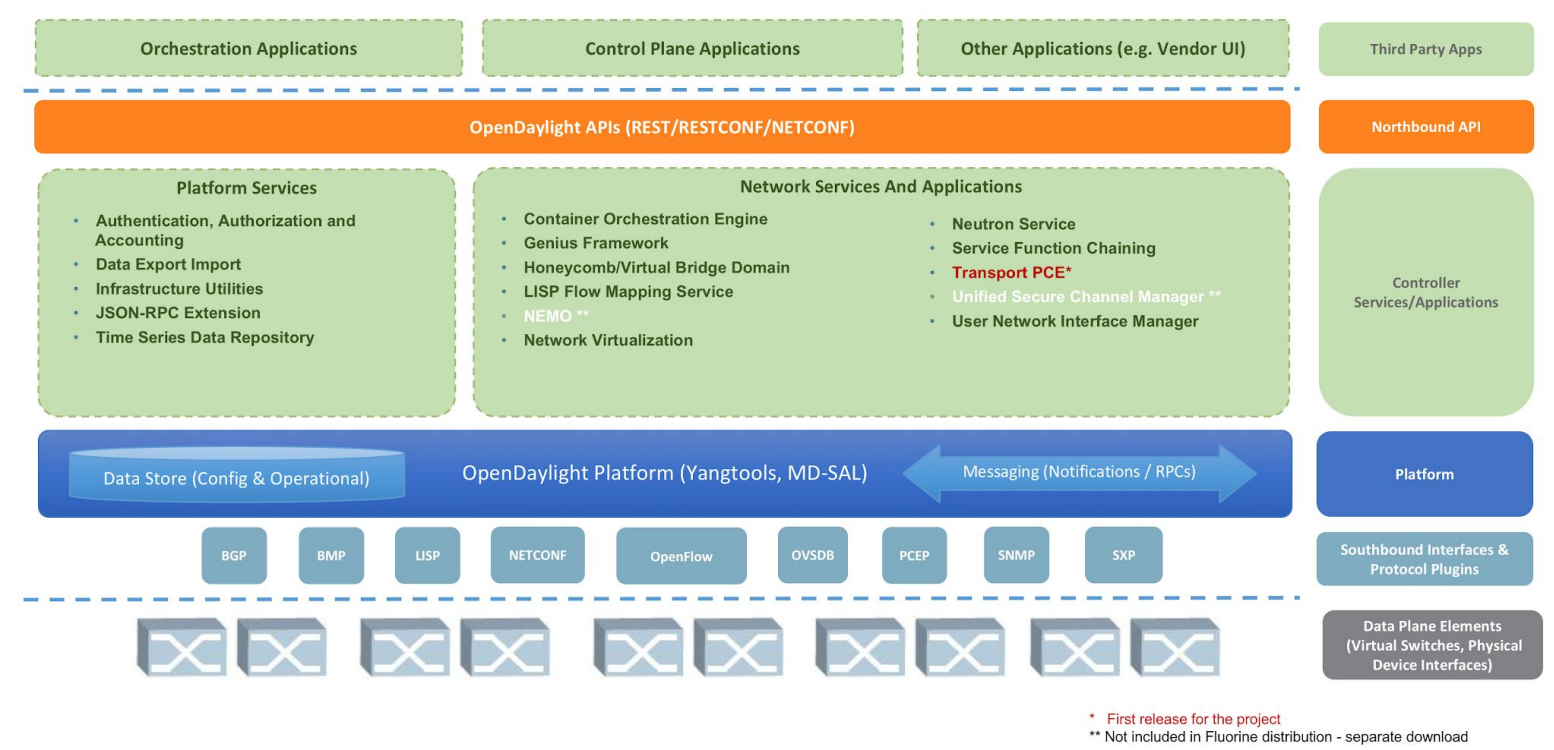
- Network virtualization
- Multi-tenancy
- Security and isolation
  - Control and forwarding planes decoupling
  - AAA
  - Support for TLS, SSH
- Multi-site
  - Federation service
  - Option for tunnel overlays instead of DC-GW

SDN for NFVI

- Resilient
  - Clustering for High Availability, Scalability and Data Persistence
- Rich datapath connectivity options
- Service Function Chaining (SFC)
  - Network slicing
- Integration with MPLS VPNs
- Unique Service Provider applications
- Enhanced policy enforcement mechanisms
  - At applications, services and groups levels*

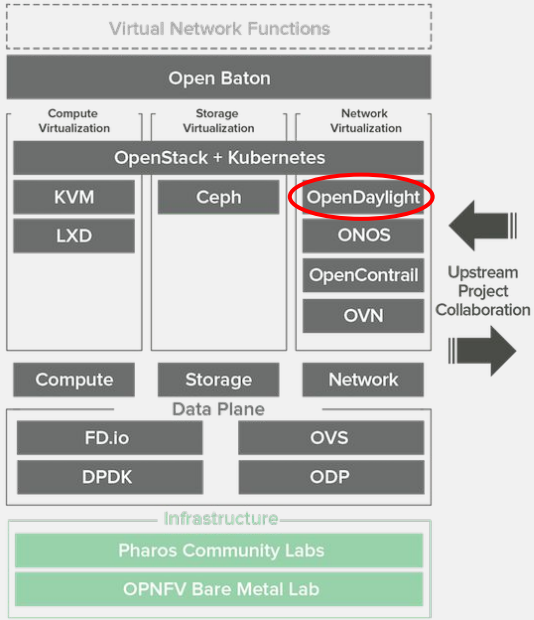
* GBP was moved out of the core ODL applications due to lack of contributions.

OpenDaylight Fluoride Architecture

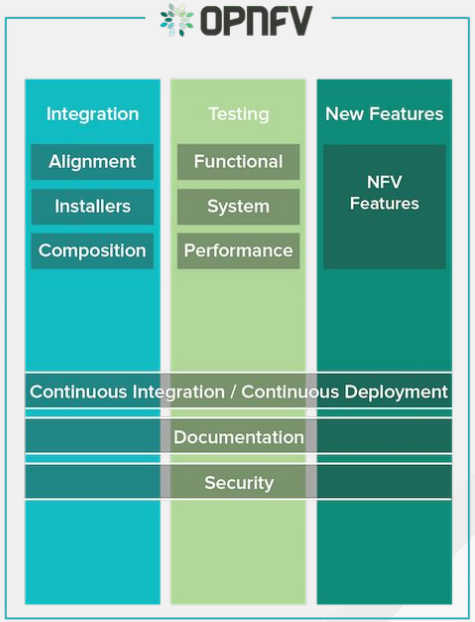


Open Platform for NFV (OPNFV)

- Open source based NFV reference architecture
- Provides integration and testing across ecosystem components
- Deployment framework
- Development assistance
- OpenDaylight Collaboration
  - Deployment image
  - Cross Community CI (XCI)
  - Testing
  - Development

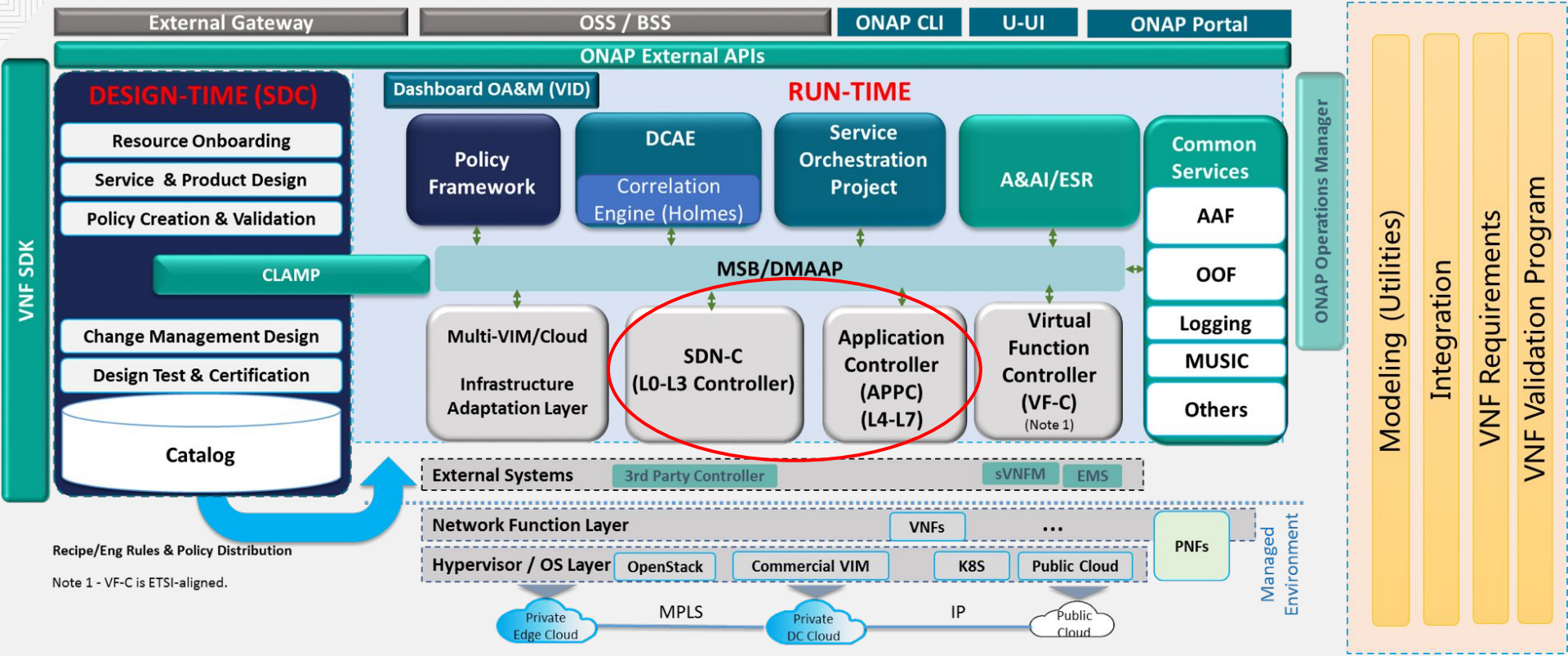


Upstream Project Collaboration



redhat.

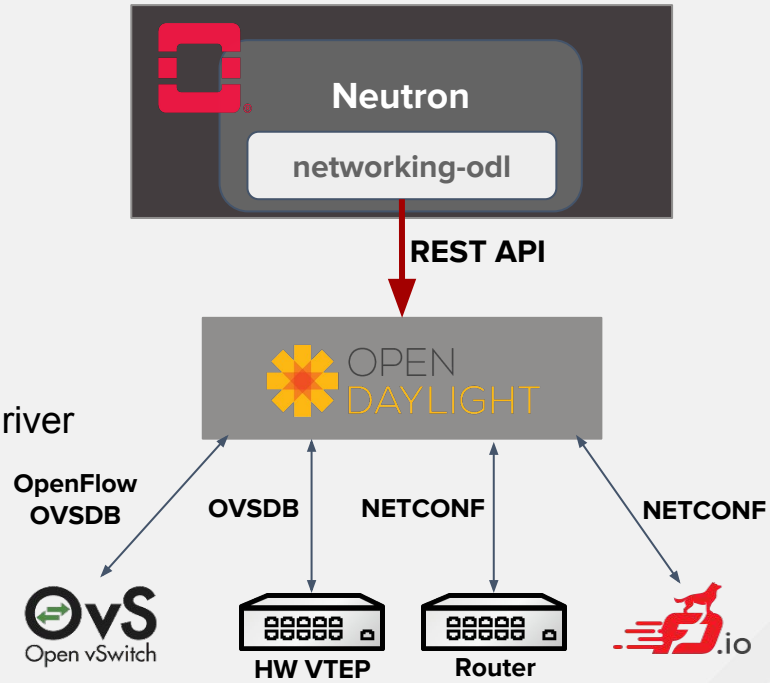
Open Network Automation Platform



OPENSTACK WITH OPENDAYLIGHT

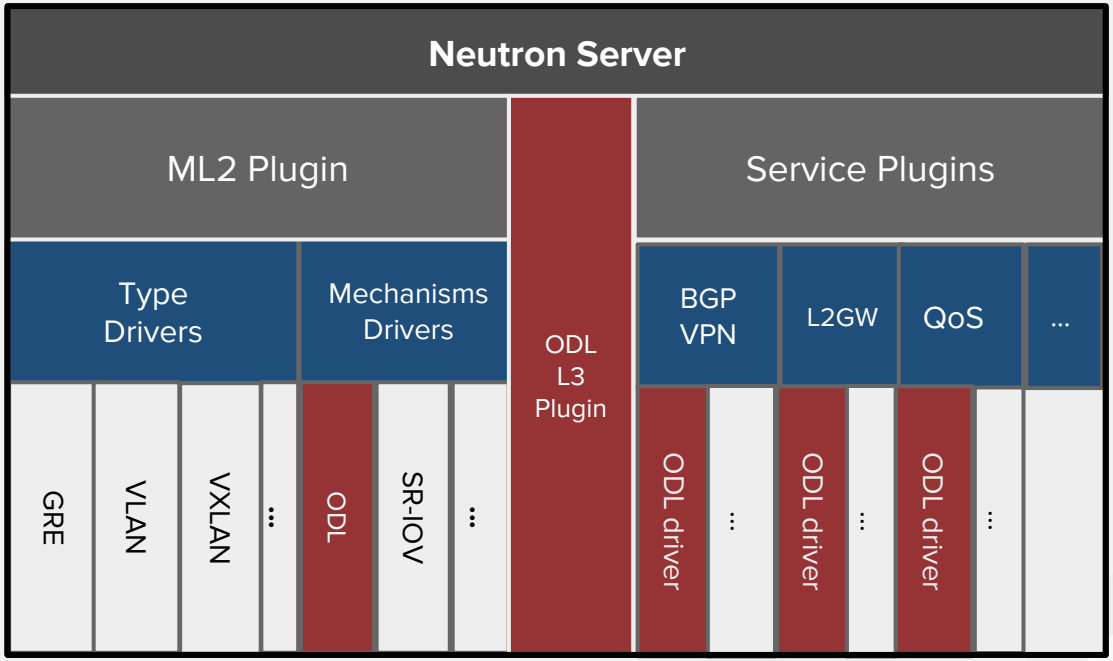
OpenStack with OpenDaylight

- OpenDaylight can be an SDN controller for OpenStack
- Neutron backend
- Replaces Neutron OVS agent
- Provides network virtualization services for OpenStack via the Neutron API
- Supports Neutron API via the **networking-odl** driver
- Can control multiple devices



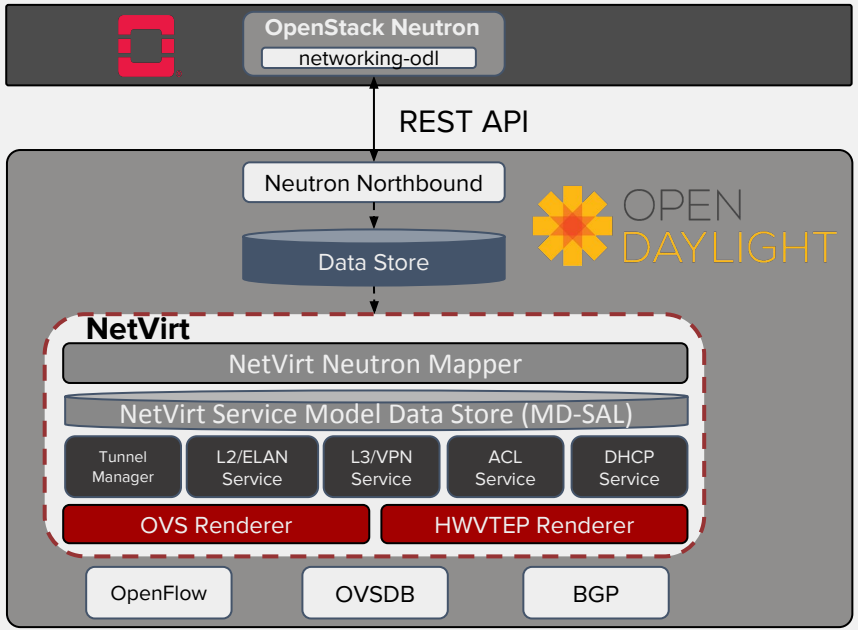
networking-odl

- Push down resource info from Neutron into ODL
- L2: ML2 Plugin
- L3: ODL L3 Plugin
- Services
  - BGP/VPN
  - L2GW
  - QoS
  - SFC
  - VLAN trunk



OpenDaylight NetVirt

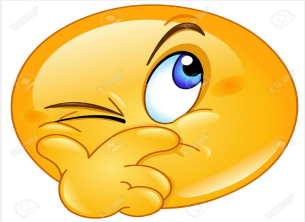
- One of the OpenStack service providers in OpenDaylight
- Translates northbound constructs to forwarding plane agnostic service YANG models
- Services: L2, L3, BGP L3VPN, EVPN, ACL, DHCP, QoS, SFC, IPv6, L2GW
- Supports OpenFlow and OVSDB based devices
- MP-BGP to interwork with physical routers



Integration methods

- Manual installation of OpenStack and ODL binaries and then editing the config files
- Using OpenStack installers
  - Devstack
  - TripleO

OpenStack



- cloud operating system [1]
- OpenStack foundation
- GitHub, Launchpad, Gerrit
- Queens - 17th release, 28 Feb
- Rocky

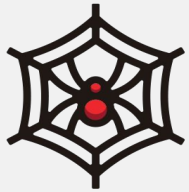
IaaS
PaaS
SaaS

[1] https://www.openstack.org/software/

Collection of Projects

NOVA
an OpenStack Community Project

SWIFT
an OpenStack Community Project



NEUTRON
an OpenStack Community Project

HEAT
an OpenStack Community Project

KOLLA
an OpenStack Community Project

IRONIC
an OpenStack Community Project

CINDER
an OpenStack Community Project

GLANCE
an OpenStack Community Project

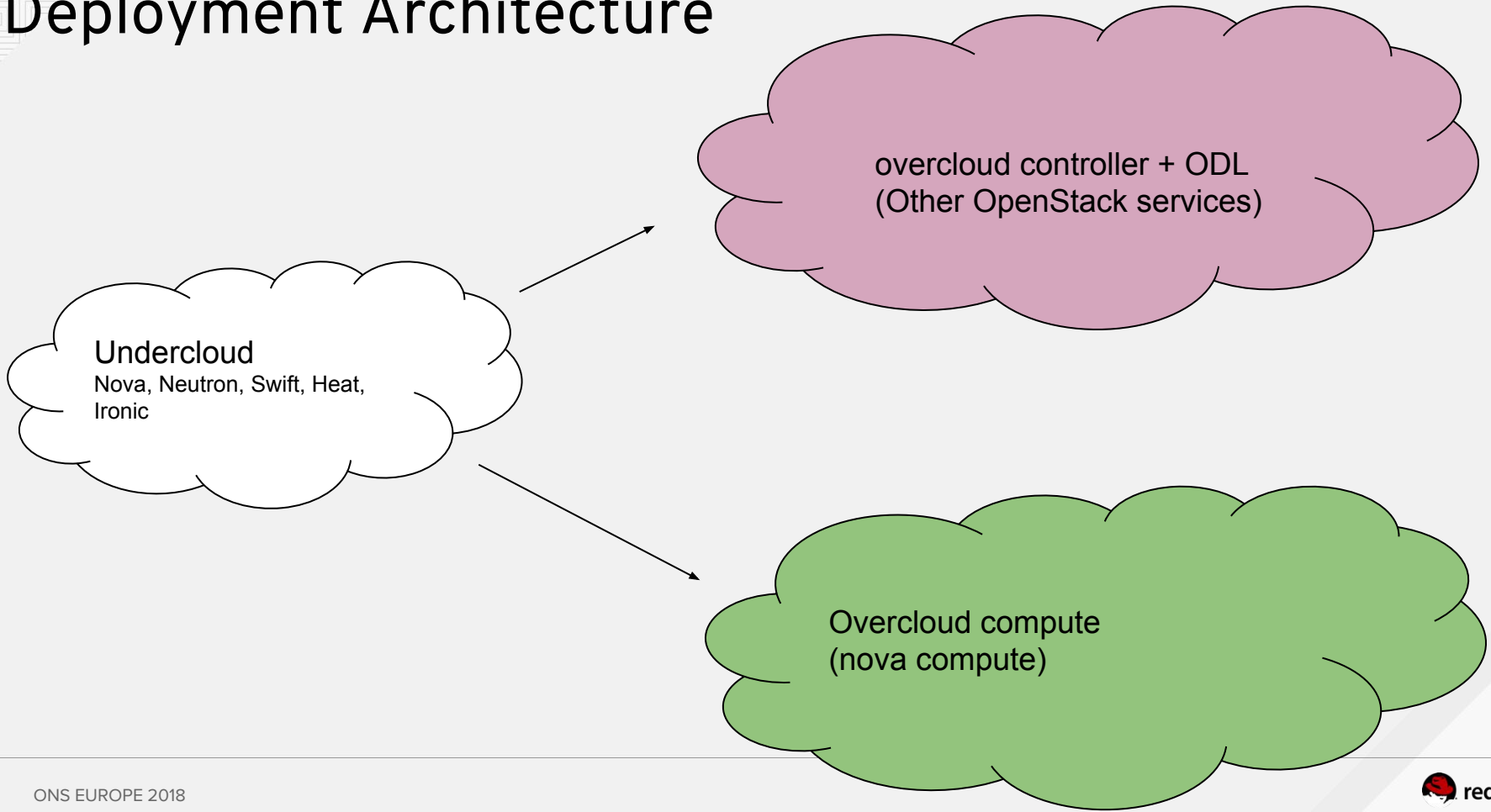
HORIZON
an OpenStack Community Project

TRIPLEO
an OpenStack Community Project

TripleO

- Installer, (Devstack and Packstack)
- OpenStack on top of OpenStack
- CLIs, UI
- Container support from Pike

Deployment Architecture



Deployment Architecture

- Containerised overcloud
  - Run inside containers 
  - Systemctl processes on host 

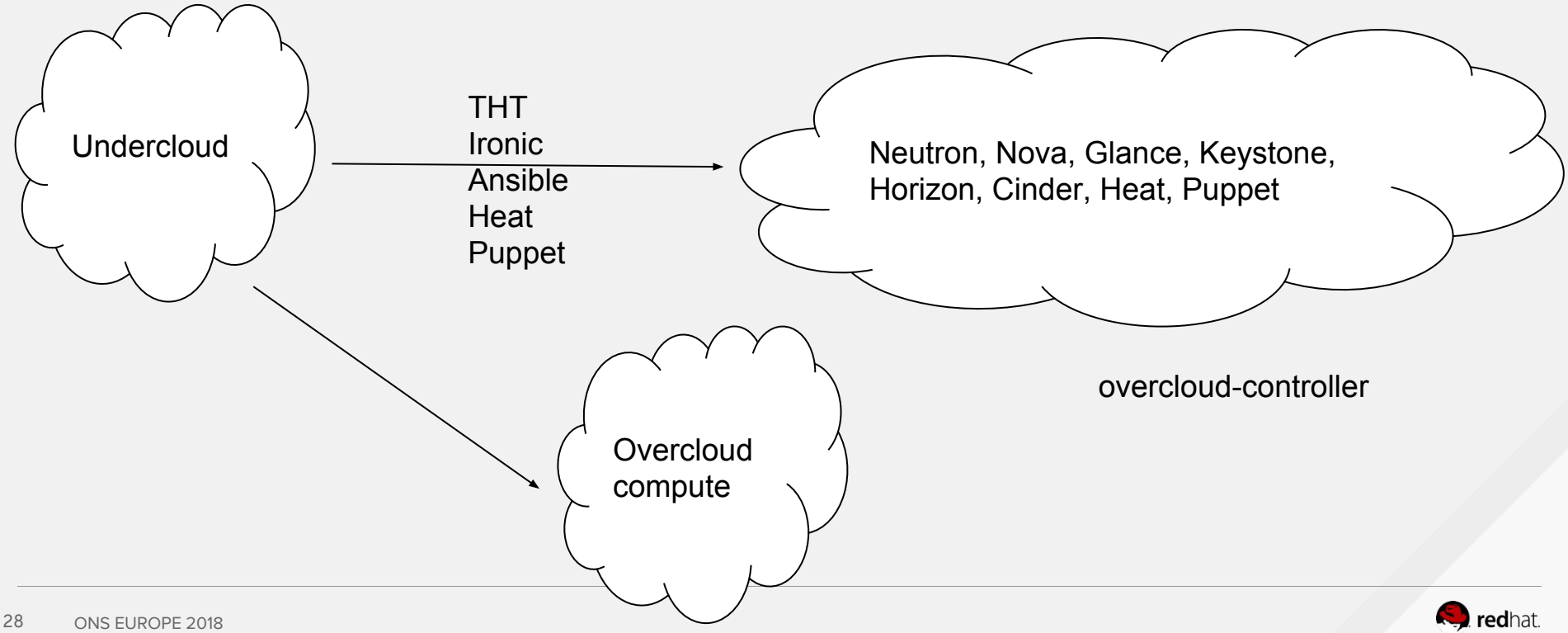
Parts of TripleO

- Collection of projects
- Tripleo-common
- Tripleoclient
- Tripleo-heat-templates
- Puppet-tripleo
- Tripleo-upgrades

Friends of TripleO

- Ironic
- Heat
- Ansible
- Puppet
- Mistral
- Zaqar
- **Kolla**

So far…..



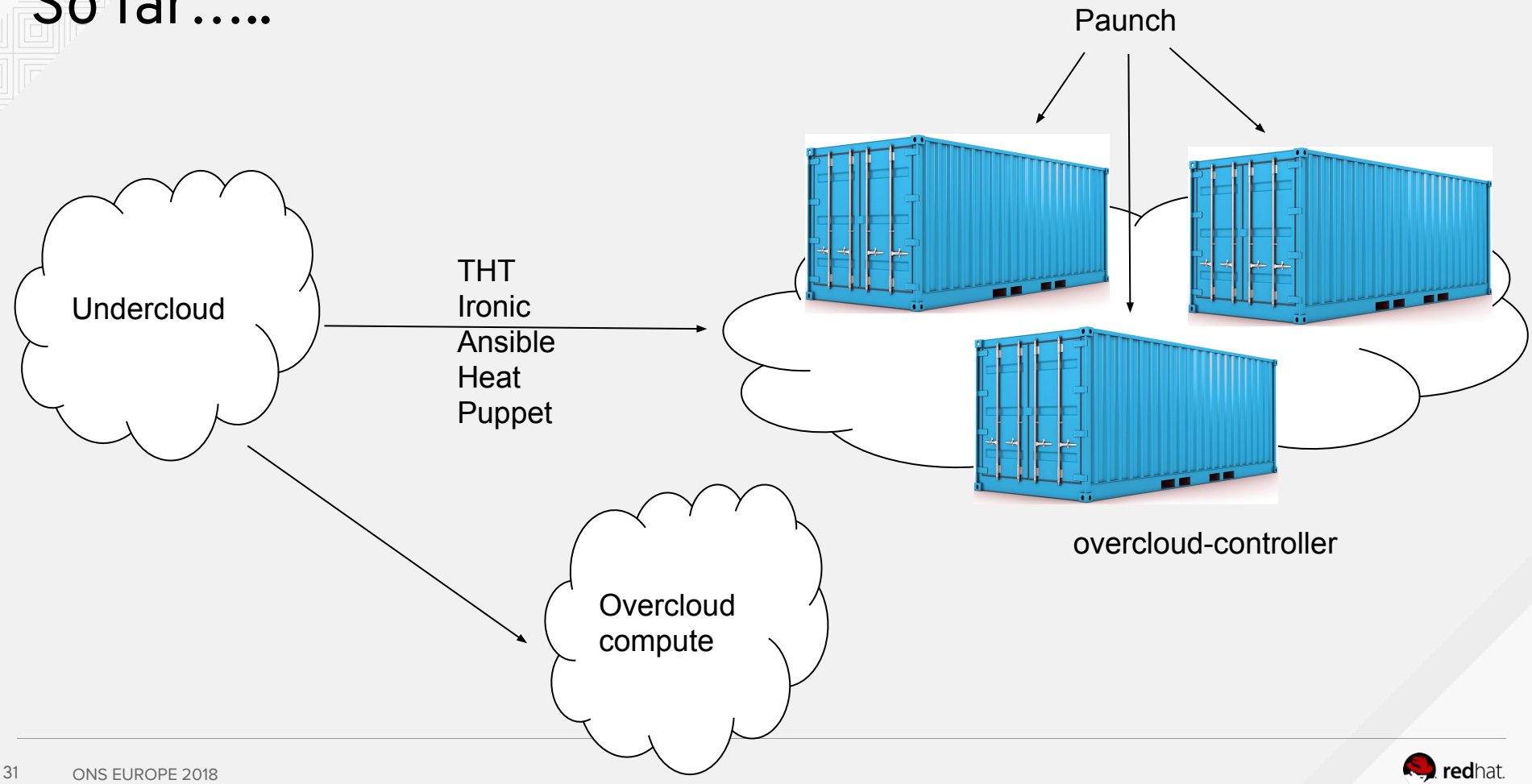
OpenStack in Containers

Dockerfile - Kolla
Locally editable and built
Pushed to Dockerhub
Customised for each OS

Paunch

Lifecycle management
As per yaml files - THT
“host” networking
restart on service restart
“docker_config”

So far…..


Paunch
Undercloud
THT
Ironic
Ansible
Heat
Puppet
overcloud-controller
Overcloud
compute

Containarizing OpenDaylight

1. Create its Dockerfile

https://github.com/openstack/kolla/blob/master/docker/opendaylight/Dockerfile.j2

```
Install odl and other services based on the distro either from source or binary
{% if base_distro in ['centos', 'oraclelinux', 'rhel'] %}
    {% set opendaylight_packages = [
        'java-1.8.0-openjdk-headless',
        'opendaylight',
    ] %}
{% elif base_distro in ['debian', 'ubuntu'] %}
    {% set opendaylight_packages = [
        'default-jre-headless',
        'Opendaylight', ] %}
```

2. Build an image (kolla build)
3. Add THT for configuring ODL container

https://github.com/openstack/tripleo-heat-templates/blob/master/docker/services/opendaylight-api.yaml

```
heat_template_version: rocky
description: >
  OpenStack containerized OpenDaylight API service
parameters:
  DockerOpendaylightApiImage:
    description: image
    type: string
  DockerOpendaylightConfigImage:
    description: image
    type: string
```

```
resources:
  OpenDaylightBase:
    type: ../../puppet/services/opendaylight-api.yaml
outputs:
  role_data:
    description: Role data for the OpenDaylight API role.
    value:
      config_settings:
        map_merge:
          - get_attr: [OpenDaylightBase, role_data, config_settings]
      # BEGIN DOCKER SETTINGS
      puppet_config:
        config_volume: opendaylight
        volumes: <volumes you want to mount>
        step_config:
          get_attr: [OpenDaylightBase, role_data, step_config]
        config_image: {get_param: DockerOpendaylightConfigImage}
      kolla_config:
        /var/lib/kolla/config_files/opendaylight_api.json:
          command: /opt/opendaylight/bin/karaf server
```

```
docker_config:
  step_1:
    opendaylight_api:
      start_order: 0
      image: &odl_api_image {get_param: DockerOpendaylightApiImage}
      privileged: false
      net: host
      detach: true
      user: odl
      restart: unless-stopped
      healthcheck:
        test: /openstack/healthcheck
      volumes:
        list_concat:
          - {get_attr: [ContainersCommon, volumes]}
          - {get_attr: [OpenDaylightApiLogging, volumes]}
          -
            - and any other
```

ODL Puppet Config File

https://github.com/openstack/tripleo-heat-templates/blob/master/puppet/services/opendaylight-api.yaml

```
heat_template_version: rocky
description: >
  OpenDaylight SDN Controller.
parameters:
  OpenDaylightUsername:
    default: 'admin'
    description: The username for the opendaylight server.
    type: string
outputs:
  role_data:
    description: Role data for the OpenDaylight service.
    value:
      service_name: opendaylight_api
      config_settings:
            opendaylight::username: {get_param: OpenDaylightUsername}
```

ODL Puppet Module

Configuring username from THT

https://github.com/opendaylight/integration-packaging-puppet-opendaylight/blob/master/manifests/init.pp#L90

```
class opendaylight (
  $username = $::opendaylight::params::username,

class opendaylight::config {
 # Configure username/password
  odl_user { $::opendaylight::username:
    password => $::opendaylight::password,
    before   => Service['opendaylight'],
  }
```

The Whole Picture

Openstack overcloud deploy …

1. Creates overcloud heat stack (VMs, networks)
2. On each overcloud node,
    a. Run puppet-docker.py - creates a docker container for each puppet-*, mounts files to host
    b. Starts services* container at each THT defined step and mount config files generated in step a. This is done by Paunch

- Can specify which service to run on which node

Customizing ODL Username

```
$ cat odl_username.yaml
paramater_defauls:
 OpenDaylightUsername: admin

$ openstack overcloud deploy <env-files> -e odl_username.yaml
```

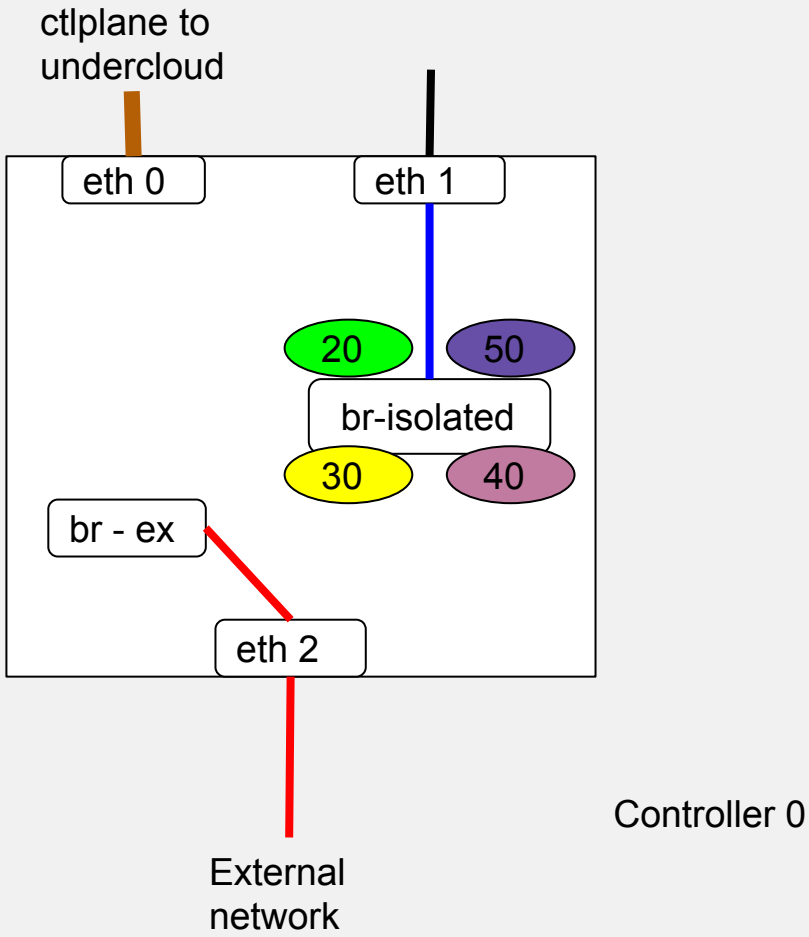
NIC Layout

internal_api 20

storage 30

storage_mgmt 40

tenant 50



Hands-on/Demo

- After deployment, create a VM on overcloud, ping and ssh into it. (commands at https://etherpad.openstack.org/p/ons_tutorial)

Q&A

redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos