Kubernetes ensures that pods are able to network with each other, and allocates each pod an IP address from an internal network. This ensures all containers within the pod behave as if they were on the same host. Giving each pod its own IP address means that pods can be treated like physical hosts or virtual machines in terms of port allocation, networking, naming, service discovery, load balancing, application configuration, and migration.

Creating links between pods is unnecessary, and it is not recommended that your pods talk to one another directly using the IP address. Instead, it is recommend that you create a service, then interact with the service.

If you are running multiple services, such as frontend and backend services for use with multiple pods, in order for the frontend pods to communicate with the backend services, environment variables are created for user names, service IPs, and more. If the service is deleted and recreated, a new IP address can be assigned to the service, and requires the frontend pods to be recreated in order to pick up the updated values for the service IP environment variable. Additionally, the backend service has to be created before any of the frontend pods to ensure that the service IP is generated properly, and that it can be provided to the frontend pods as an environment variable.

For this reason, OpenShift Origin has a built-in DNS so that the services can be reached by the service DNS as well as the service IP/port.

OpenShift Origin supports the Kubernetes Container Network Interface (CNI) as the interface between the OpenShift Origin and Kubernetes. Software defined network (SDN) plug-ins are a powerful and flexible way to match network capabilities to your networking needs. Additional plug-ins that support the CNI interface can be added as needed.

The following network plug-ins are currently supported by OpenShift Origin:

- OpenShift SDN
- OpenShift Network Isolation SDN
- Flannel SDN
- Contiv SDN
- Nuage Networks SDN

OpenShift SDN

OpenShift Origin deploys a software-defined networking (SDN) approach for connecting pods in an OpenShift Origin cluster. The OpenShift SDN connects all pods across all node hosts, providing a unified cluster network.

OpenShift SDN is installed and configured by default as part of the Ansible-based installation procedure. See the OpenShift SDNsection for more information.

Flannel SDN

flannel is a virtual networking layer designed specifically for containers. OpenShift Origin can use it for networking containers instead of the default software-defined networking (SDN) components. This is useful if running OpenShift Origin within a cloud provider platform that also relies on SDN, such as OpenStack, and you want to avoid encapsulating packets twice through both platforms.

Architecture

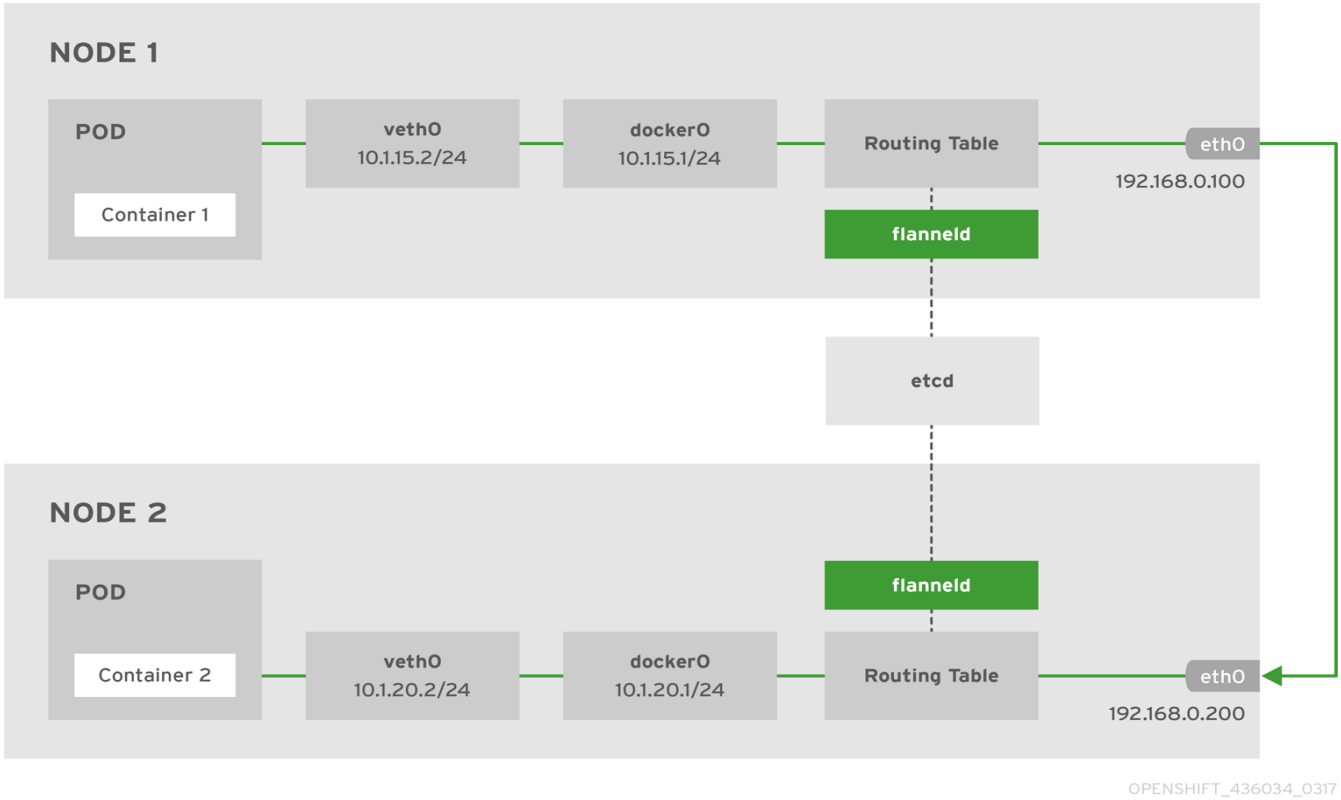
OpenShift Origin runs **flannel** in **host-gw** mode, which maps routes from container to container. Each host within the network runs an agent called **flanneld**, which is responsible for:

- Managing a unique subnet on each host

- Distributing IP addresses to each container on its host
- Mapping routes from one container to another, even if on different hosts

Each **flanneld** agent provides this information to a centralized **etcd** store so other agents on hosts can route packets to other containers within the **flannel** network.

The following diagram illustrates the architecture and data flow from one container to another using a **flannel** network:



Node 1 would contain the following routes:

```
default via 192.168.0.100 dev eth0 proto static metric 100

10.1.15.0/24 dev docker0 proto kernel scope link src 10.1.15.1

10.1.20.0/24 via 192.168.0.200 dev eth0
```

Node 2 would contain the following routes:

```
default via 192.168.0.200 dev eth0 proto static metric 100

10.1.20.0/24 dev docker0 proto kernel scope link src 10.1.20.1

10.1.15.0/24 via 192.168.0.100 dev eth0
```

Contiv SDN

Contiv is an open-source networking plug-in module for container infrastructure. Contiv provides an infrastructure for application-oriented network policies and support for a range of multiple networking modes. These include:

- A configurable set of overlay networking modes.
- Physical networking modes.
- Support for industry-leading hardware.

OpenShift Origin can use Contiv for networking containers instead of the default OpenShift SDN.

Architecture

Each node within the cluster runs a Contiv agent called **netplugin** while the master hosts run the Contiv controller (called **netmaster**), along with supporting control plane components (such as **etcd**).

Together the components of Contiv (**netmaster** and **netplugin**) handle key networking functions for OpenShift Origin including:

- Assigning IP addresses to each container pod on each cluster node.
- Creating and managing multiple separate container network instances for different groups of containers.
- Configuring the network forwarding layer components for layer two or layer three forwarding.
- Configuring and enforcing a range of network policies.
- Providing management interfaces (including both CLI and GUI) to configure and manage Contiv-specific features and configurations.
- Providing an infrastructure for role-based controls that allow for multiple role-based network operations workflows.

Contiv uses the Container Network Interface (CNI) to interface with OpenShift Origin and Kubernetes. A key value store based on **etcd** is used to store Contiv-specific state information. This is in addition to and separate from the instance of **etcd** used by other components in the system, including OpenShift Origin itself.

Nuage SDN for OpenShift Origin

Nuage Networks' SDN solution delivers highly scalable, policy-based overlay networking for pods in an OpenShift Origin cluster. Nuage SDN can be installed and configured as a part of the Ansible-based installation procedure. See the Advanced Installationsection for information on how to install and deploy OpenShift Origin with Nuage SDN.

Nuage Networks provides a highly scalable, policy-based SDN platform called Virtualized Services Platform (VSP). Nuage VSP uses an SDN Controller, along with the open source Open vSwitch for the data plane.

Nuage uses overlays to provide policy-based networking between OpenShift Origin and other environments consisting of VMs and bare metal servers. The platform’s real-time analytics engine enables visibility and security monitoring for OpenShift Origin applications.

Nuage VSP integrates with OpenShift Origin to allows business applications to be quickly turned up and updated by removing the network lag faced by DevOps teams.

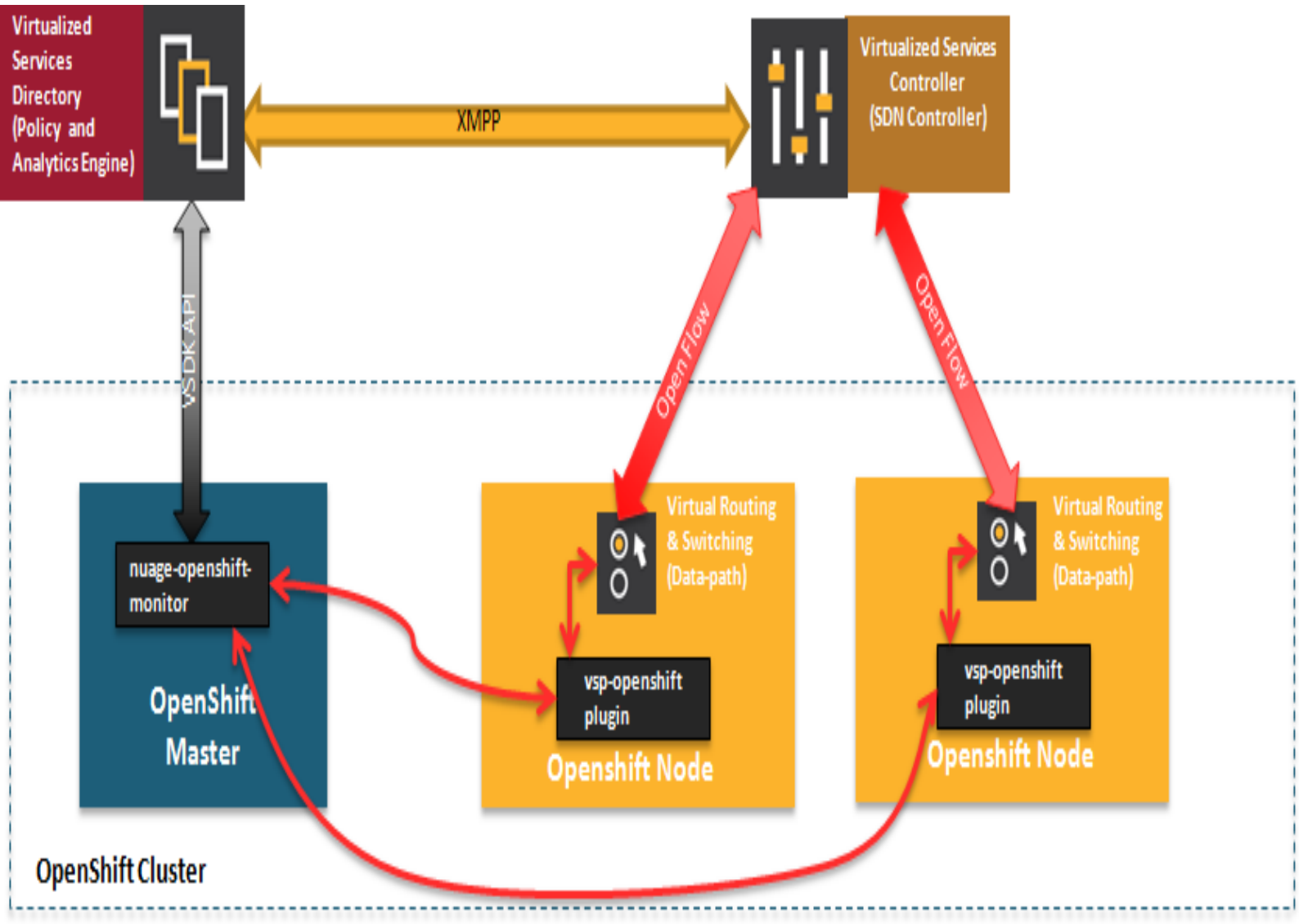


Figure 1. Nuage VSP Integration with OpenShift Origin

There are two specific components responsible for the integration.

1. The **nuage-openshift-monitor** service, which runs as a separate service on the OpenShift Origin master node.
2. The **vsp-openshift** plug-in, which is invoked by the OpenShift Origin runtime on each of the nodes of the cluster.

Nuage Virtual Routing and Switching software (VRS) is based on open source Open vSwitch and is responsible for the datapath forwarding. The VRS runs on each node and gets policy configuration from the controller.

Nuage VSP Terminology

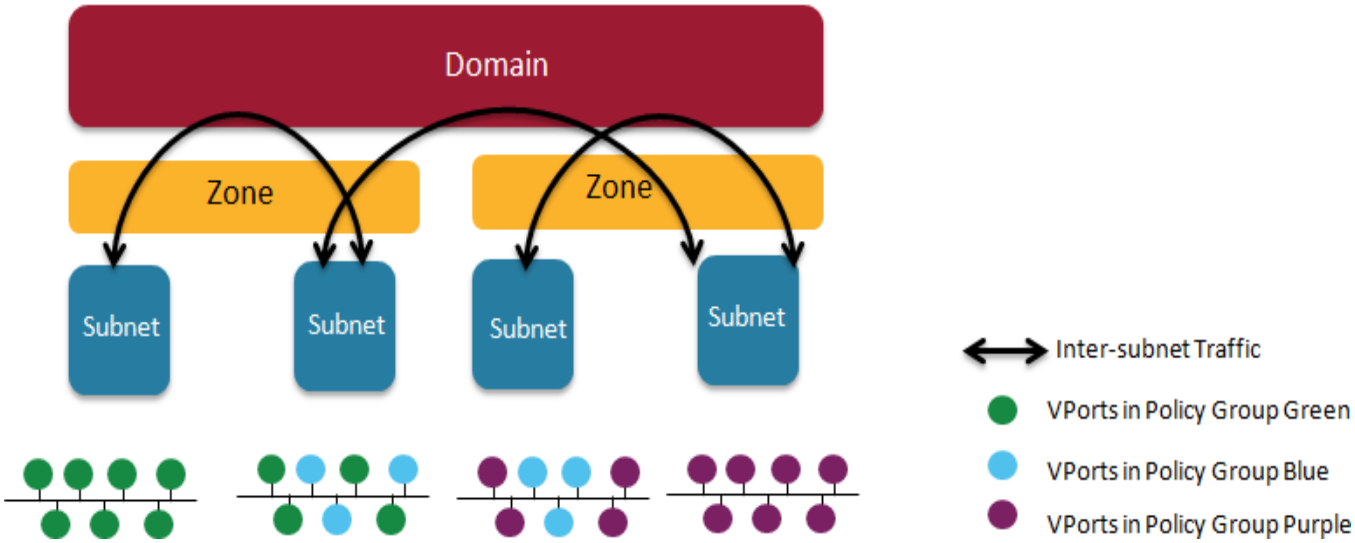


Figure 2. Nuage VSP Building Blocks

1. Domains: An organization contains one or more domains. A domain is a single "Layer 3" space. In standard networking terminology, a domain maps to a VRF instance.
2. Zones: Zones are defined under a domain. A zone does not map to anything on the network directly, but instead acts as an object with which policies are associated such that all endpoints in the zone adhere to the same set of policies.
3. Subnets: Subnets are defined under a zone. A subnet is a specific Layer 2 subnet within the domain instance. A subnet is unique and distinct within a domain, that is, subnets within a Domain are not allowed to overlap or to contain other subnets in accordance with the standard IP subnet definitions.
4. VPorts: A VPort is a new level in the domain hierarchy, intended to provide more granular configuration. In addition to containers and VMs, VPorts are also used to attach Host and Bridge Interfaces, which provide connectivity to Bare Metal servers, Appliances, and Legacy VLANs.
5. Policy Group: Policy Groups are collections of VPorts.

F5 BIG-IP® Router Plug-in

A router is one way to get traffic into the cluster. The F5 BIG-IP® Router plug-in is one of the available router plugins.

The F5 router plug-in integrates with an existing **F5 BIG-IP®** system in your environment. **F5 BIG-IP®** version 11.4 or newer is required in order to have the F5 iControl REST API. The F5 router supports unsecured, edge terminated, re-encryption terminated, and passthrough terminated routes matching on HTTP vhost and request path.

The F5 router has feature parity with the HAProxy template router, and has additional features over the **F5 BIG-IP®** support in OpenShift v2. Compared with the **routing-daemon** used in earlier versions, the F5 router additionally supports:

- path-based routing (using policy rules),
- re-encryption (implemented using client and server SSL profiles)
- passthrough of encrypted connections (implemented using an iRule that parses the SNI protocol and uses a data group that is maintained by the F5 router for the servername lookup).