Lecture 13: Software-Defined Networking

Fall 2022

Joshua Reynolds

Thanks to Sandip Chakraborty from IIT Kharagpur

Software Defined Networking

Separate:

- router configuration and control
- Hardware actually doing packet forwarding

Allows:

- Centralized control of all routing in large networks
- Centralized security monitoring of large networks
- Network users just need to plug in a device and it “just works”

Times when you are glad you have SDN

When you need to replace 100 routers in a datacenter, add 100 more, and configure them all

When you want to change the way routing works in your datacenter without changing the 100,000 ethernet cables that are already plugged in

You are partway through adding 100 routers and your manager gets an order for 100 more, and you have to go find the 57 routers you already configured to adjust them to route to these 100 extra routers.

When you have 250 routers built by 3 different companies in 40 model types to manage

Control Plane vs Data Plane

“Plane” in the geometry sense

Control Plane:

- Decides and communicates routing rules and policies
- Low bandwidth, high CPU needs

Data Plane:

- Carries out decisions from the control plane
- High bandwidth, low CPU needs

Different Router Vendors - Different Proprietary Planes

CISCO

Huawei

Juniper Networks

Arista

Lock you into their ecosystem!

Use a data plane built by anyone, and a unified control plane

An SDN network has 2 types of devices (no routers!)

1. **Network Controllers**

   Any computer with a CPU

2. **Switches**

   “blind”

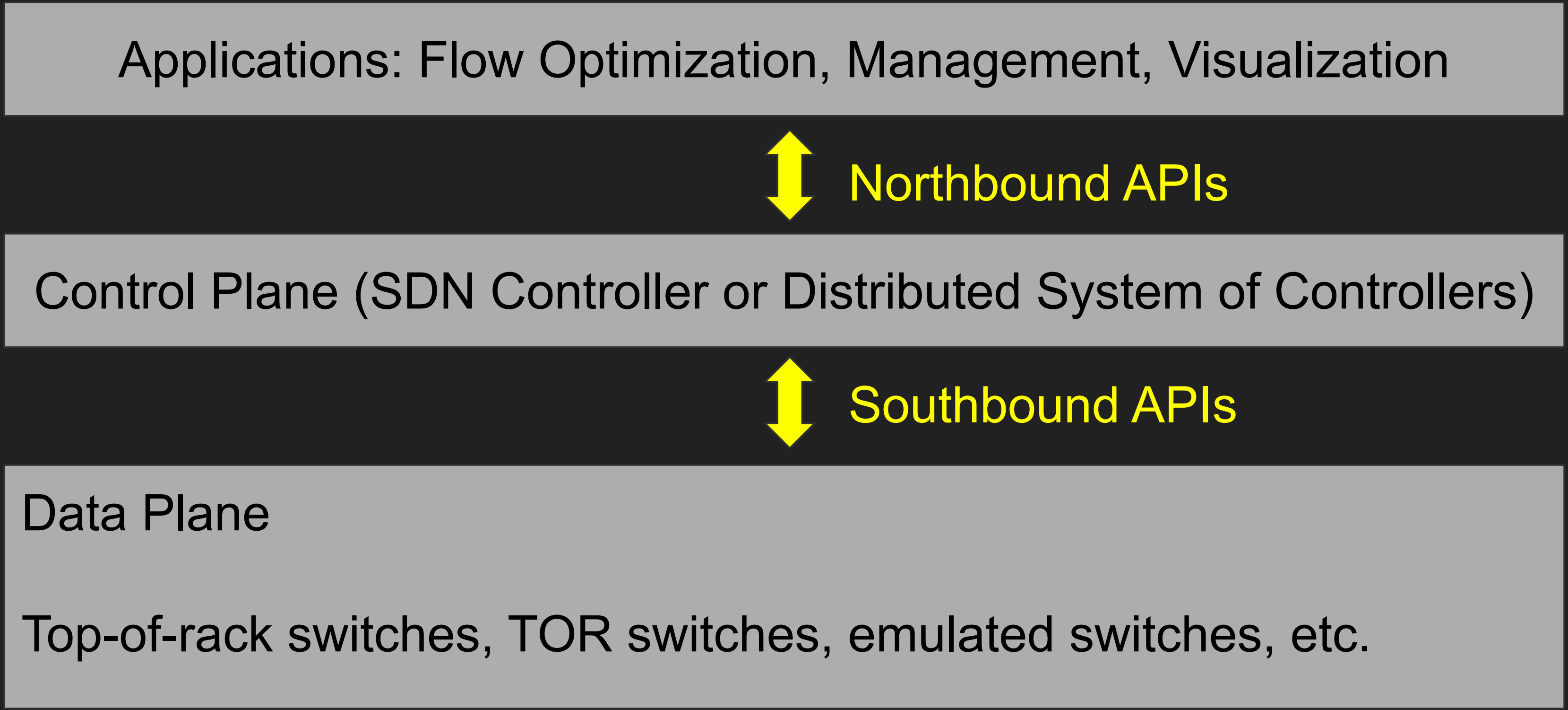
   They take instruction from network controllers as to where to forward packets

Controller to Switch Communication

Hey controller! I see this TCP packet for 10.45.33.11:6881
I don't have any applicable rule. What do I do?

Hey switch! Here is the rule to follow: 10.45.33.0/24 TCP DENY-ALL port 6880-7000

SDN Topology, a Distributed Sysem



Applications: Flow Optimization, Management, Visualization

Northbound APIs

Control Plane (SDN Controller or Distributed System of Controllers)

Southbound APIs

Data Plane

Top-of-rack switches, TOR switches, emulated switches, etc.

OpenFlow - a controller-switch communication protocol

OpenFlow uses TCP in a client-server model where the controller is the server and the switches are the clients.

The controller sends **rules** to the switches to follow.

The rule can match combinations of things like SRC MAC, DST MAC, SRC IP, DST IP, etc (anything from any layer’s packet header)

The rule points to an action, usually the layer 2 interface where this switch should forward the packet to.

It can also drop packets, NAT translation, send to multiple outputs

OpenFlow Rule Tables

In addition to matching criteria and actions, rules also store:

Statistics on how many packets and bytes have been forwarded on this rule.

Switches with powerful CPUs can execute simple programs on data flows.

Rules can also state their level of priority (precedence)

OpenFlow Rule Example Activity

Implement a firewall rule blocking incoming HTTP requests

Implement a firewall rule blocking any outgoing connection to a BitTorrent tracker

Implement a rule for NAT translation

Implement a rule to only allow an SMTP server to respond to the internal network

Implement a rule to send browser data over one path, and OS updates over another???

Flow Specific Routing

With these rules, you can have different processes get routed differently from the same computer!

Just like how ports let us have multiple TCP conversations going, we can make rules for every combination of <SRC IP, DST IP, SPORT, DPORT>

OpenFlow Protocol

OpenFlow Hello: I support version 1.5

OpenFlow Hello: I support version 1.4.8, let’s go with that

Hey switch, what features do you have?

Hey controller, I am a brainless switch with 4 ethernet ports.
I call them eth0 eth1 eth2 and eth3

Controller to Switch Communication

Hey controller! I see this TCP packet for 10.45.33.11:6881
I don’t have any applicable rule. What do I do?

Hey switch! Here is the rule to follow: 10.45.33.0/24 TCP
DENY-ALL port 6880-7000

Controller to Switch Communication

Hey controller! You still there? ECHO REQUEST

Hey switch! I am still here, just have nothing to say. ECHO RESPONSE

Other OpenFlow messages

Failed to apply a rule

A physical ethernet port is broken or unplugged

Virtualized Networks and Distributed Systems

Virtual Machines

Type 1 Hypervisor - runs on bare metal

ESXi

Windows Server Hyper-V

Citrix Xen

Red Hat Enterprise Virtualization

Type 2 Hypervisor - runs on top of an OS

VMWare

VirtualBox

QEMU

Virtualized Networking

VMWare NSX

SolarWinds Virtualization Manager

Cisco Enterprise Network Functions Virtualization

Containers

Virtual machine “lite”

Docker

Containerd

Built using OS process isolation features

In Linux, can emulate a different distro

Linux Kernel CGroups

Limits a group of processes to specific subset of hardware in the computer (cores, RAM, network card)

Available since kernel version 2.6.24 (2008)

Linux Kernel Namespaces

Like CGroups, but for kernel resources instead of hardware

Different PID numbering per namespace

No IPC outside the namespace

Kubernetes

An army of computers

All can be different hardware

Identical containers can be deployed across all of them

Containers can be redundant backups for each other

Containers can scale to match load

And it is a distributed system!

Kubernetes Node Types

Kube-apiserver (APIs to control the cluster)

Etcd (distributed key-value store)

Kube-scheduler (Decides where to put pods, groups of containers sharing resources)

Kube-controller-scheduler (Watches for node failures, runs one-off jobs, manages namespaces)

Cloud-controller-manager (Specifically for integrating with cloud resources)

Kubelet (talks to cluster to prioritizing co-locating containers that need to share resources. Containers w/ shared resources called a pod)

Kube-proxy (SDN node w/ customizable firewall or VPN)

Container-runtime (the program provisioning containers e.g. Docker)

Deploying an application on Kubernetes

Connect to the kube-apiserver (e.g. through HTTPS Web control interface)

Upload a config file (like a Dockerfile) along with any needed resources (or a command to clone from a repo)

Specify how many container copies you want, and whether you want them restarted if they die.

Runs a distributed system inside a distributed system!

Totally agnostic to underlying hardware

For application developers

Still need folks running networking and keeping the OSes up to date

Kubernetes is used by

Google

Microsoft

Alibaba

IBM

Oracle

Amazon

System Design Interview Review Activity

Build a deployment pipeline