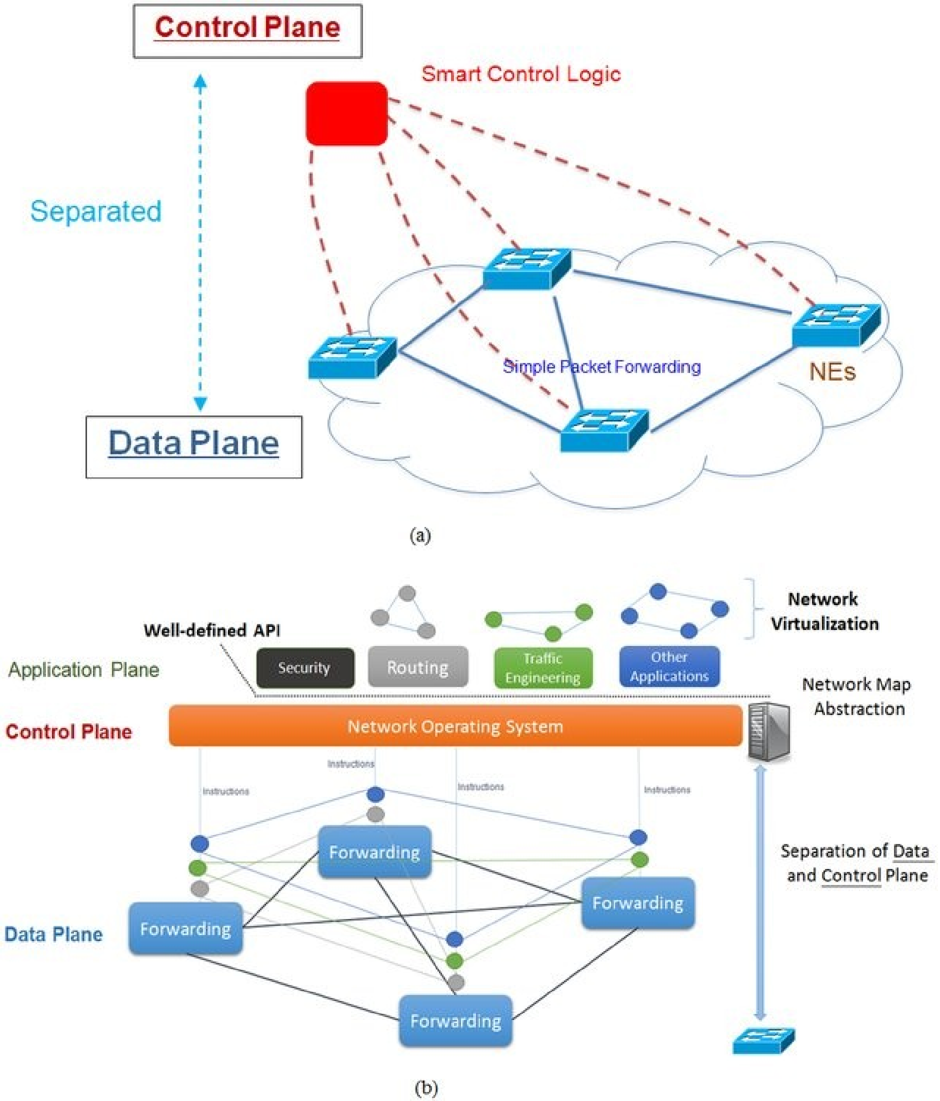
CSC7970 – NEXT-GENERATION NETWORKING

SOFTWARE DEFINED NETWOKRING

Instructor: Susmit Shannigrahi
sshannigrahi@tntech.edu

Control vs Data Plane



(a)

(b)

In traditional networks, they are in the same device.

Problems?

Difficult to change
Difficult to deploy new protocols
Many vendors and so on

https://www.researchgate.net/publication/309917884_An_adaptive_load_monitoring_solution_for_logically_centralized_SDN_controller/figures?lo=1

SDN

- What is Software-Defined Networking?
  - Networking paradigm where control and forwarding plane are decoupled
  - Network intelligence is (logically) centralized and physically separate from forwarding devices

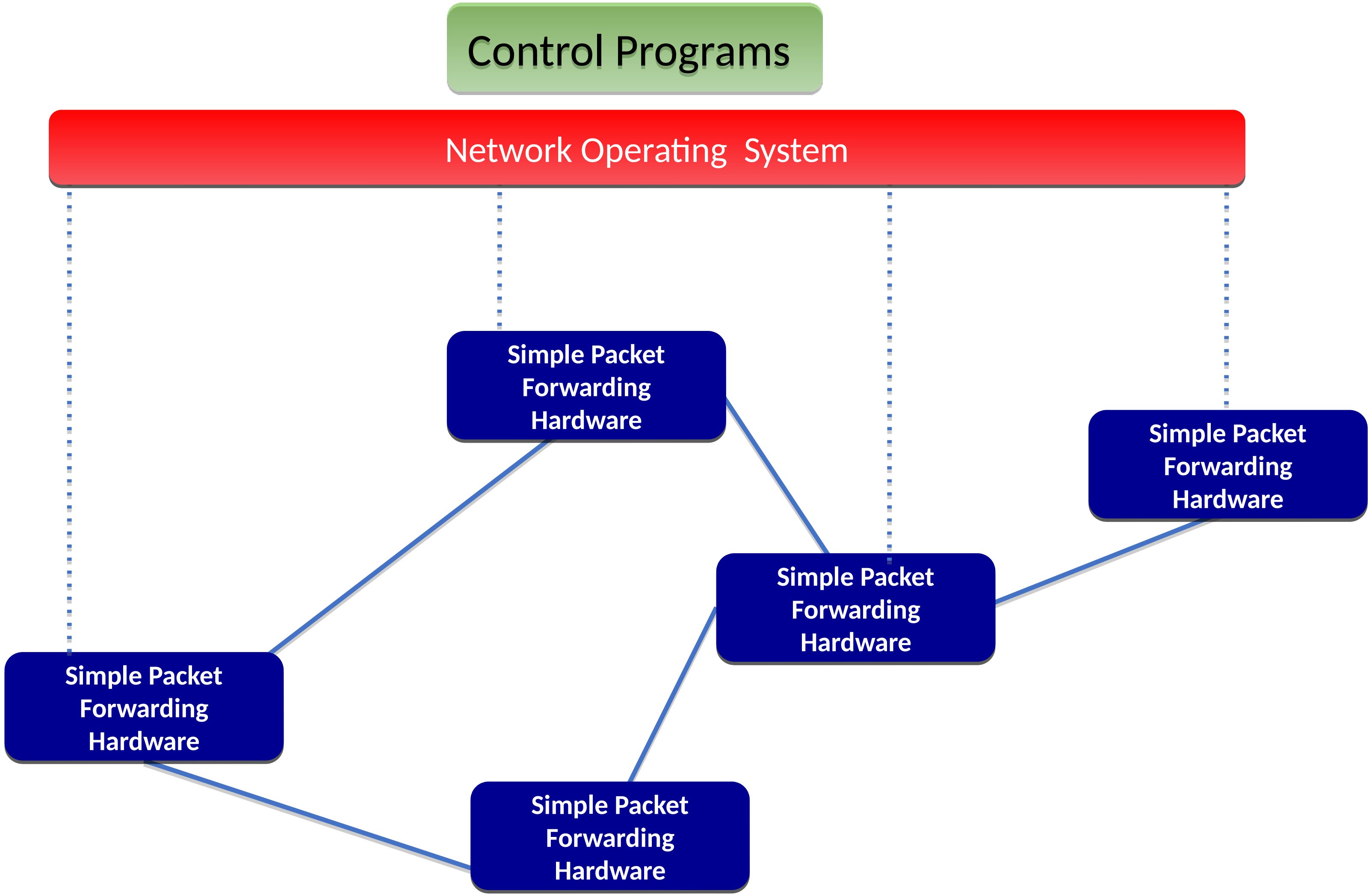
- Controller/forwarding device model
  - Controller is the network intelligence!
  - Forwarding device is “dumb”
  - Forwarding device asks the controller how to forward traffic

Why SDN?

- (Distributed) Networks are hard to manage/configure nowadays
  - Massive scale
  - Many different stakeholders
  - Many vendors with their own implementations

- Simplify network configuration and management
  - Centralize the management/configuration entity in a network
  - Create a network Operating System (OS)
  - Much higher uptime
  - Days vs years to deploy new protocols
  - Problem solved?

Idea: An OS for Networks

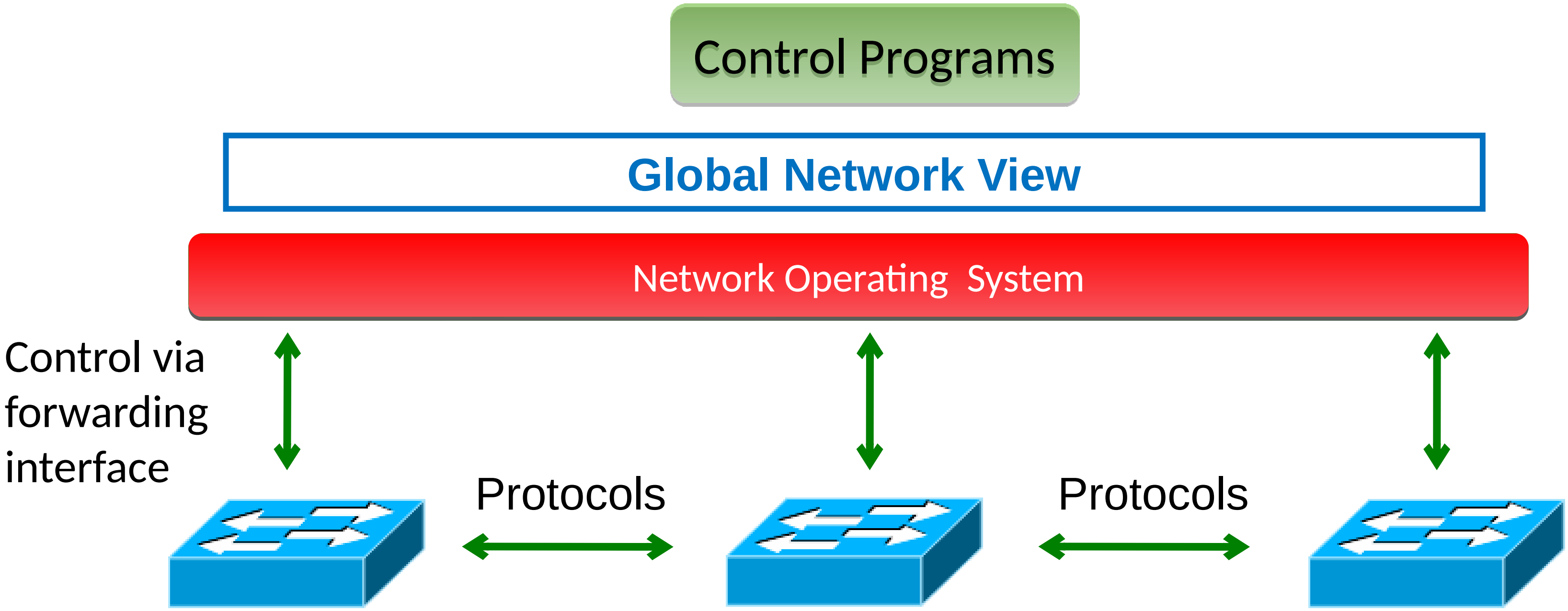

Control Programs
Network Operating System
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware
Simple Packet Forwarding Hardware

Centralized intelligence

- ***No more distributed protocols!***
  - Network intelligence is centralized
  - Easier to implement, write code for, debug, maintain..

- Network OS is the fundamental control block (the “abstraction”)
  - Global view of the network

Global Network View



The Future of Networking, and the Past of Protocols, Scott Shenker, *with Martin Casado, Teemu Koponen, Nick McKeown*

Does SDN solve TCP/IP’s problems?

- SDN is centralized!
- SDN works well for enterprise networks (a single data-center, etc.)
  - Small (to medium)-scale networks operated by a single stakeholder
  - Mostly stable, protected, well-managed environments

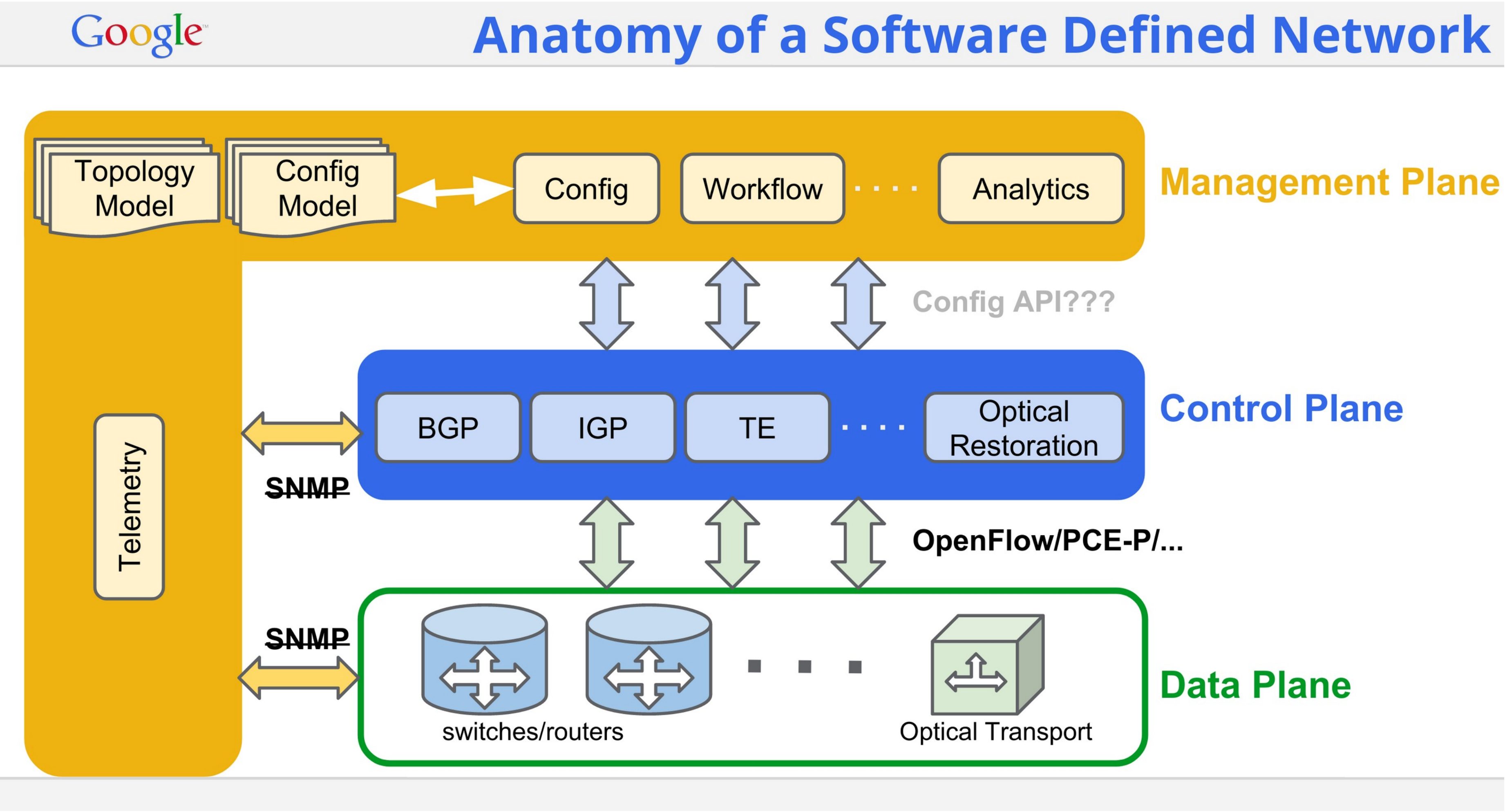
- Can it be deployed across the Internet?
- Does it work across networks of different stakeholders?
- What about fault tolerance? ← Important!

Fault tolerance tradeoffs

- Central controller → Bottleneck and Single point of failure
  - Can have multiple central controllers
    - But how do you synchronize?
  - Hierarchy of controllers
    - Single point of failure still exists

- Works well when you have control over your network
  - Google data centers

Speaking of Google



Google

Anatomy of a Software Defined Network

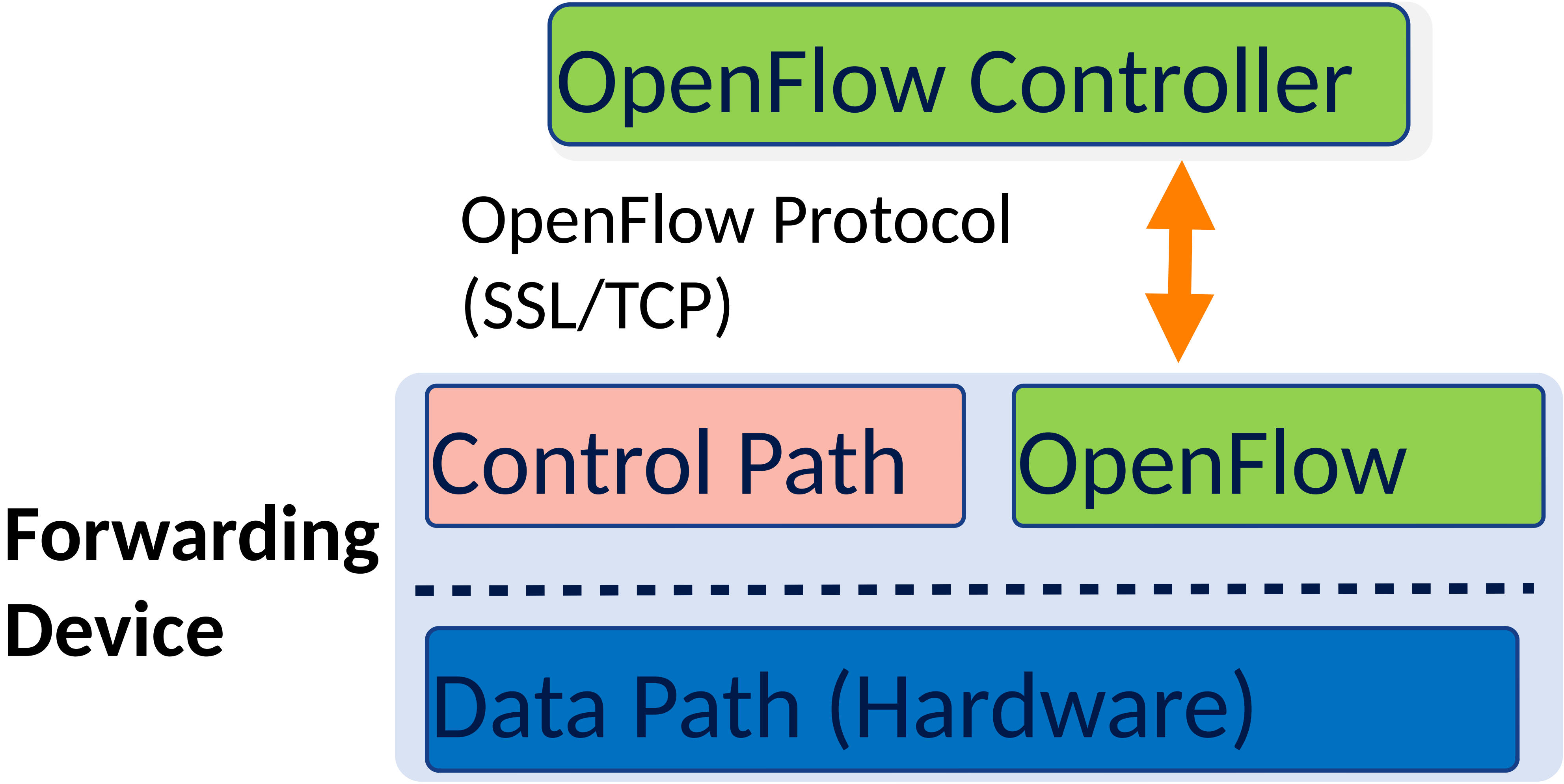
https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/

OpenFlow

- Part 1: Protocol for network controllers to communicate with forwarding devices and vice versa
  - Common language between controllers and devices

- Part 2: Devices ask the controller how to forward the traffic they receive
  - They categorize traffic into flows
  - They ask the controller how to forward a flow, then cache this decision for some amount of time

Architecture

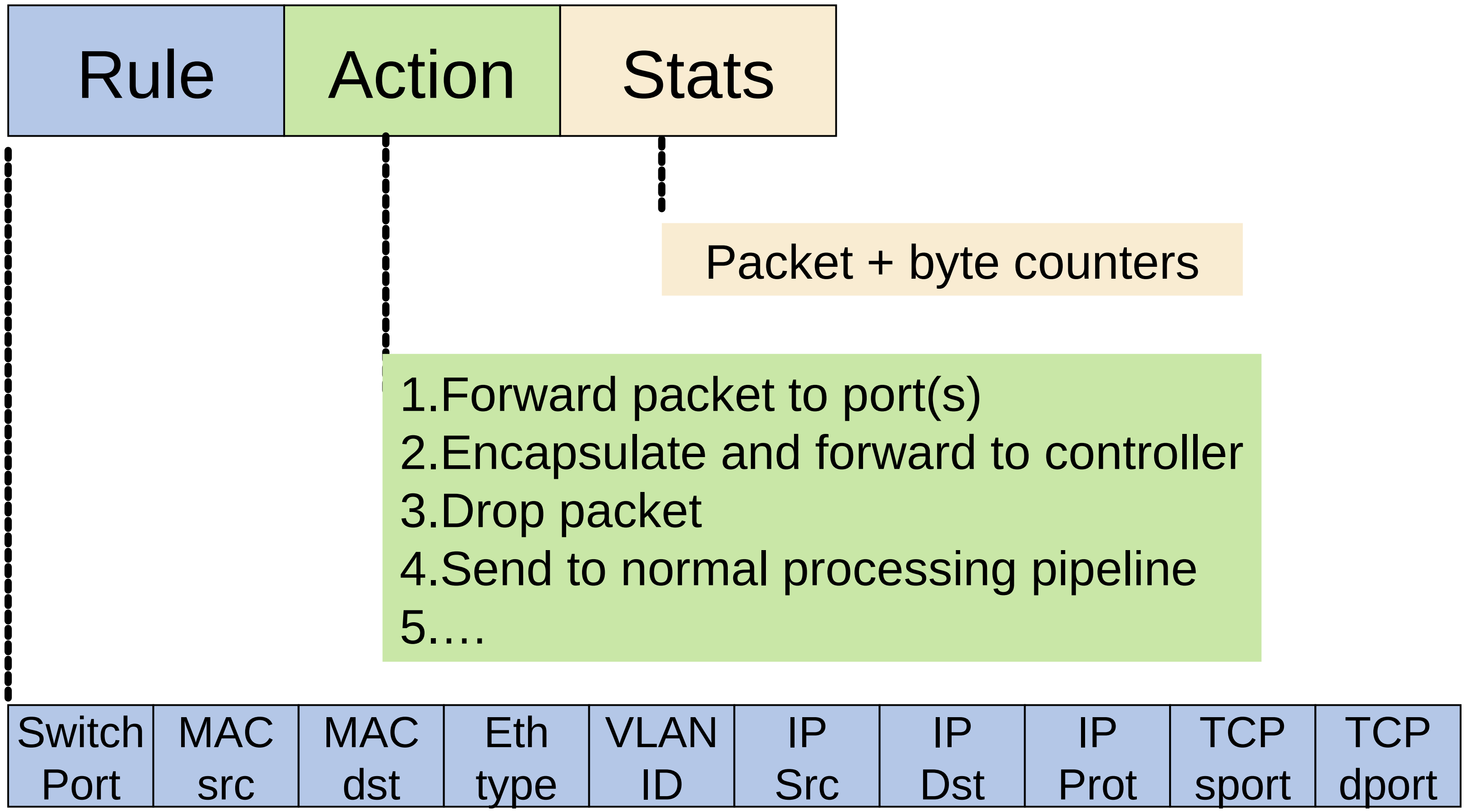


Flow table & traffic flows

- Each forwarding device (switches) maintains a flow table
  - Specifies the actions to take for each traffic flow
  - Override a “class” ← networking device capability
  - Use one of the many data structure available in modern devices to store additional information

- Creation of flows of packets is based on L2-L4 header fields
  - Combinations of MAC src, MAC dst, IP src, IP dst, TCP sport, TCP dport, etc.

Flow table entry



Rule	Action	Stats

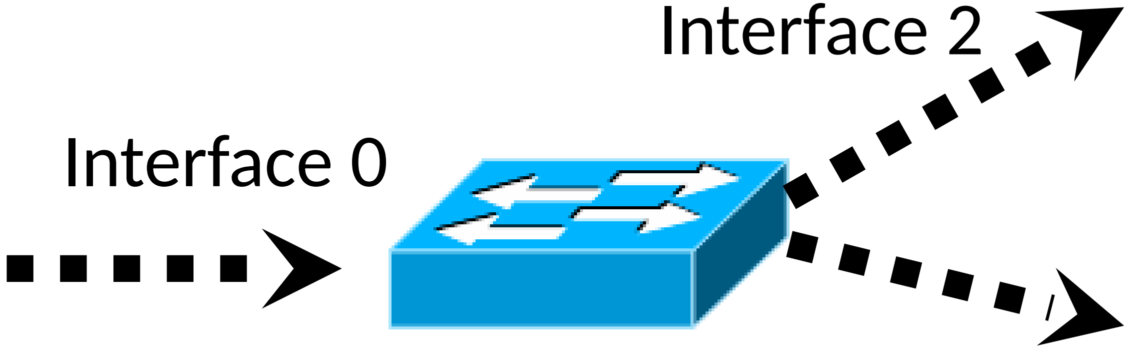
Packet + byte counters

1. Forward packet to port(s)
2. Encapsulate and forward to controller
3. Drop packet
4. Send to normal processing pipeline
5. …

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport

Example of operation

Controller



Interface 2

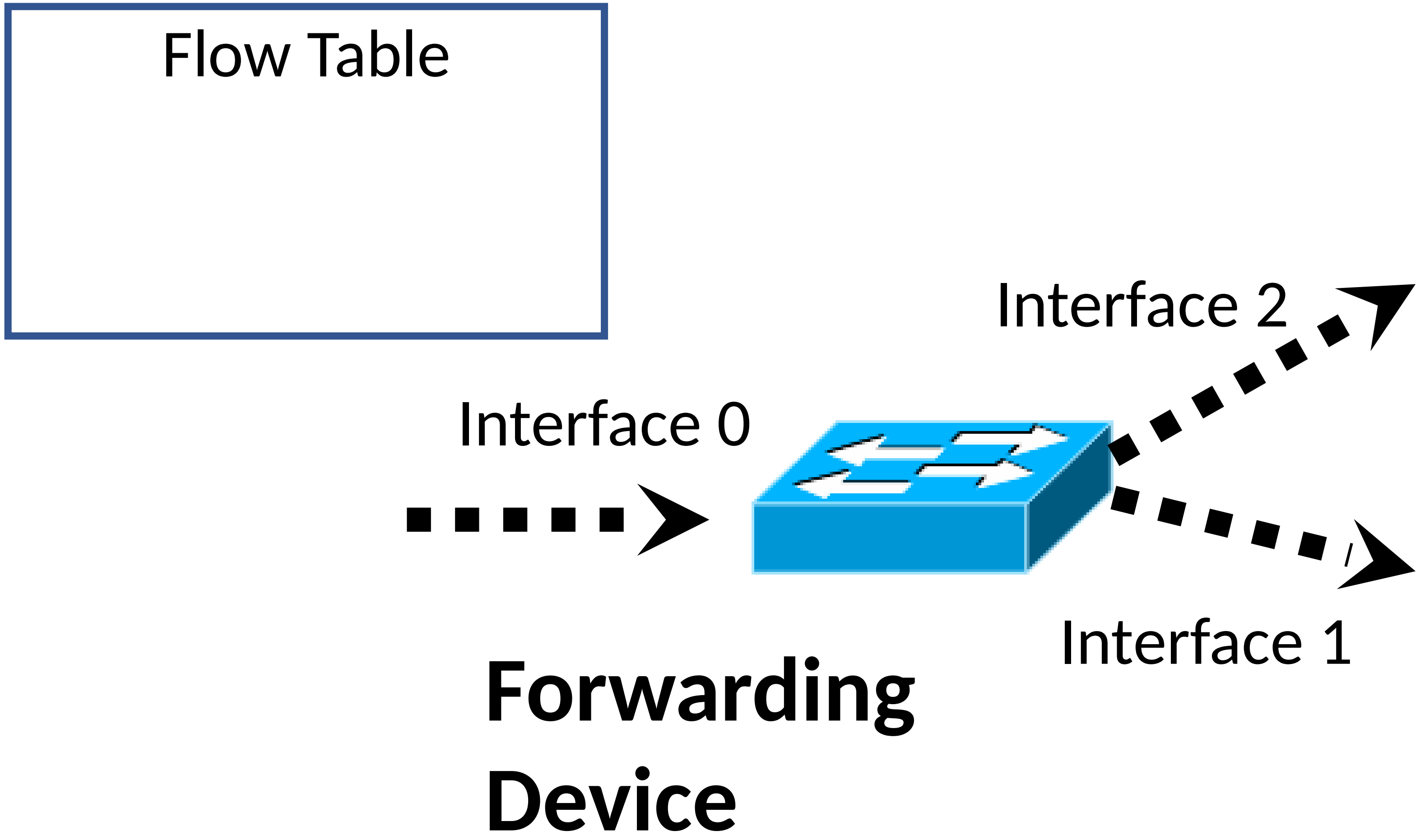
Interface 0

Interface 1

Forwarding Device

Example of operation

Controller



Flow Table

Interface 0

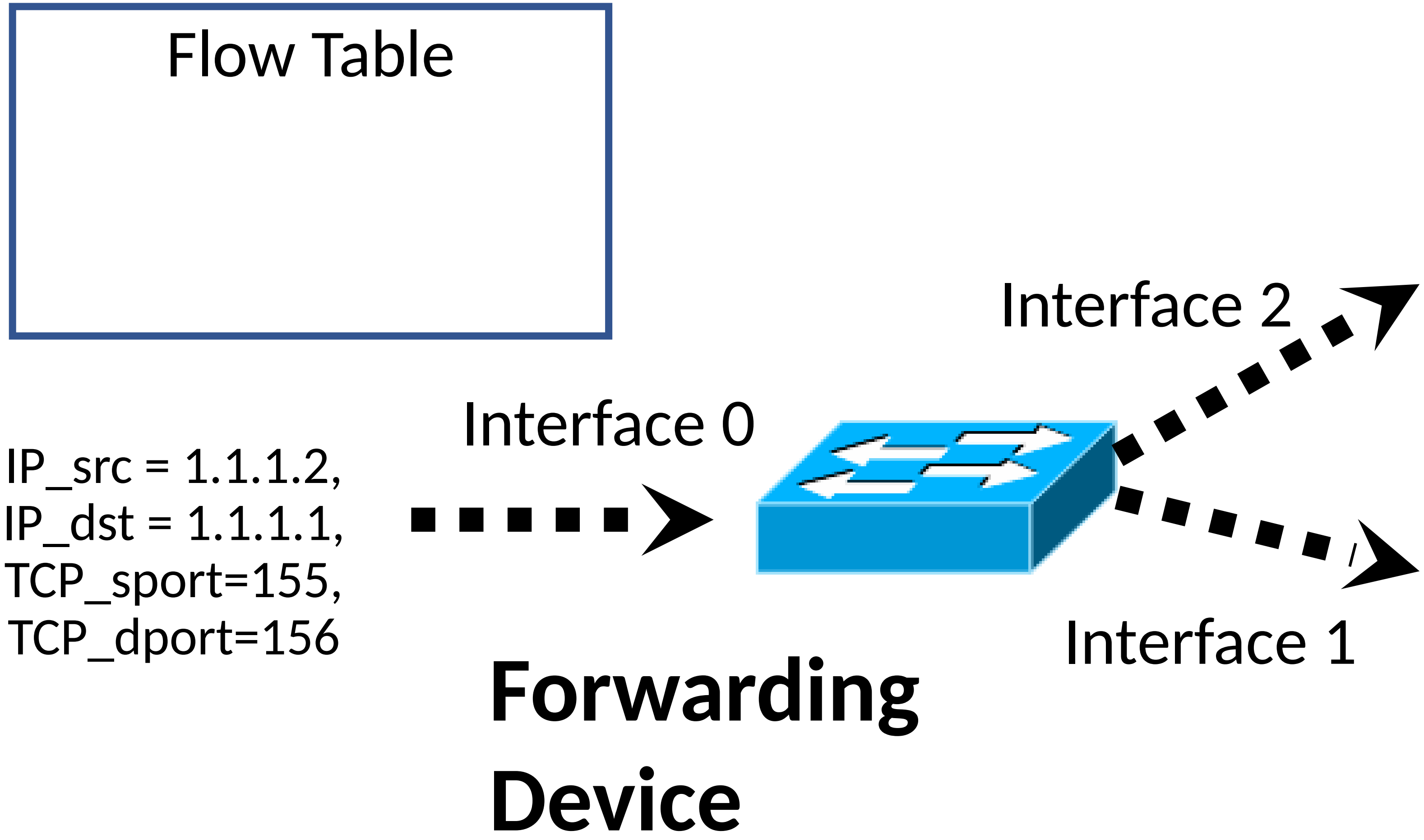
Interface 2

Interface 1

Forwarding Device

Example of operation

Controller



Flow Table

IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=156

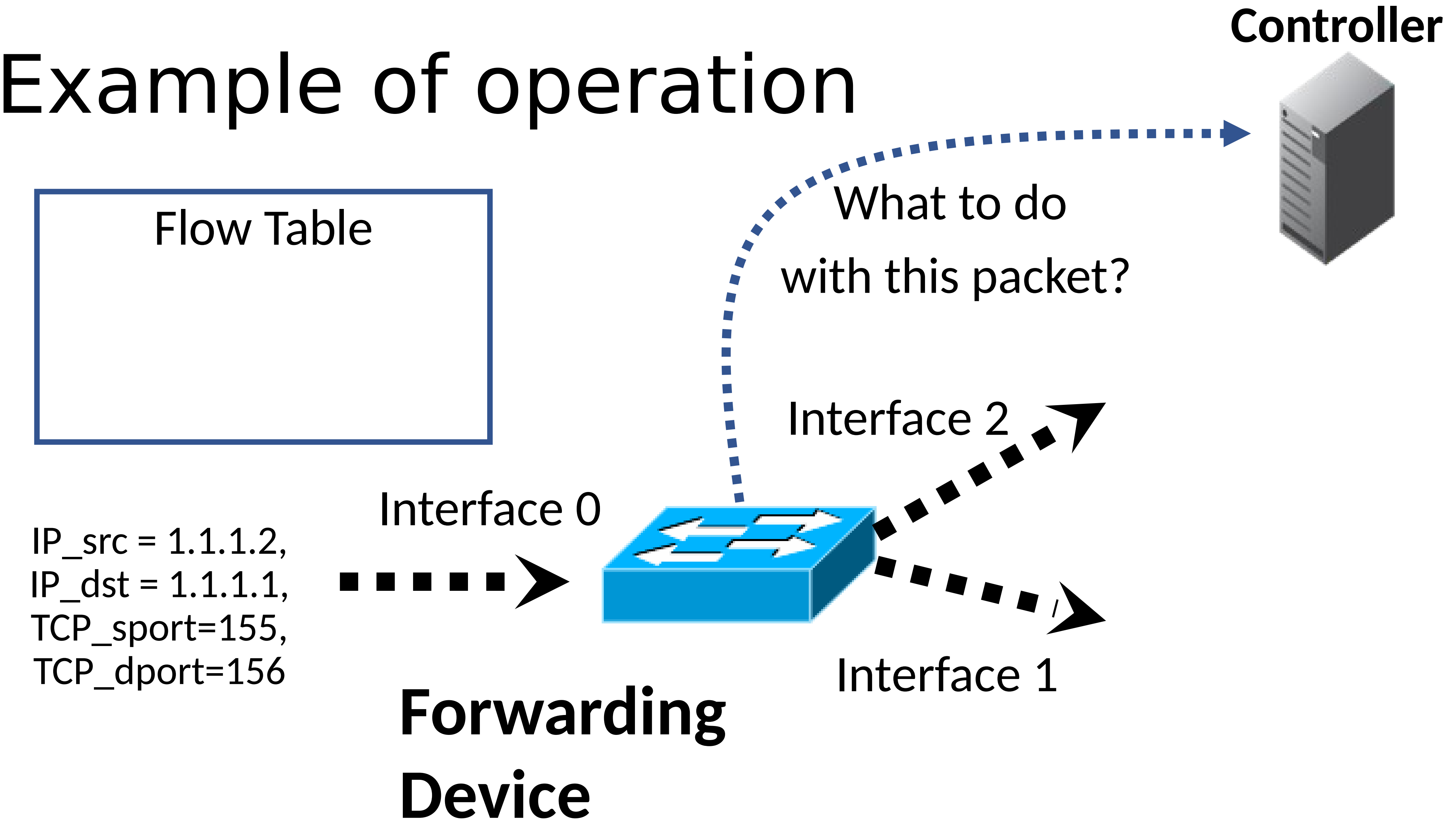
Interface 0

Interface 2

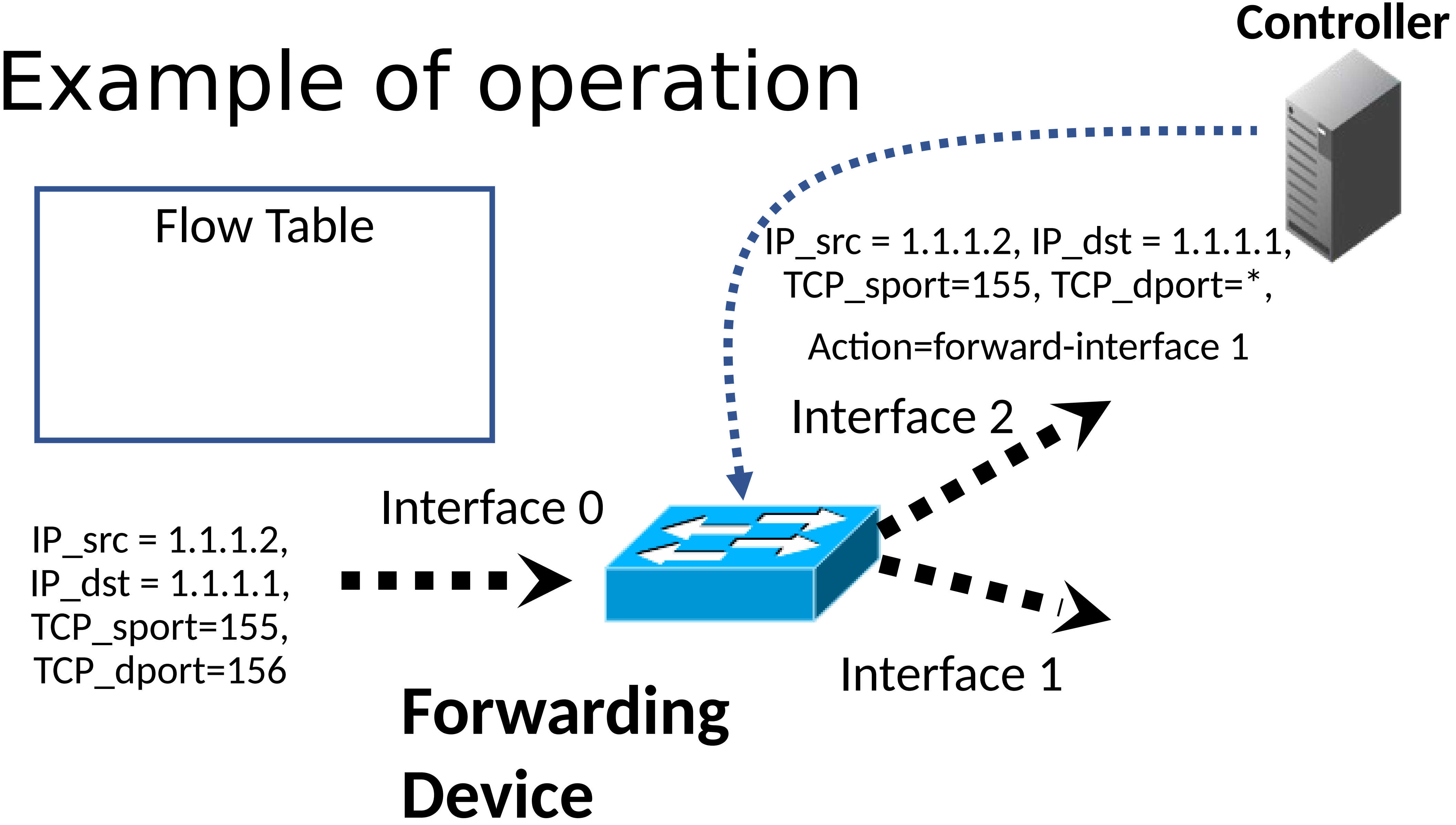
Interface 1

Forwarding Device

Example of operation



Example of operation



Flow Table

Controller

IP_src = 1.1.1.2, IP_dst = 1.1.1.1,
TCP_sport=155, TCP_dport=*,

Action=forward-interface 1

Interface 2

Interface 0

IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=156

Interface 1

Forwarding Device

Example of operation

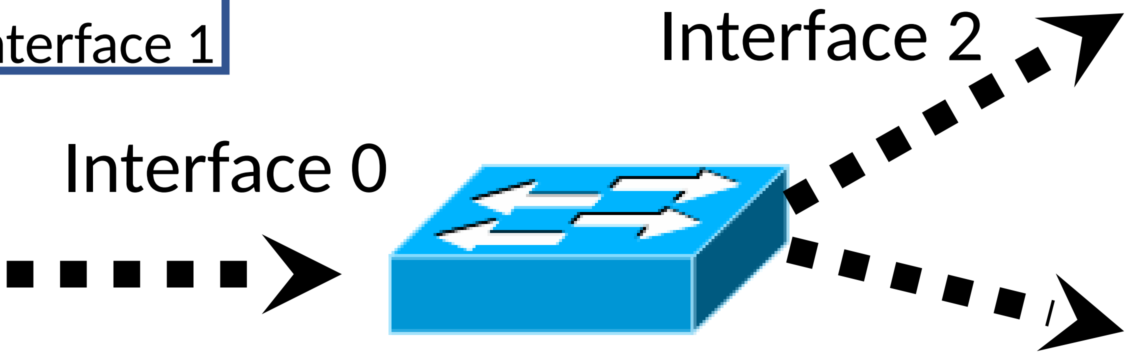
Controller

Flow Table

IP_src = 1.1.1.2, IP_dst = 1.1.1.1, TCP_sport=155, TCP_dport=*,

Action=forward-interface 1

IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=156



Forwarding Device

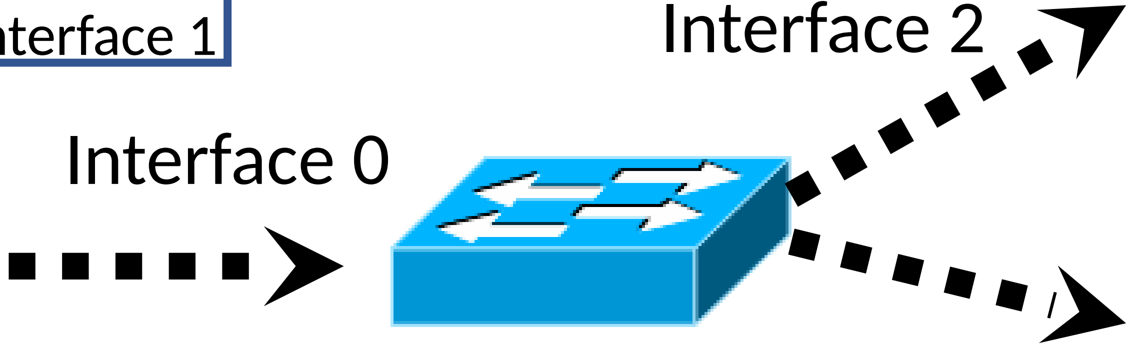
Example of operation

Flow Table

IP_src = 1.1.1.2, IP_dst = 1.1.1.1, TCP_sport=155, TCP_dport=*,

Action=forward-interface 1

Controller



IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=156

Forwarding Device

Example of operation

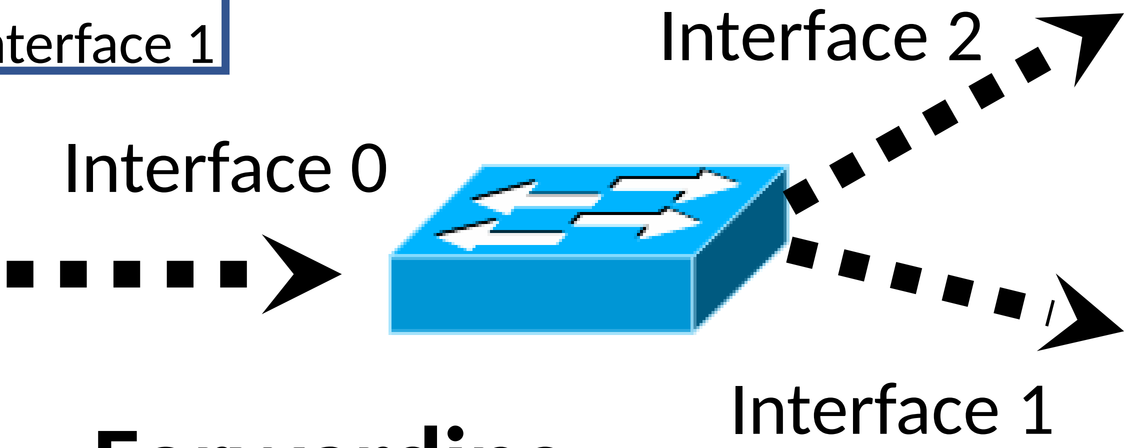
Controller

Flow Table

IP_src = 1.1.1.2, IP_dst = 1.1.1.1, TCP_sport=155, TCP_dport=*,

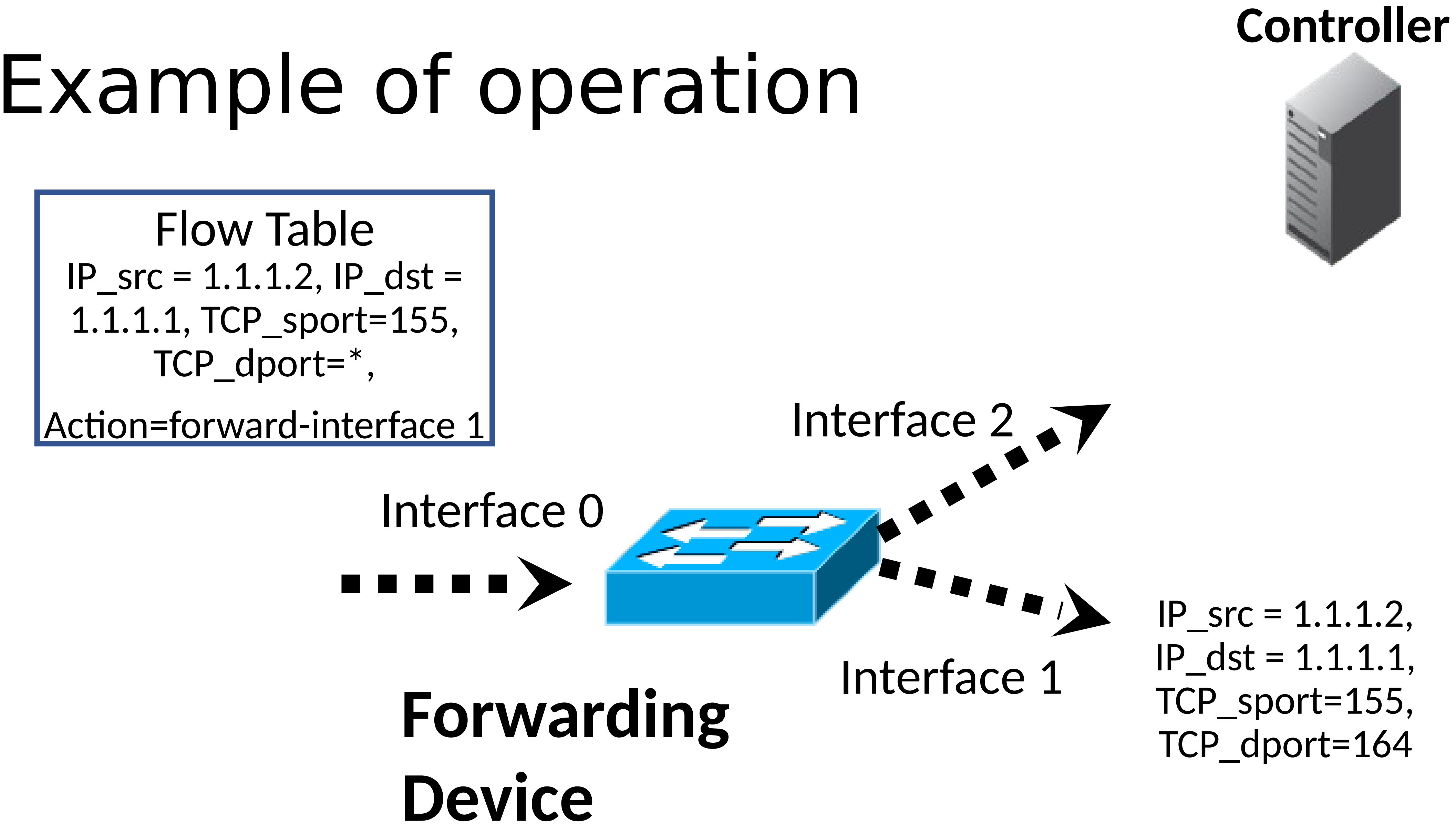
Action=forward-interface 1

IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=164



Forwarding Device

Example of operation



Controller

Flow Table

IP_src = 1.1.1.2, IP_dst = 1.1.1.1, TCP_sport=155, TCP_dport=*,

Action=forward-interface 1

Interface 0

Interface 2

Interface 1

IP_src = 1.1.1.2,
IP_dst = 1.1.1.1,
TCP_sport=155,
TCP_dport=164

Forwarding Device

Network programmability

- Users create their own flow rules
  - They write code to define how the network should handle their traffic
  - Rules are installed by the controller into the forwarding devices
- Users can change their rules over time
  - Rules on the devices expire (soft state)
  - Devices request updated rules from controller over time

Industry adoption/deployment

- A number of vendors exist
  - Widely vary in capability and implementation
    - Expect things to break and not work!
    - Everyone rolls their own deployment
- SDN hardware support

Juniper MX-series

NEC IP8800

WiMax (NEC)

HP Procurve 5400

Netgear 7324

PC Engines

Why companies have bought in?

- SDN/OpenFlow solves problems in enterprise networks
  - People understand it since it is based on TCP/IP
  - Centralized intelligence has its origins in telephony (even before TCP/IP)..

- Single point of control/intelligence
  - Network verification/management/deployment are hard problems
  - SDN makes them easier..

- SDN works well in environments with:
  - Stable connectivity
  - (Minor to no) fault tolerance (or other means to achieve fault tolerance)
  - A single administrative entity

When SDN does not work well?

- Environments with:
  - Intermittent connectivity – if you can't connect to the controller
  - Fault tolerance as an absolute requirement (or with no other means to achieve fault tolerance)
  - Multiple administrative entities

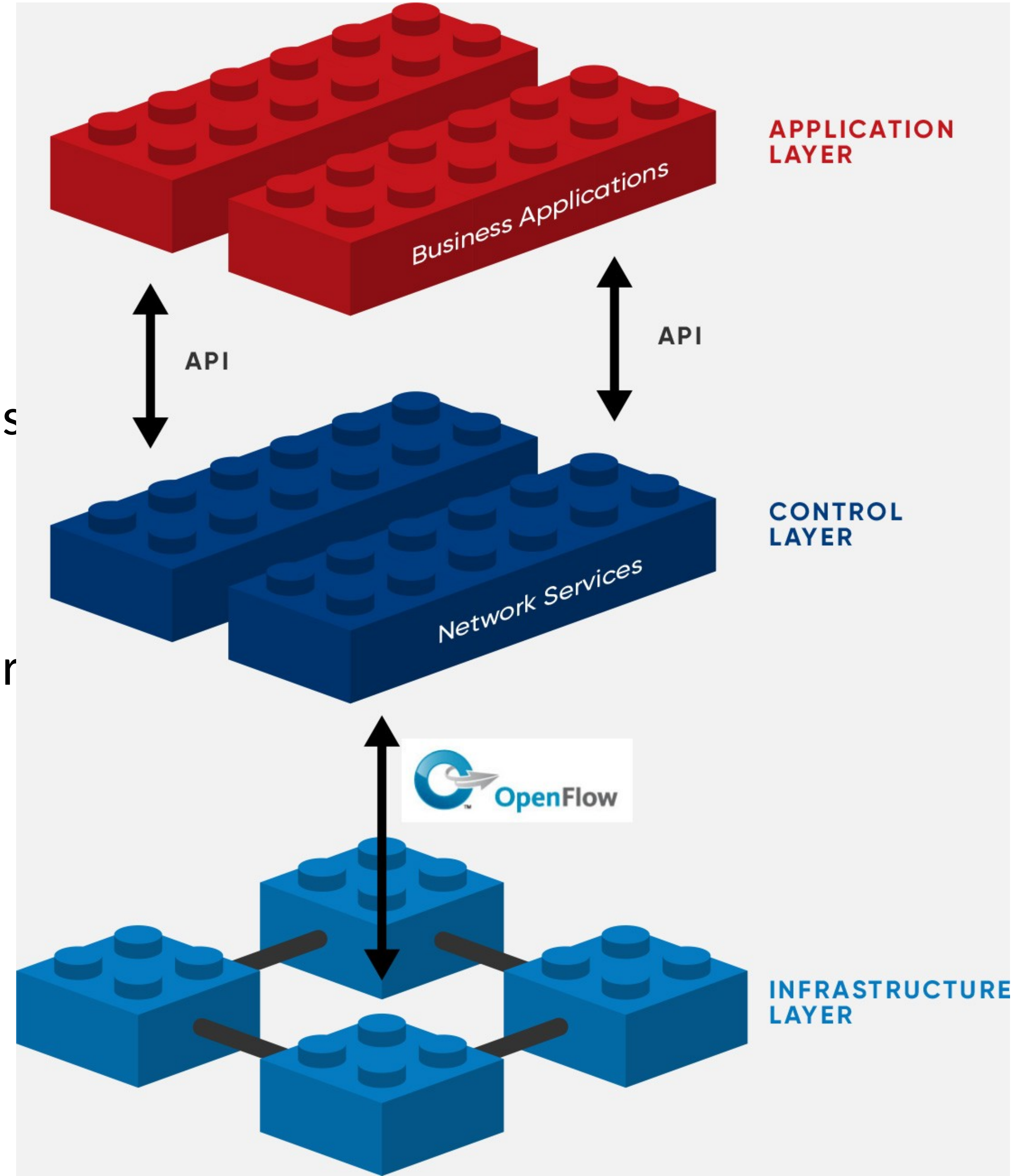
- SDN is an incremental patch to TCP/IP
  - Makes management easier!
  - Adds one more dependency (i.e., the controller)
  - Does not change any of the fundamentals
    - Flows are end-to-end, no security

Security Concerns?

- DDoS the controller, your whole network becomes defunct!
- Man-in-the-middle attacks
- Authentication vulnerability
- And many others….

Conclusion

- SDN: Programmable control plane
  - Users write their own control programs
  - Controller installs packet forwarding rules to forwarding elements
  - OpenFlow is a protocol that facilitates communication between the controller and the forwarding elements



Next Lecture

- Survey due tomorrow
- Next Tuesday –
  - Presentations – Bulbul and Vaibabh
- Next Thursday -
  - Quiz on NDN
  - Presentation - Grant