Network Layer – Control Plane Part III

ECE 6607
GATECH Shenzhen, Fall 2022

Dr. Min Luo

1

Chapter Goals

- *Goals: To* understand principles behind network control plane
  - traditional routing algorithms
  - SDN controllers
  - Internet Control Message Protocol
  - network management
- and their instantiation, implementation in the Internet:
  - OSPF, BGP, OpenFlow, ODL and ONOS controllers, ICMP, SNMP

2

Outline

- Intro
- Routing protocols
  - link state vs. distance vector
- Intra-AS routing in the Internet: OSPF
- Routing among the ISPs: BGP
- **The SDN control plane**
- ICMP: The Internet Control Message Protocol
- Network management and SNMP

3

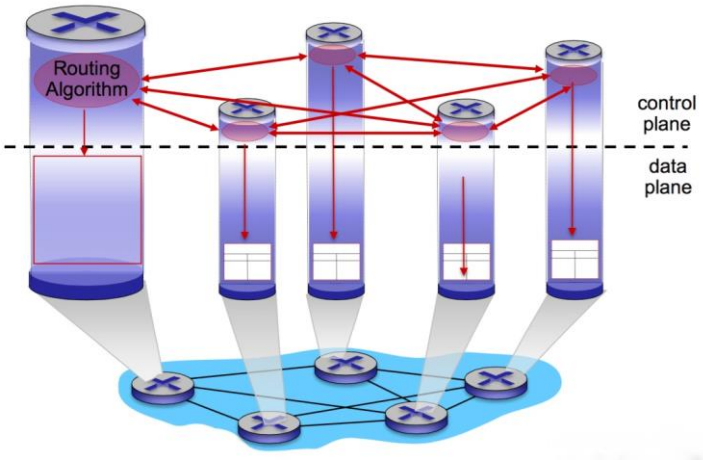
Software defined networking (SDN)

- Internet network layer: historically has been implemented via distributed, per-router approach
  - *monolithic* router contains switching hardware, runs proprietary implementation of Internet standard protocols (IP, RIP, IS-IS, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)
  - different "middleboxes" for different network layer functions: firewalls, load balancers, NAT boxes, ..
- ~2005: renewed interest in rethinking network control plane

4

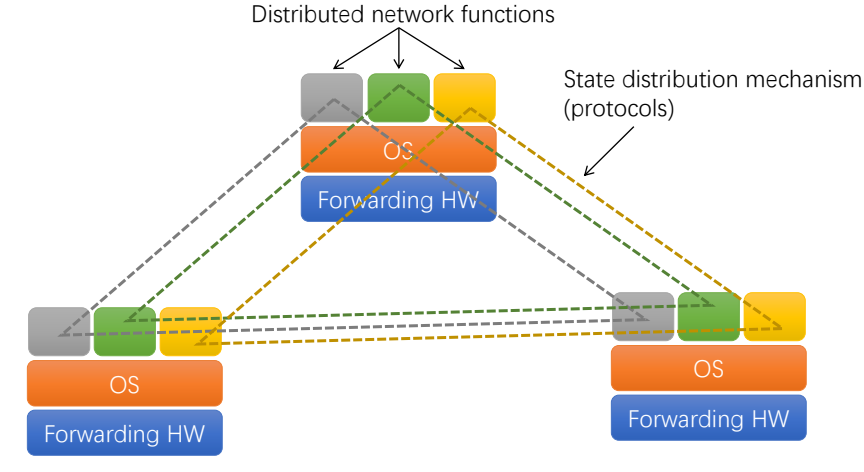
Recall: per-router control plane

- Individual routing algorithm components *in each and every router* interact forwarding tab



5

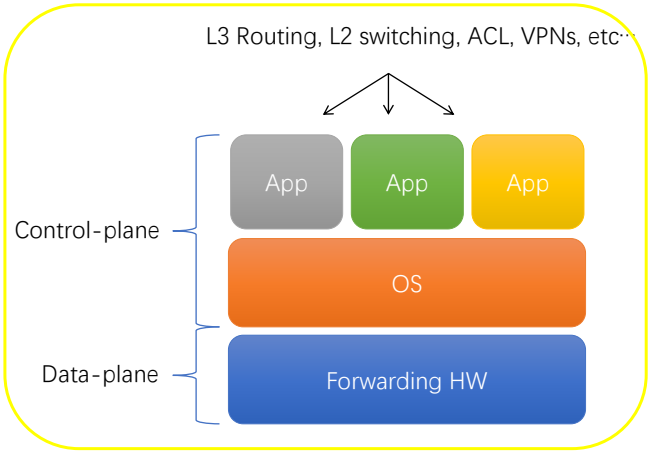
Classic network paradigm



6

6

Vertically integrated



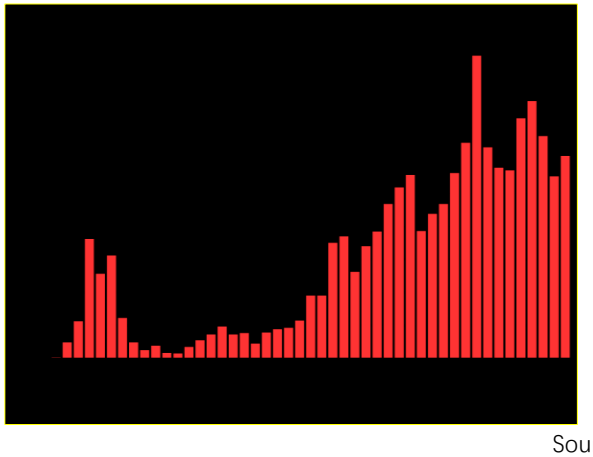
Closed platform!

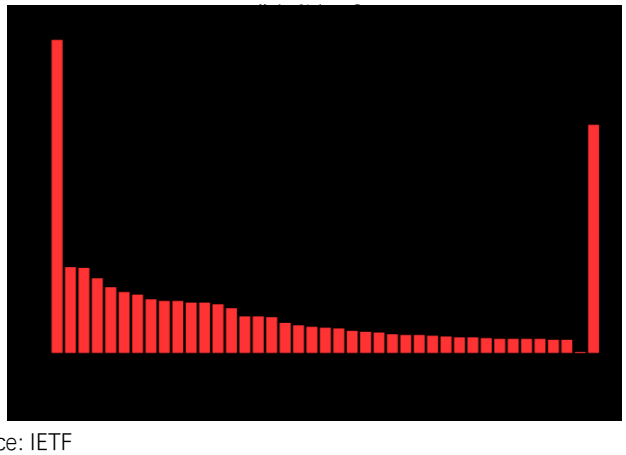
Protocols guarantee interoperability…
But what's the drawback?

7

7

Way too many standards, dominated by some vendors





Source: IETF

8

8

Traffic engineering

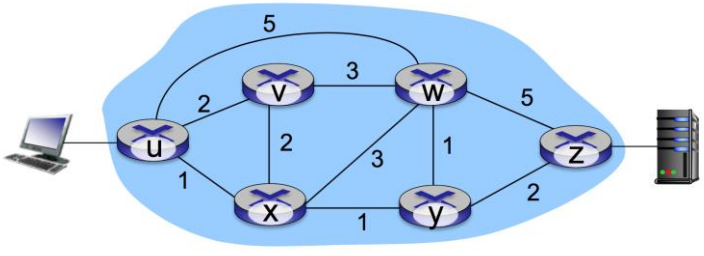
- Traffic Engineering (TE) is the process of distributing traffic flows through the network to achieve load balancing
- TE leads to:
  - Reduced congestion
  - **Improved bandwidth utilization**
- Approaches:
  - Preplanned:
    - OSPF + smart link weight setting
    - MPLS + optimal general routing
  - On demand
    - MPLS + Constraint-Based Routing

TE optimization objectives:
Different algorithms need to be developed for different optimization objectives

- *Minimizing congestion and pack losses in the network*
- *Improving link utilization*
- *Minimizing the total delay experienced by packets*
- *Increasing the number of customers*

9

Traffic engineering: traditional routing NOT ADEQUATE!



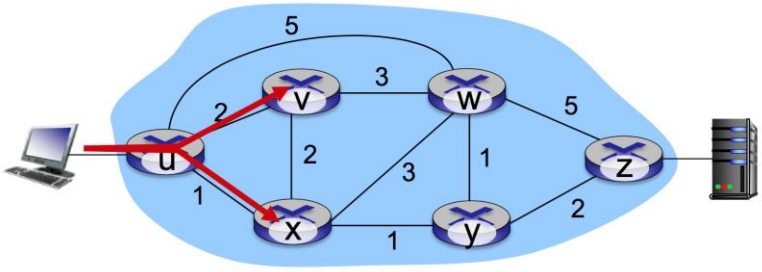
*Q: W*hat if network operator wants u-to-z traffic to flow along *uvwz*, x-to-z traffic to flow *xwyz*?

*A: N*eed to define link weights so traffic routing algorithm computes routes accordingly (or need a new routing algorithm)!

Link weights are only control "knobs": wrong!

10

Traffic engineering: difficult

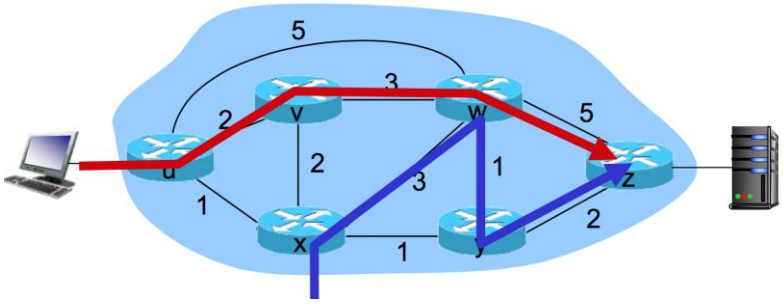


*Q: W*hat if network operator wants to split u-to-z traffic along uvwz *and* uxyz (load balancing)?

*A: C*an't do it (or need a new routing algorithm)

11

Traffic engineering: difficult



- *Q: W*hat if w wants to route blue and red traffic differently?
- *A: C*an't do it (with destination based forwarding, and LS, DV routing)

12

Non-standard management

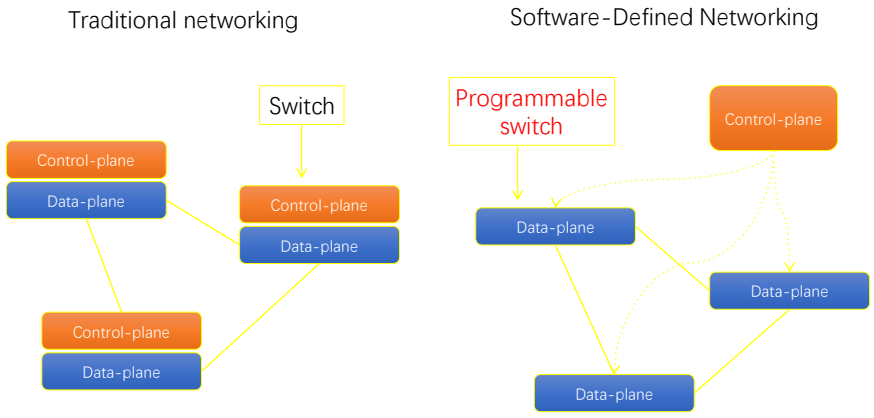
- **Configuration interfaces** vary across:
  - Different vendors
  - Different devices of same vendor *Manual, scripts!*
  - Different firmware versions of same device!
- **Simple Network Management Protocol** (***SNMP***) fail
  - Proliferation of non-standard MIBs
  - Partially implemented standard MIBS
  - IETF recently published a recommendation to stop producing writable MIB modules

MIB: Management Information Base

- Collections of various network objects that are operated with the use of a SNMP.
- The exact structure of the objects included in the management information base will depend on the configuration of the particular SNMP

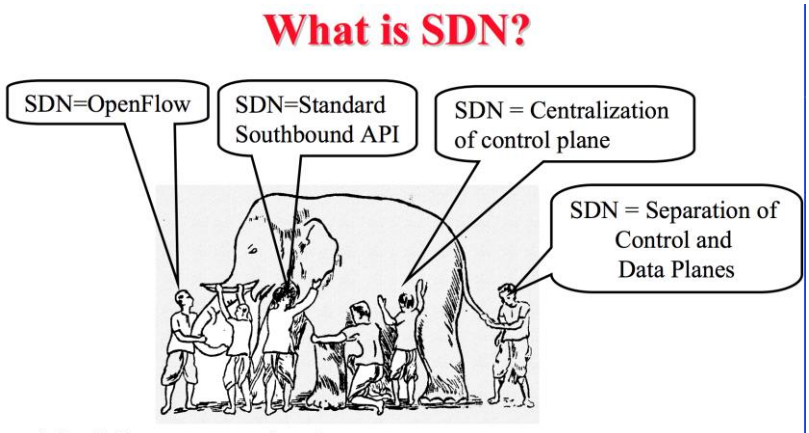
13

The (new) paradigm



14

SO what's really SDN?

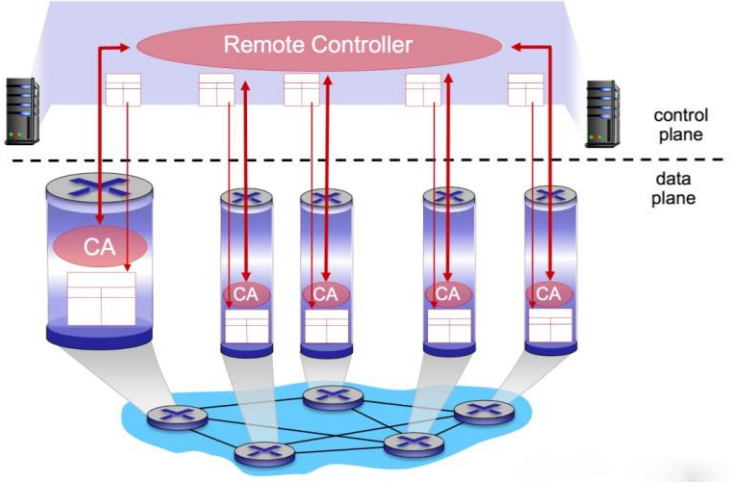


- All of these are mechanisms.
- SDN is *not* a mechanism.
- It is a framework to solve a set of problems ⇒ Many solutions

15

Recall: logically centralized control plane

- A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables



16

From protocols to API

- HW forwarding abstraction
  - Low-level primitives to describe packet forwarding
- Control plane API
  - Network topology abstraction
  - High-level access to switch programming
  - Common libraries
    - Host tracking
    - Shortest-path
    - Etc..

17

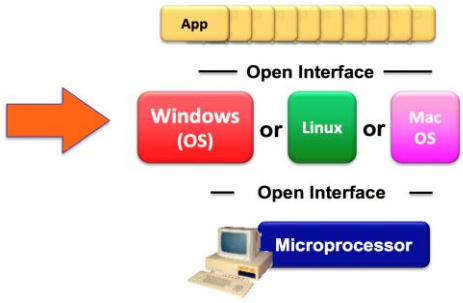
Software defined networking (SDN)

- *Why* a *logically centralized* control plane?
  - Easier network management:
    - Avoid router misconfigurations
    - Greater flexibility of traffic flows
  - Table-based forwarding ➔"programming" routers : OpenFlow API
    - *Centralized "programming"* easier: compute tables centrally and distribute
    - *Distributed "programming"* more difficult - compute tables as result of *distributed algorithm* (protocol) implemented in each and every router
- *Open (non-proprietary) implementation* of control plane : Open market for third-party network application developers
    - Network app store

But Be Careful:
Cisco Opendaylight &
China SDN Tragedy!

18

Analogy: mainframe to PC evolution



Vertically integrated
Closed, proprietary
Slow innovation
Small industry

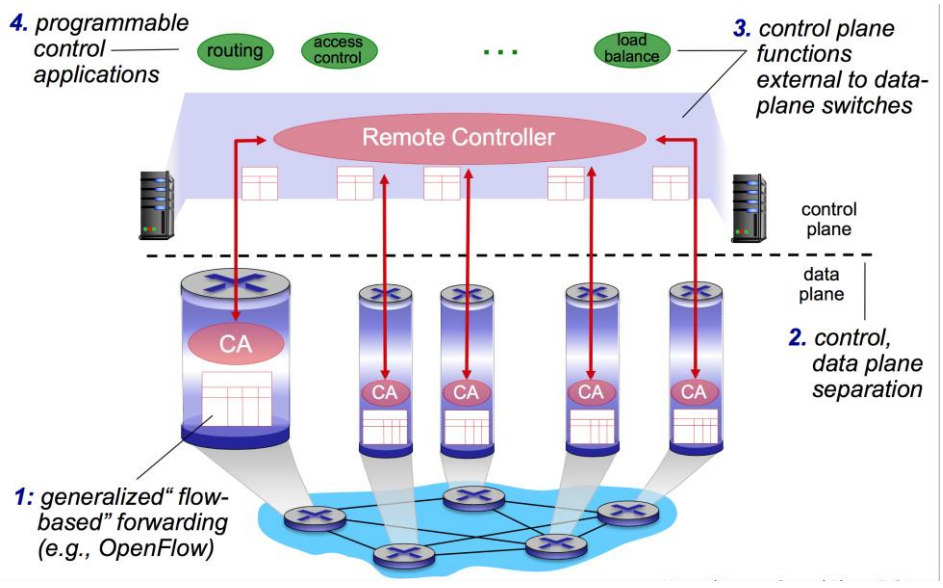
Horizontal
Open interfaces
Rapid innovation
Huge industry

Success keys

- Low-level HW open interface
- Good, extensible and possibly open-source Network OS
- Open market for third-party network application developers
  - Network app store
- **Several attempts** (Active Networks, IETF ForCES), but one winner …

19

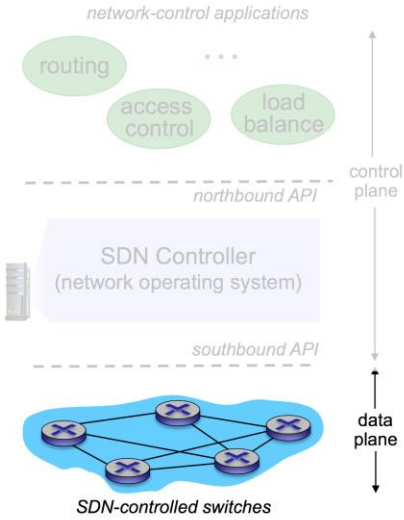
Software defined networking (SDN)



20

SDN perspective - data plane switches

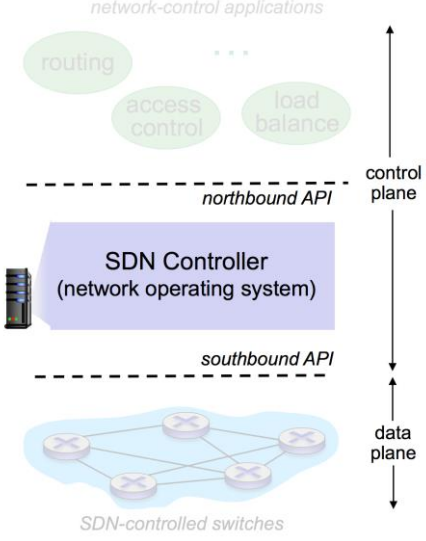
- Fast, simple, *commodity* switches implementing generalized data-plane forwarding in hardware
- Switch flow table computed, installed by controller
- API for table-based switch control (e.g., OpenFlow)
  - Defines what is *controllable* and what is not
- Protocol for communicating with controller (e.g., OpenFlow)



21

SDN perspective - SDN controller (NOS)

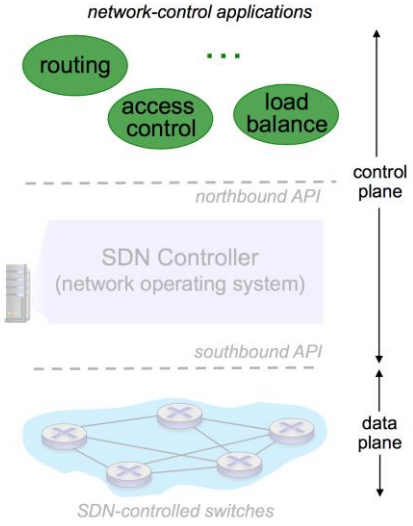
- A *logical entity for centralized* configuration and management
  - Interacts with network control applications "above" via northbound API
    - receives commands/instructions from the application layer and transmits them to the networking components
    - with an abstract view of the network, including various activities/events
  - Interacts with network switches "below" via southbound API
  - Maintain *network state* information
- *Implemented as distributed system* for performance, scalability, fault-tolerance, robustness



22

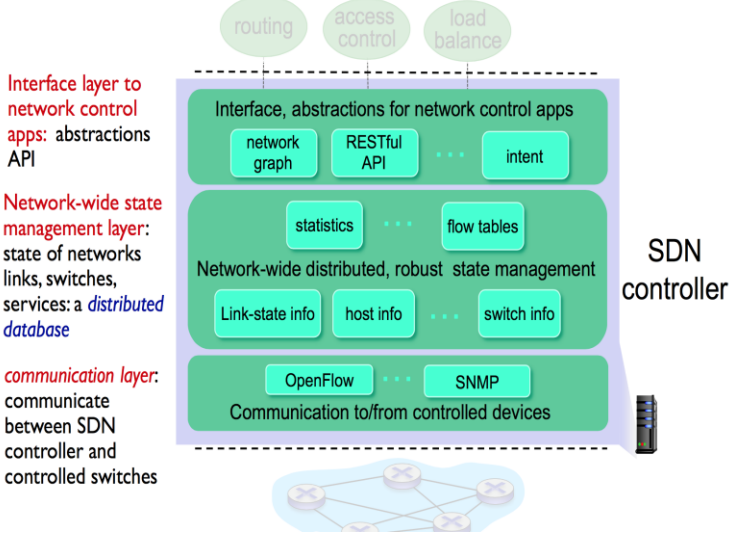
SDN perspective - Network control applications

- "Brains" of control: implement control functions using lower-level services, API provided by SND controller
- *Unbundled:* can be provided by 3rd party: distinct from routing vendor, or SDN controller



23

Components of SDN controller

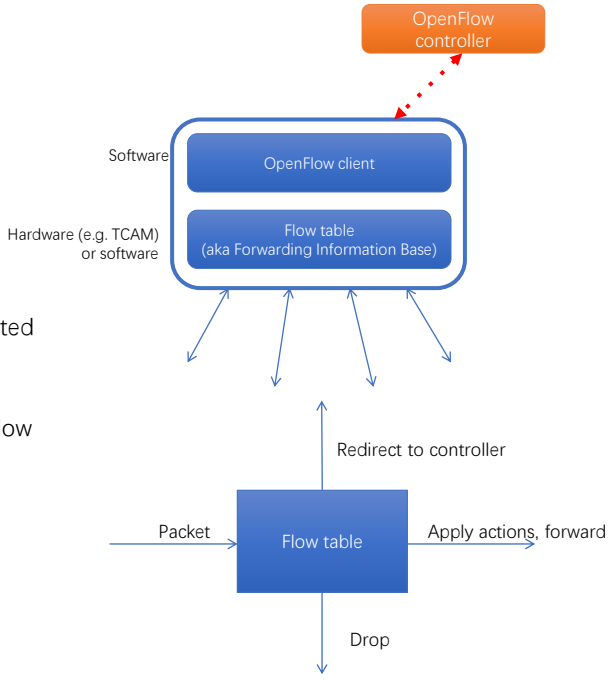


- A **RESTful API** is based on representational *state* transfer (REST) technology, an architectural style and approach to communications often used in web services development.
  ➔ *Lightweight , maintainable, and scalable*
- A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data.
  - Restful Web Service, expose API from applications in a *secure, uniform, stateless* manner to the calling client.
  - The calling client can perform predefined operations using the Restful service.
- **Restful Architecture**
  - **State and functionality are divided into distributed resources** – This means that every resource should be accessible via the normal HTTP commands of GET, POST, PUT, or DELETE
  - **The architecture is client/server, stateless, layered, and supports caching**

24

What is OpenFlow

- Switch abstraction
  - Match/action **flow table**
  - Flow counters
  - It doesn't describe how this should be implemented in switches (**vendor neutral !!!**)
- Application layer protocol
  - Binary wire protocol, messages to program the flow table
- Transport protocol
  - TCP, TLS



25

Example

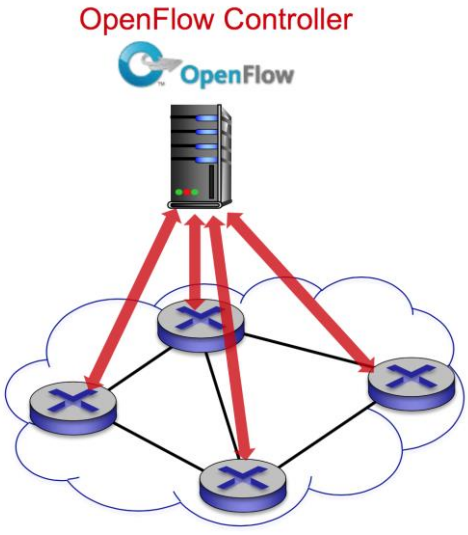
Description	Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dest	TCP sport	TCP dport	Action
L2 switching	*	*	00:1f:..	*	*	*	*	*	*	Port6
L3 routing	*	*	*	*	*	*	5.6.*.*	*	*	Port6
Micro-flow handling	3	00:20..	00:1f..	0x800	Vlan1	1.2.3.4	5.6.7.8	4	17264	Port4
Firewall	*	*	*	*	*	*	*	*	22	Drop
VLAN switching	*	*	00:1f..	*	Vlan1	*	*	*	*	Port6, port7, port8

26

26

OpenFlow protocol

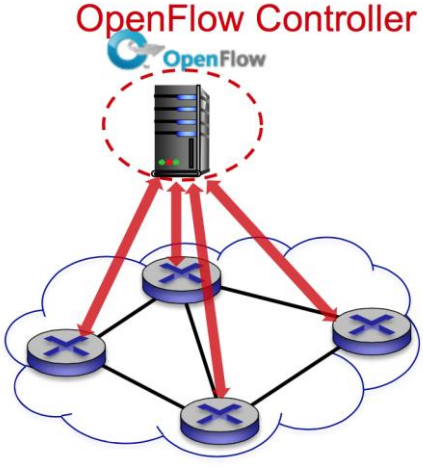
- Operates between controller, switch
- TCP used to exchange messages
  - optional encryption
- Three classes of OpenFlow messages:
  - controller-to-switch
  - asynchronous (switch to controller)
  - symmetric (misc)



27

OpenFlow: controller-to-switch messages

- Key *controller-to-switch* messages
  - *Features:* Controller queries switch features, switch replies
  - *Configure:* Controller queries/sets switch configuration parameters
  - *Modify-state:* Add, delete, modify flow entries in the OpenFlow tables
  - *Packet-out:* Controller can send this packet-out msg of specific switch port(s)

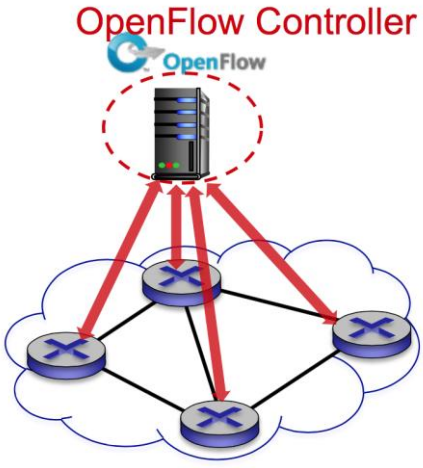


28

OpenFlow: switch-to-controller messages

- Key *switch-to-controller* messages
  - *Packet-in:* transfer packet (and its control) to controller.

    ➔ Packet-out message from controller
  - *Flow-removed:* flow table entry deleted at switch
  - *Port status:* inform controller of a change on a port.



Fortunately, network operators don't "program" switches by creating/sending OpenFlow messages directly.

- Instead use higher-level abstraction at controller

29

Evolution of Openflow

OF 1.0, ~2008, Dec 2009

The match/action model can ideally be used to program any network behavior and to get rid of protocol limitations at any level

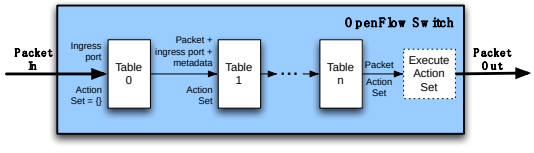
But unfortunately, with OF:

- Matches can be done only on a set of predefined header fields (Ethernet, IPv4, MPLS, VLAN tag, etc.)
- Actions are limited to a rather small set
- Header manipulation (like adding label/tags, rewriting of fields, etc.) is limited to standard schemes

As a result, OF is not really protocol independent and standards (including OF standards) are still necessary

OF 1.1, Feb 2011

Single tables are costly: all possible combinations of header values in a single long table
Solution: **Multiple Match Tables (MMT)**



New actions:

Add metadata: parameters added and passed to next table
Goto table: possibility to go to specific tables for further processing

Fast failover

30

Evolution of Openflow

OF 1.2, Feb 2012

Support for IPv6, new match fields:
- source address, destination address, protocol number, traffic class, ICMPv6 type, ICMPv6 code, IPv6 neighbor discovery header fields, and IPv6 flow labels

Extensible match (Type Length Value)
Experimenter extensions
Full VLAN and MPLS support
Multiple controllers

OF 1.3, June 2012

Initial traffic shaping and QoS support
- **Meters:** tables (accessed as usual with "goto table") for collecting statistics on traffic flows and applying rate-limiters

More extensible wire protocol
Synchronized tables
- tables with synchronized flow entries

Bundles
- similar to transactional updates in DB

Support for optical ports

OF 1.5/1.6: 2015 …

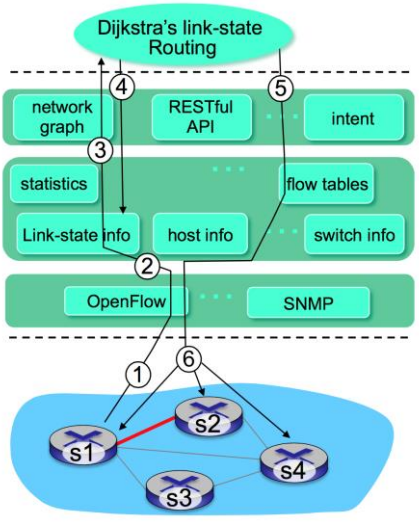
Future:?

Within ONF
Tunnel support
L4-L7 service support
Error handling
Fitness for carrier use
- Support for OAM in its various forms

Flow state (…)

31

SDN: control/data plane interaction example



① S1, experiencing link failure using OpenFlow port status message to notify controller

② SDN controller receives OpenFlow message, updates link status info

③ Dijkstra's routing algorithm application has previously registered to be called when ever link status changes. It is called.

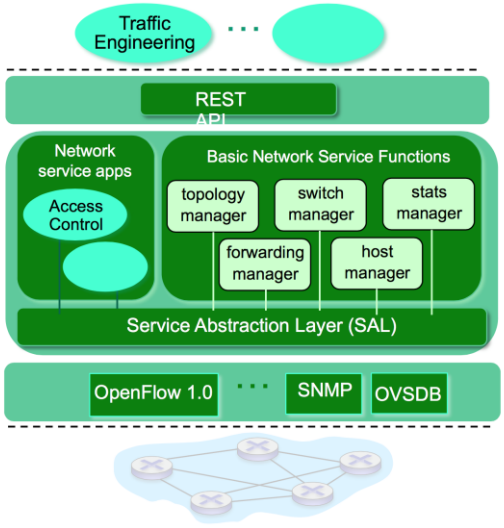
④ Dijkstra's routing algorithm access network graph info, link state info in controller, computes new routes

⑤ link state routing app interacts with flow-table-computation component in SDN controller, which computes new flow tables needed

⑥ Controller uses OpenFlow to install new tables in switches that need updating

32

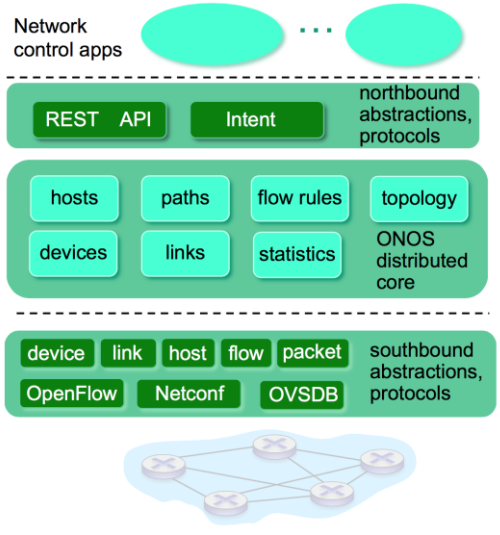
OpenDaylight (ODL) controller



- ODL Lithium controller
- Network apps may be contained within, or be external to SDN controller
- Service Abstraction Layer: interconnects internal, external applications and services

33

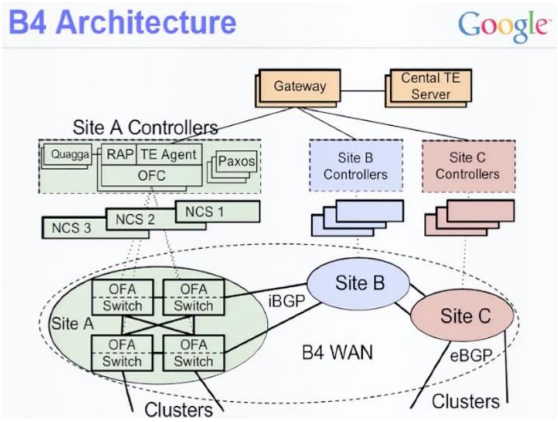
ONOS controller



- Control apps separate from controller
- Intent framework: high-level specification of service: what rather than how
- Considerable emphasis on distributed core: service reliability, replication performance scaling

34

SDN at google

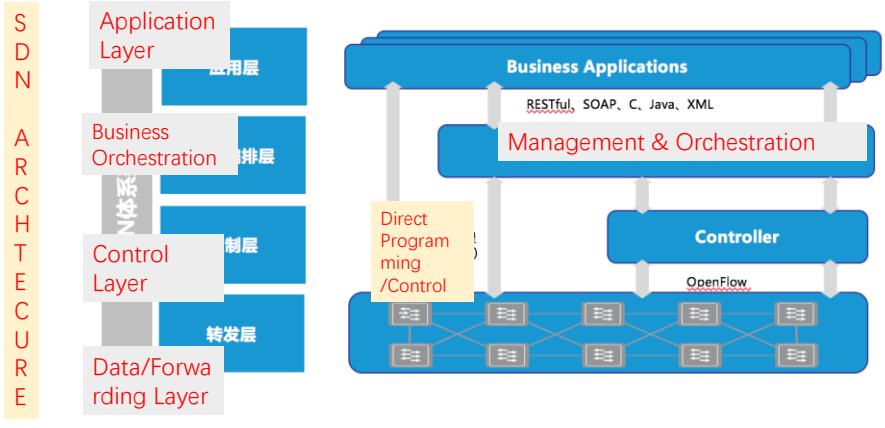


Interconect 11 DCs with SDN WAN
Started in 2010, completed in 2012!

- *Bandwidth utilization for the SDN-WAN ➔ 100%*
- *Fault recovery convergence time ➔ 9s to 1s*

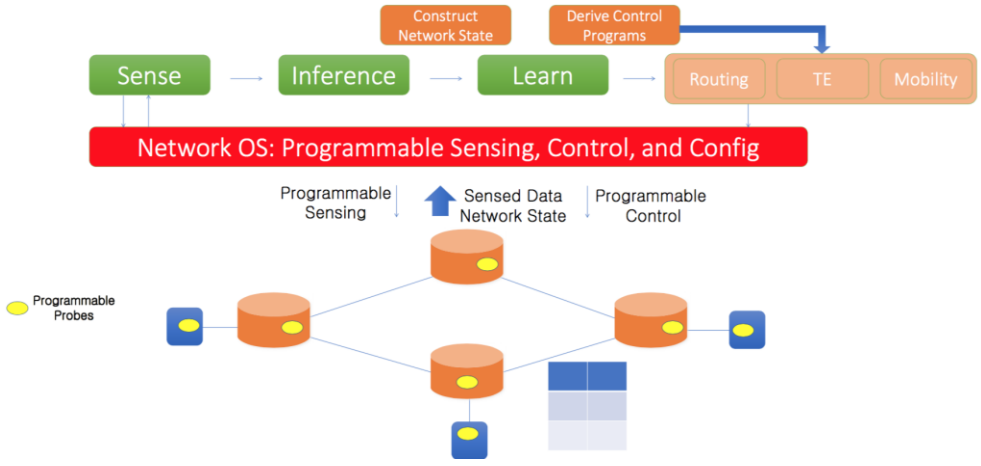
35

How do we leverage SDN in business apps



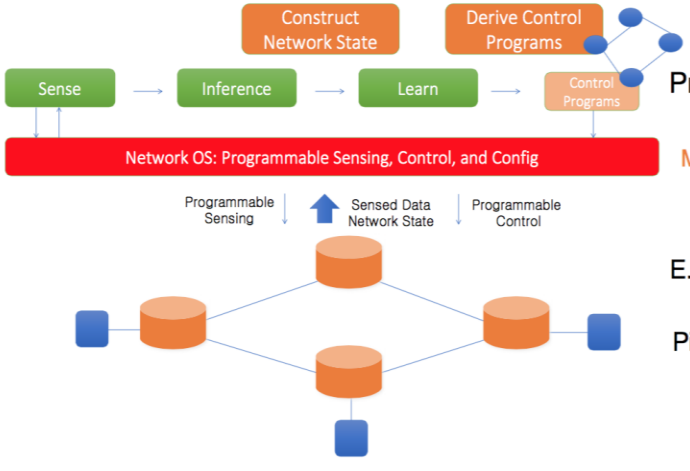
36

Intelligent networking



37

Intelligent networking



Hypothesis

Programmable sensing makes reconstruction easier & faster

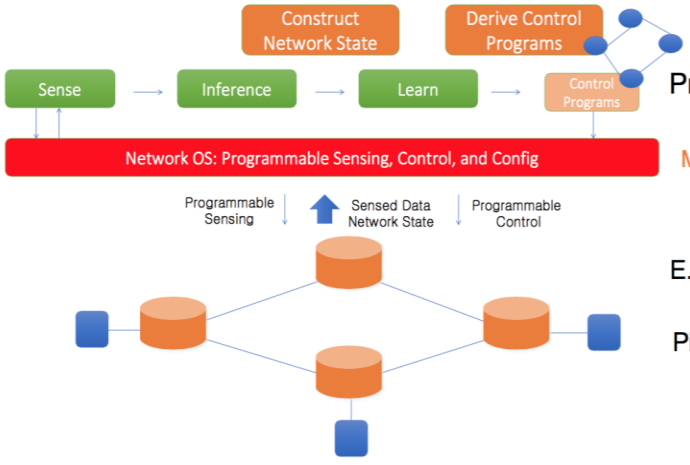
More/customized data helps ML algorithms to be more effective

E.g. ping data + queue statistics is better than just ping data

Ping data + queue statistics per network slice is much better than just ping data

38

Intelligent networking



Hypothesis

Programmable sensing makes reconstruction easier & faster

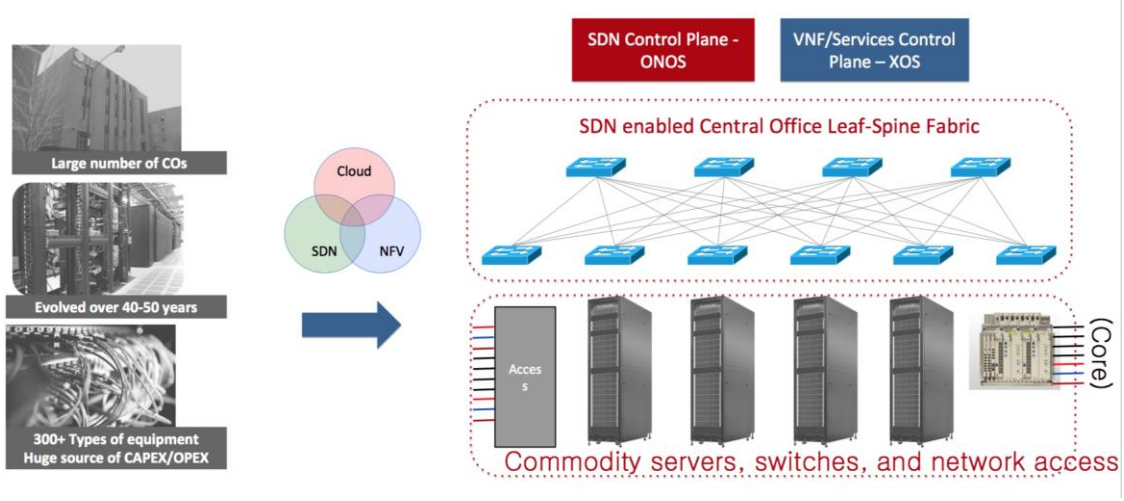
More/customized data helps ML algorithms to be more effective

E.g. ping data + queue statistics is better than just ping data

Ping data + queue statistics per network slice is much better than just ping data

39

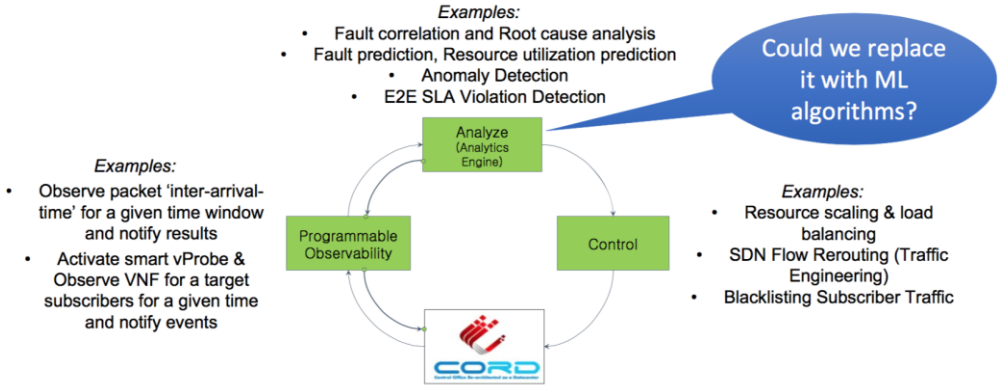
CORD (Central Office Re-architectured as Data center)



40

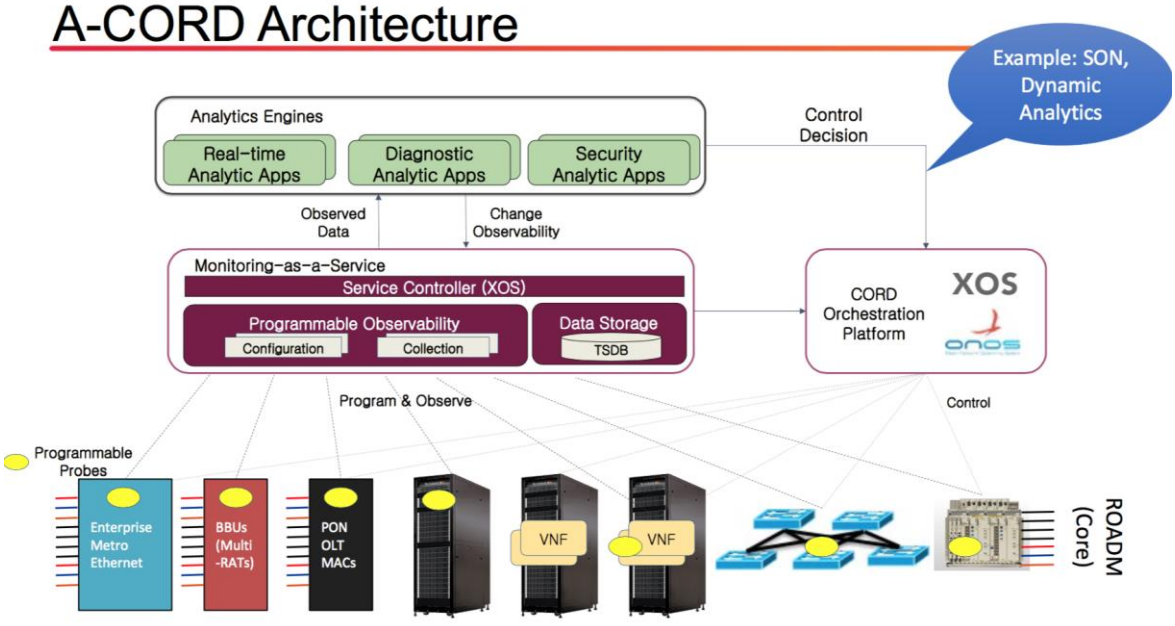
CORD's Goal

- Enable programmable observability & closed loop control based on analytics



41

A-CORD Architecture

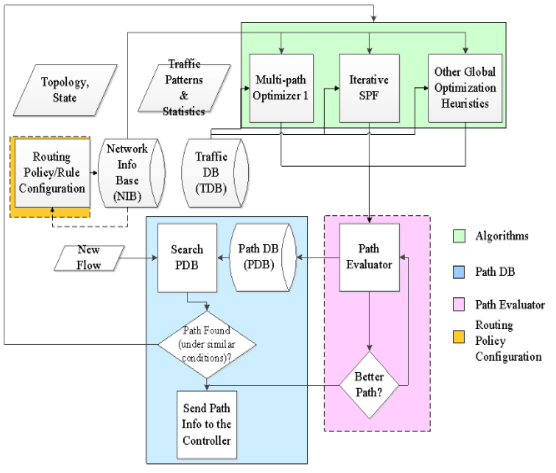


42

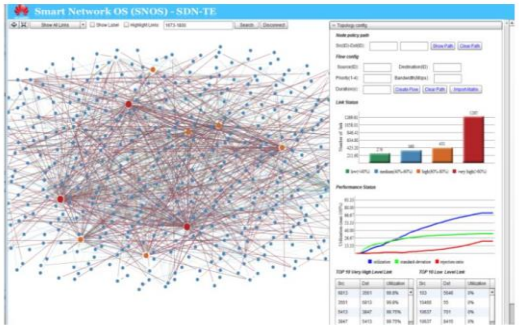
SDN in action – at Huawei (2012 – 2016)

Controllers SOX 1.x, 2.x
OpenFlow Switch SN640

Adaptive ***multi-path*** computation framework for centrally controlled networks



- Traffic Patterns & Stats
- Fast multi-path computation
- Path learning and adaptation
- Path Evaluation

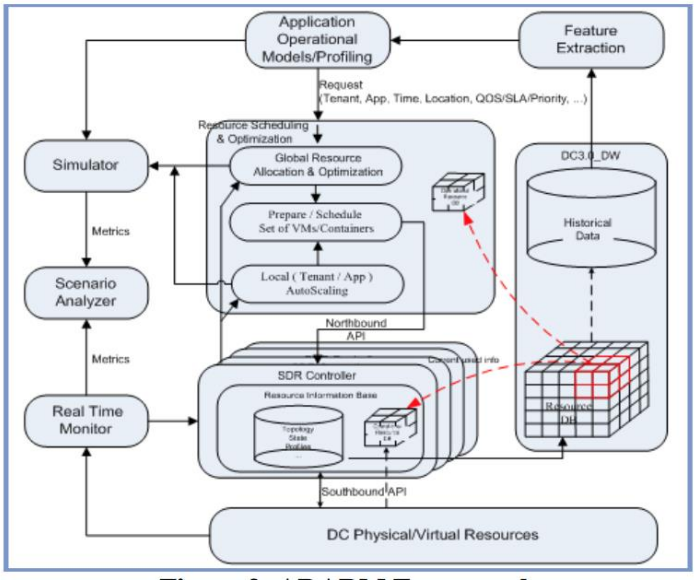


43

SDN in action – at Huawei (2012 – 2016)

Application-Driven Adaptive Resource Management Framework for Data Centers

- Resource as services
- Application & Operations profiling & stats
- Global resource allocation and optimization
- Autoscaling
- Scenario analyzer



44

Ian F. Akyildiz, Ahyoung Lee, Pu Wang, Min Luo, and Wu Chou, "*A Roadmap for Traffic Engineering in Software Defined Networks*", Journal of Computer Networks, Elsevier, October 2014 (electronic version available July 2014)

Status of SDN: Controller & Switch – Standards, capacity, performance,…
SDN for Traffic Engineering
Challenges

45

SDN: selected challenges

- Hardening the control plane: dependable, reliable, performance-scalable, secure distributed system
  - robustness to failures: leverage strong theory of reliable distributed system for control plane
  - dependability, security: "baked in" from day one?
- Networks, protocols meeting mission-specific requirements
  - e.g., real-time, ultra-reliable, ultra-secure
- Internet-scaling

46

Summary

- SDN controllers
  - Control vs data
  - Application – Control – data forwarding
  - Controller Components
  - SDN TE
  - SDN in Action: Google, Huawei,…

47

Assignments

- Reading: 5.5
- Problems: P21

48