SDN Deep Dive

Jaime Caamaño Ruiz
jcaamano@suse.com

Ruben Tornero Marijnissen
rtorreromarijnissen@suse.com

Team Networking
Topics

- Why SDN?
- What is SDN?
- SDN in OpenStack and K8s
- Overview of SDN controllers
Why SDN?

Limitations of Traditional Networking
Traditional networking
It's hardware centric!
Closed systems

- Vendor specific software
- Costly
- Hard to inter-operate
Not scalable!
No abstractions

- Hard to maintain
- Hard to innovate
- Hard to experiment
Server virtualization

VLANs are not flexible enough (e.g. server is moved)
Traffic differs from the classic server-client model
Connect a new machine

1. Reach the place

2. Plug the cable

3. Configure
What is SDN?
What's SDN Goal?

Enable network engineers and administrators to respond quickly to changing business requirements
How does SDN work?

- Separate control plane from data plane
- Centralization of control
- Program a network vs configure network
- Forwarding decisions are flow based
Separate control plane from data plane

Switch

Control Plane
- Build information
  - ARP, routing protocols, MAC Learning
- Store information
  - L2/L3 forwarding tables
- Forwarding Decision

Data Plane
- Forwarding Path

Externally controlled Switch

OpenFlow Controller
- Build information
  - Programmatic
- Store information
  - Policy, Topology
- Forwarding Decision

Switch
- OpenFlow Interface API
- Data Plane
- Forwarding Path
Centralization of control
Flow based forwarding

- A flow of packets are those that should be forwarded in the same way.
- A packet is classified into a flow by data contained within the packet (packet headers).
- A packet is forwarded by applying a set of actions to it.
- Those actions will be the same for all packets of the same flow.

- An abstraction to packet-switching that allows to design and control pure forwarding network devices.
OpenFlow

- Open standard
- Separation of control plane and data plane
  - OF switch has flow tables
  - OF controller programs the flow entries
- Flow = match + action
OpenFlow switch

Packet In

Table 0
Ingress port
Action Set = {}

Table 1
Packet + ingress port + metadata
Action Set

Table n
Packet Action Set

Execute Action Set

Packet Out
Overlay network

- Encapsulation decouples a network service from the underlying infrastructure
SDN Benefits

• Simpler hardware
  ○ Controller runs on commodity hardware
  ○ Network devices are pure forwarding elements
  ○ Independent development of software and hardware
  ○ Reduced CapEx

• Network becomes a computation/software problem
  ○ Software abstractions and open standards
  ○ Easier to innovate, design, deploy, manage and scale
  ○ Improved flexibility and agility
  ○ Reduced OpEX

• Automation, Optimization and Integration
SDN Use Cases

● For carriers and service providers
  ○ Network resource optimization
  ○ SD-WAN
  ○ NFV
● For enterprise
  ○ Network access control
  ○ Network monitoring
● For cloud computing and data center workloads
  ○ Network virtualization
  ○ Automated service delivery
SDN Characterization

- Cross platform or hardware specific?
- Open vs. proprietary
- Southbound protocols
- Northbound APIs & services
- Networking features
- Data plane stack: overlay protocol, hypervisor vSwitch, acceleration...
- Efficiency: performance, reliability, scalability...
- Integration: OpenStack, Kubernetes, Cloud-Native...
- Monitoring & Analytics features
SDN networking in OpenStack
Connect a new machine in the virtual world
Neutron

- Neutron is an OpenStack project to provide “networking as a service” between interface devices (e.g., vNICs) managed by other Openstack services (e.g., nova)
- provides a powerful API to define the network connectivity
Neutron abstractions

- Network: L2 broadcast domain
- Subnet: a block of v4 or v6 IP addresses and associated configuration state.
- Port: a connection point for attaching a single device, such as the NIC of a virtual server, to a virtual network. Also describes the associated network configuration, such as the MAC and IP addresses to be used on that port.
- Router: interconnects networks
Modular architecture

- Plugin: custom back-end implementation of the Networking API
- Neutron-server: exposes the API
Neutron as SDN controller
Neutron as SDN application
SDN networking in K8s
Containers are cool but...

- Containers need to be reachable
- Containers need to be connected together
Container Network Interface

Container Runtime

Container Network Interface (CNI)

Built-in:
- loopback
- bridge
- ipvlan
- dhcp
- flannel

Third-party:
- calico
- cilium
- SDN
Mixing it all with SDN

Kubernetes

- Container Runtime
- Container Network Interface (CNI)
- SDN Plugin

OPENSTACK

- HORIZON / CLI
  - Client
- NOVA
  - Compute
- GLANCE
  - Image Service
- NEUTRON
  - Networking
  - Plugin

KEYSTONE

- Identity Service

SDN CONTROLLER

- CNI Agent
- Northbound App
- Openstack Agent

Control Layer Abstractions

Southbound protocols

Network Elements
SDN controllers overview

CISCO ACI
Cisco ACI: Overview (I)

- Cisco’s approach: Application requirements to define the network behavior
  - Policy-driven solution
  - Combining both SW and HW
  - Common platform for physical, virtual, and cloud.

- IPv6 support

- Protocol Stack
  - Northbound REST APIs
  - Southbound OpFlex agents
  - Overlay support: NVGRE, VXLAN
Cisco ACI: Overview (II)

● HA support
  ○ 2 member active/standby APIC controller cluster

● Multi-Hypervisor
  ○ KVM
  ○ ESXi
  ○ Hyper-V

● Integrations
  ○ OpenStack
  ○ Kubernetes
  ○ Cloud
What is ACI?

ACI Fabric

Group Based Policy

Controller

Application Centric Infrastructure
OpenStack integration
Kubernetes Integration
SDN controllers

Tungsten Fabric
Tungsten Fabric: Overview (I)

● Open Source & Part of the Linux Foundation
● Application-based security policies
● IPv6 support

● Protocol Stack
  ○ REST APIs & Python bindings
  ○ XMPP Southbound agents
  ○ MPLSoGRE & VXLAN overlay

● Interesting network features
  ○ BGPaaS
  ○ SFC
Tungsten Fabric: Overview (II)

- **Dataplane optimizations in TF vRouter:**
  - DPDK
  - SR-IOV
  - SmartNIC

- **Supports HA:**
  - active/active (for LB and failover)

- **Containerized control plane**
OpenStack & Kubernetes Integration

Cloud orchestrator can be any combination of OpenStack, Kubernetes, OpenShift, ...

Contrail controller listens on orchestrator event bus for workload-related events sends XMPP messages to vRouters to implement security policies

User

Users access portal over enterprise intranet or Internet

Data Center

Portal

VM

Virtual networks have VRFs with routes and ACLs in vRouters to implement network and application policies

Private Cloud Cluster

Pod of containers

Traffic is carried between vRouters using encapsulation tunnels (MPLSoUDP, MPLSoGRE, VXLAN)

Image from tungsten.io
SDN controllers

VMWare NSX-T
NSX-T: Overview (I)

- Software driven, virtual appliances
- IPv6 support
- HA
  - 3 node clustering

- Multi-hypervisor
  - KVM
  - native vCenter support

- Integrations
  - Kubernetes
  - OpenStack
NSX-T: Overview (II)

- Multi-Cloud
  - Azure
  - AWS
  - ...

- Protocol stack:
  - Custom OvS & southbound agent for KVM
  - Overlay: Geneve
  - Northbound REST APIs

- Dataplane optimizations
  - For ESXi
  - Enhanced N-VDS (DPKD-based)
OpenStack integration
Kubernetes integration
SDN controllers

OpenDaylight
OpenDaylight: Overview

- Open Source & Part of the Linux Foundation
- Multi-project platform
- Multiple Southbound protocol support
- Modular Northbound services & APIs
- Cross-platform: Java
- Perfect for learning & SDN innovation
Modular Architecture

Image from opendaylight.org
Thanks!

Questions?