b

UNIVERSITÄT
BERN

b
UNIVERSITÄT
BERN

# Advanced Networking and Future Internet

## V. Software-Defined Networking

**Prof. Dr. Torsten Braun, Institut für Informatik**

Bern, 12.10.2020

# Advanced Networking and Future Internet: Software-Defined Networking

## Table of Contents

1. Introduction
   1. Legacy Networking Technology Limitations
   2. Network Device Architecture
   3. Restructuring Networks
   4. Mainframes
   5. Routers
   6. Traditional Computer Networks
      1. Data Plane
      2. Control Plane
      3. Management Plane
2. Software Defined Networking
   1. Architecture
   2. Controller: Programmability
   3. Distributed Controller
   4. Data Plane: Simple Packet Handling
   5. Network Devices
3. OpenFlow
   1. Operation
   2. Flow Table
   3. Packet Processing
   4. Instructions
   5. Actions and Action Sets
   6. Packet Flow
   7. Matching and Instruction Execution
   8. Group Table
   9. Meter Table
   10. Protocol
4. SDN Summary
   1. Key Issues
   2. Benefits
5. Applications
   1. Dynamic Access Control
   2. Seamless Mobility/Migration
   3. Server Load Balancing
   4. Network Virtualization

# 1. Introduction

## 1. Legacy Networking Technology Limitations

- Large set of various protocols, slow standardization
- Buggy equipment software
- Huge network operation costs
- Risk of inconsistent policies and configurations
- Scalability
- Vendor dependence and closed equipment

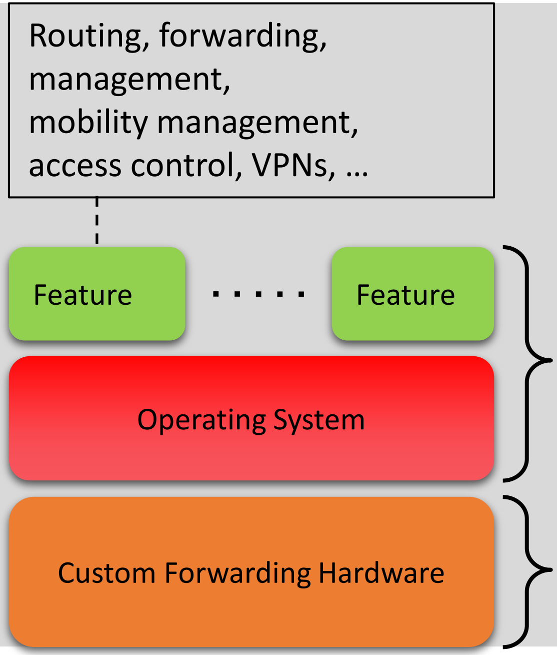
# 1. Introduction

# 2. Network Device Architecture

– extremely complex

– mainframe mentality

– very expensive

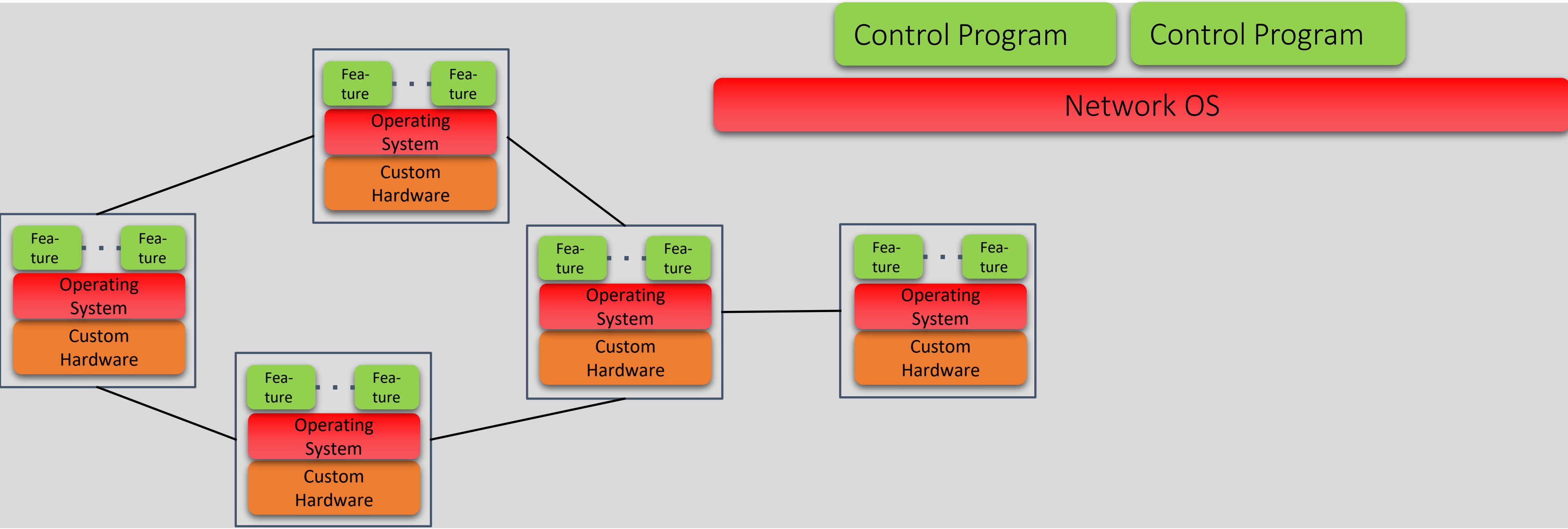


millions of lines of source code (based on > 8000 RFCs)

billions of power-hungry gates

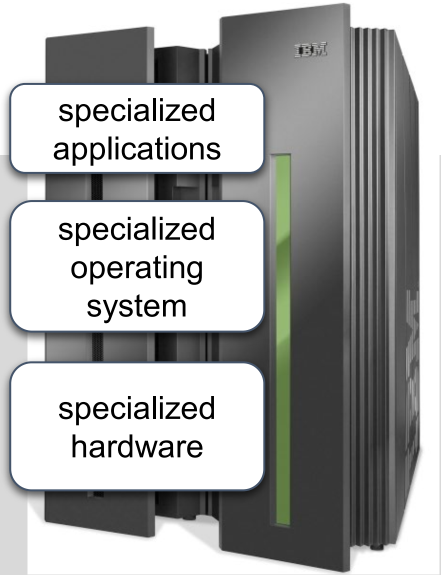
# 1. Introduction

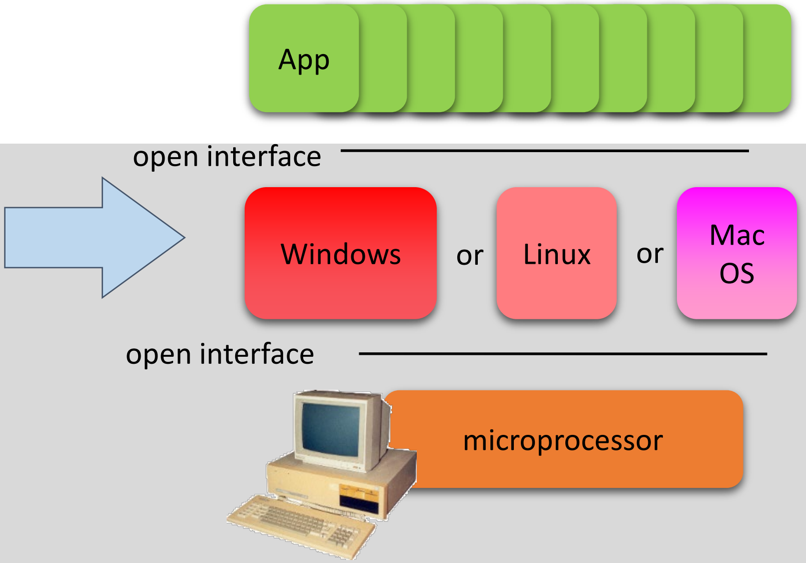
# 3. Restructuring Networks



# 1. Introduction

## 4. Mainframes



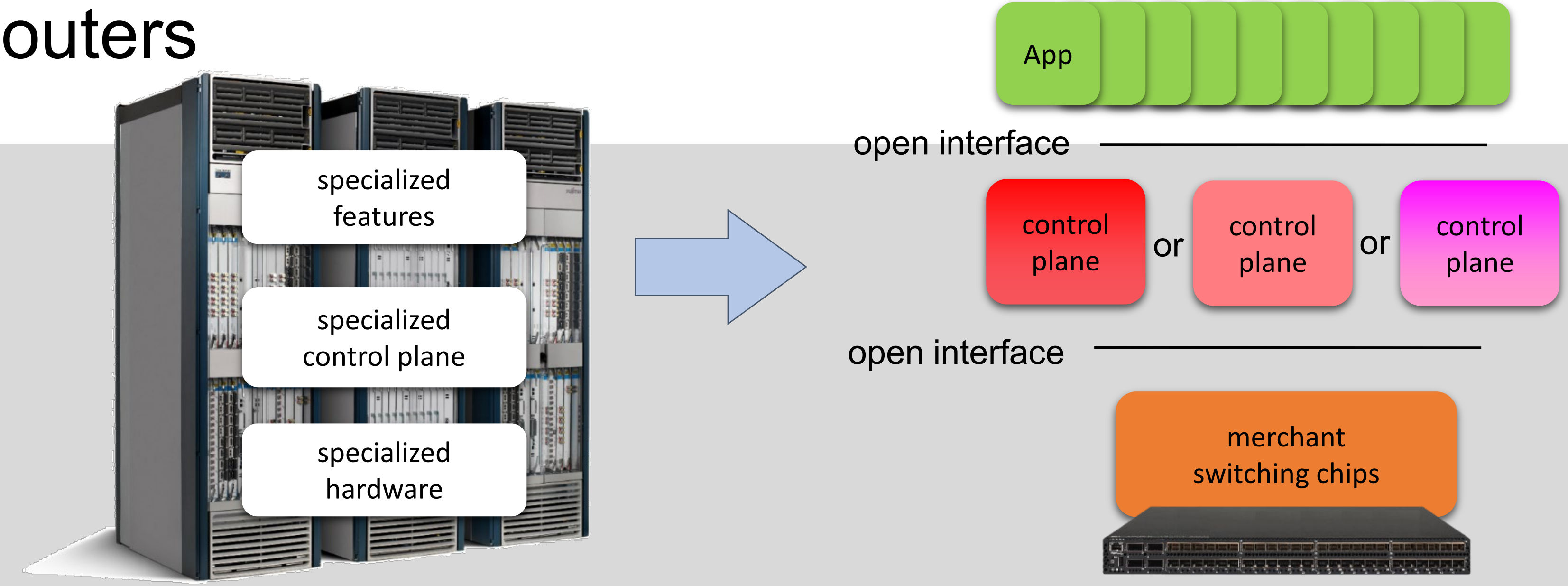


- vertically integrated
- closed, proprietary
- slow innovation
- small industry

- horizontal
- open interfaces
- rapid innovation
- huge industry

# 1. Introduction

## 5. Routers



specialized features

specialized control plane

specialized hardware

App

open interface

control plane or control plane or control plane

open interface

merchant switching chips

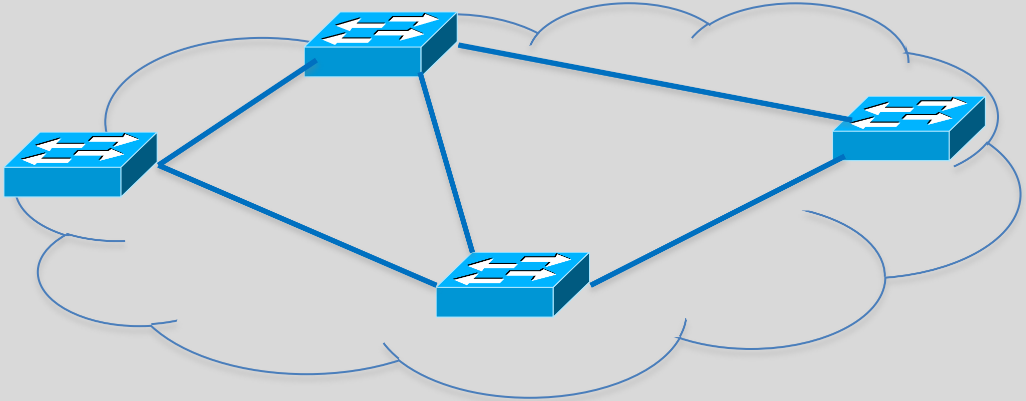
- vertically integrated
- closed, proprietary
- slow innovation

- horizontal
- open interfaces
- rapid innovation

# 1. Introduction

## 6.1 Traditional Computer Networks: Data Plane

data plane: packet handling

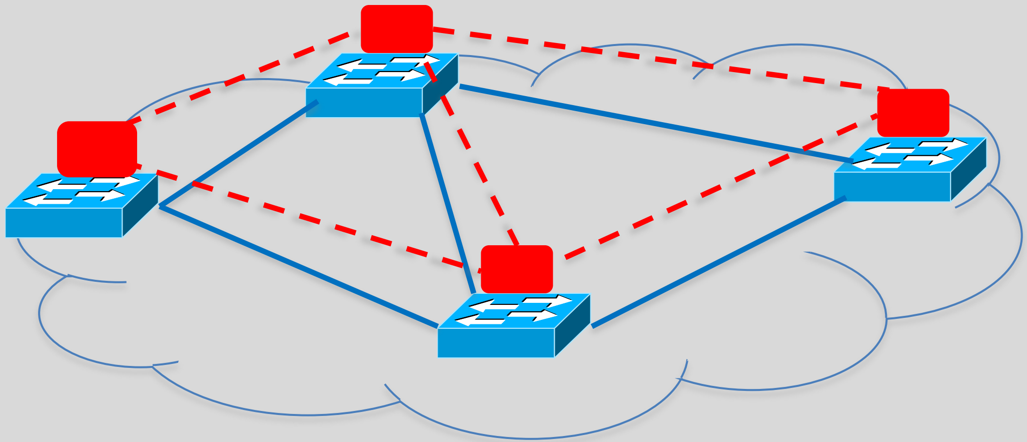


forward, filter, buffer, mark, rate-limit, measure packets

# 1. Introduction

## 6.2 Traditional Computer Networks: Control Plane

control plane: distributed algorithms

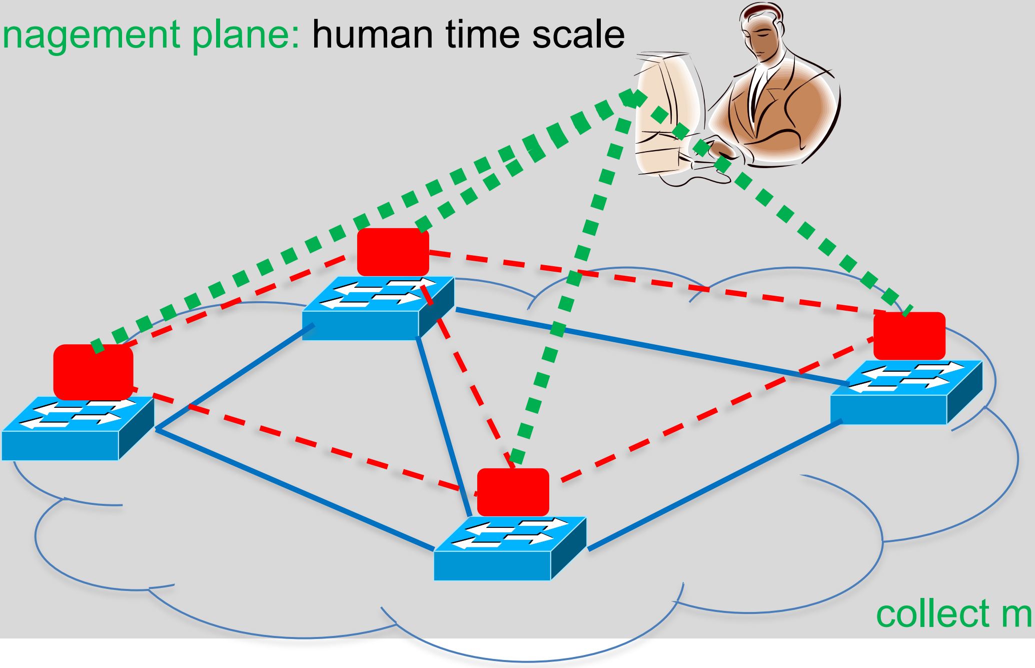


track topology changes, compute routes, install forwarding rules

# 1. Introduction

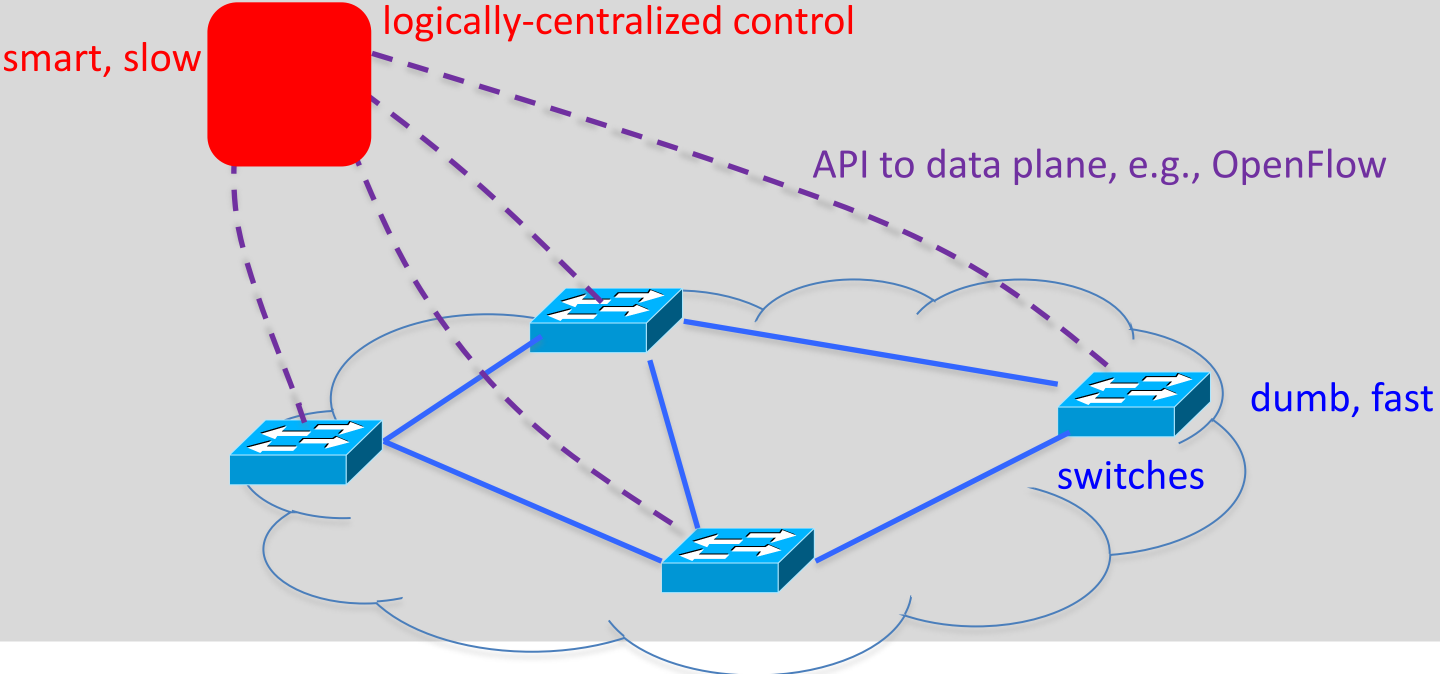
## 6.3 Traditional Computer Networks: Management Plane

management plane: human time scale



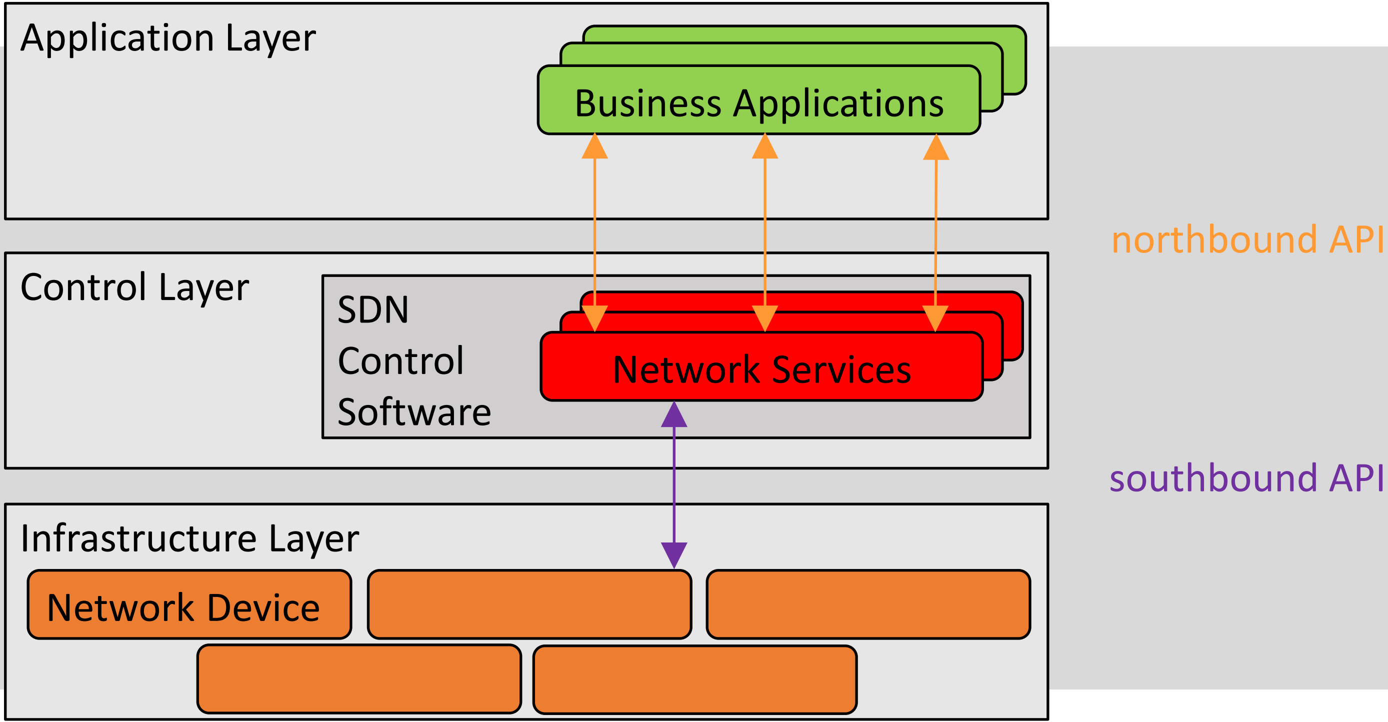
collect measurements and configure equipment

# 2. Software Defined Networking


logically-centralized control
smart, slow
API to data plane, e.g., OpenFlow
dumb, fast
switches

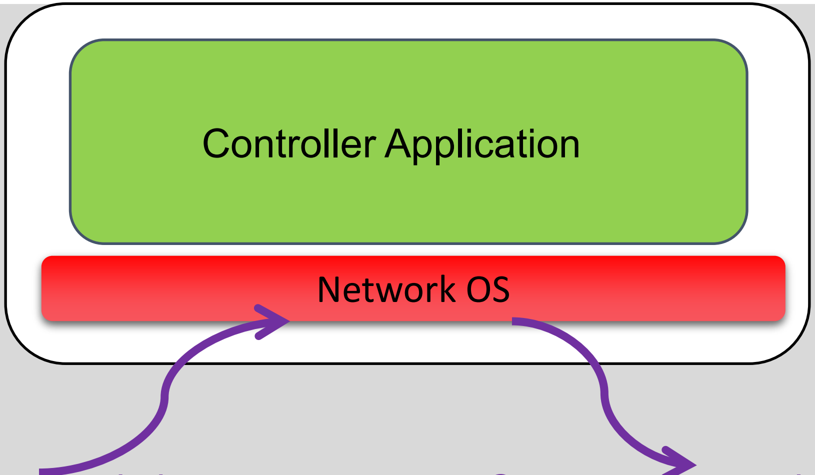
# 2. Software Defined Networking

## 1. Architecture



# 2. Software Defined Networking

## 2. Controller: Programmability



Events from switches

- topology changes
- traffic statistics
- arriving packets

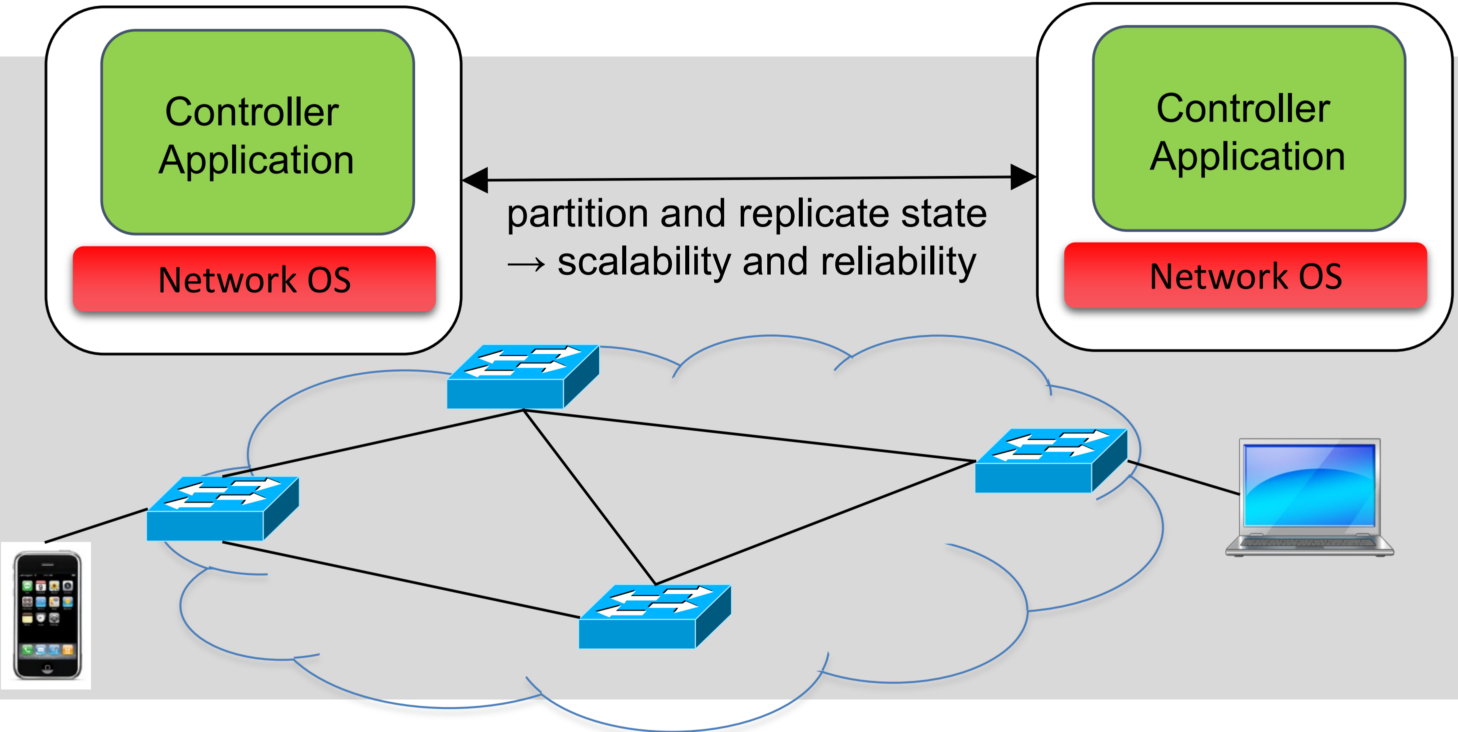
Commands to switches

- (un)install rules
- query statistics
- send packets

UNIVERSITÄT BERN

## 2. Software Defined Networking

## 3. Distributed Controller



partition and replicate state
→ scalability and reliability

# 2. Software Defined Networking

## 4. Data Plane: Simple Packet Handling Rules

- Pattern: match packet header bits
- Actions: drop, forward, modify, send to controller
- Priority: disambiguate overlapping patterns
- Counters: #bytes and #packets (received, transmitted, dropped, erroneous, …)

| MAC src | MAC dst | IP src | IP dst | TCP dst port | … | Action | Count |
|---|---|---|---|---|---|---|---|
| * | 10:20:… | * | * | * | * | Port1 | 250 |
| * | * | * | 5.6.7.8 | * | * | Port2 | 300 |
| * | * | * | * | 25 | * | Drop | 892 |
| * | * | * | 192.* | * | * | Local | 120 |
| * | * | * | * | * | * | Controller | 11 |

# 2. Software Defined Networking

## 5. Network Devices

**Router**

- Match:
  longest destination IP prefix
- Action: forward via a link

**Switch**

- Match:
  destination MAC address
- Action: forward or flood

**Firewall**

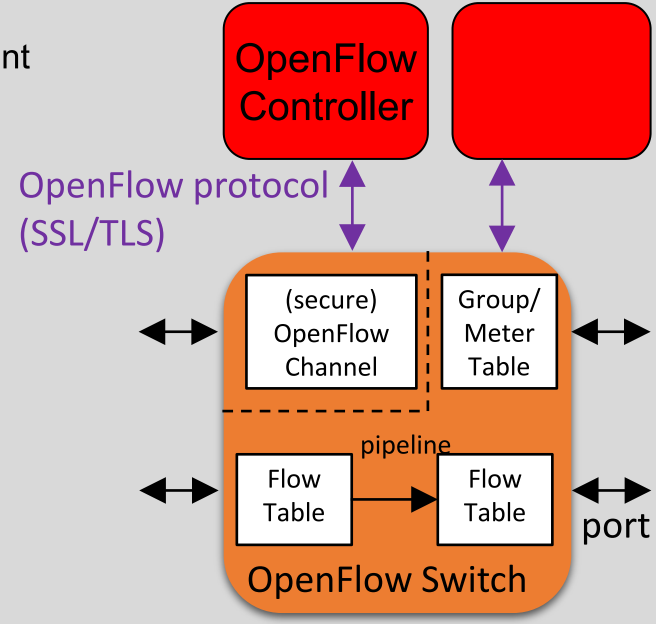
- Match: IP addresses and TCP/UDP port numbers
- Action: permit or deny

**Network Address Translator**

- Match: IP address and port
- Action: rewrite address and port

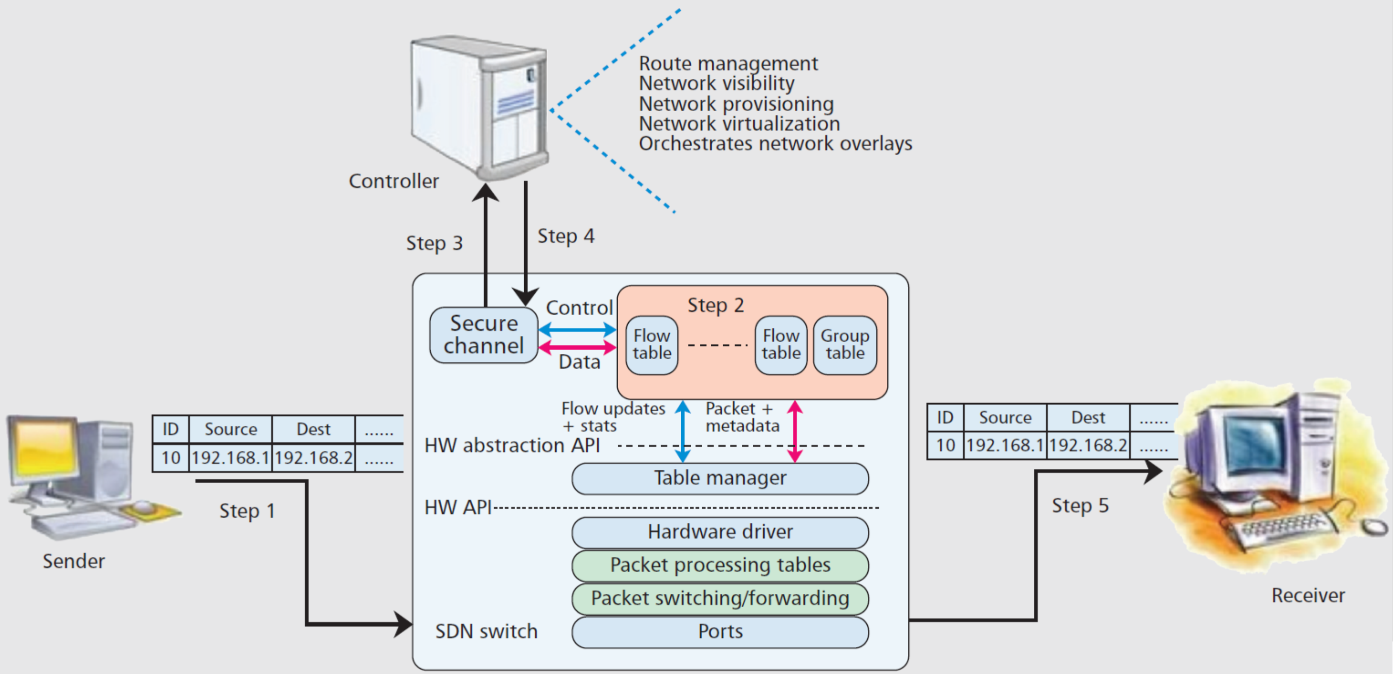
# 3. OpenFlow

- Current version: 1.5
- Most Ethernet switches and routers contain flow tables based on content addressable memory running at line-rate to
  - support QoS
  - implement firewalls and NATs
  - collect statistics
- Switches and routers provide similar functionality, but have proprietary flow tables (data path = flow table + actions).
- OpenFlow aims to provide a standard interface to program flow tables.
- OpenFlow Switch
  - Flow / group / meter tables
  - Secure channel for configuration
  - OpenFlow protocol
- OpenFlow Controller
  - adds / changes / removes table entries.



# 3. OpenFlow

## 1. Operation



| ID | Source | Dest | ...... |
|---|---|---|---|
| 10 | 192.168.1 | 192.168.2 | ...... |

| ID | Source | Dest | ...... |
|---|---|---|---|
| 10 | 192.168.1 | 192.168.2 | ...... |

# 3. OpenFlow

## 2. Flow Table

| | | | | | | |
|---|---|---|---|---|---|---|
| Match fields | Priority | Counter | Instructions | Timeouts | Cookie | Flags |

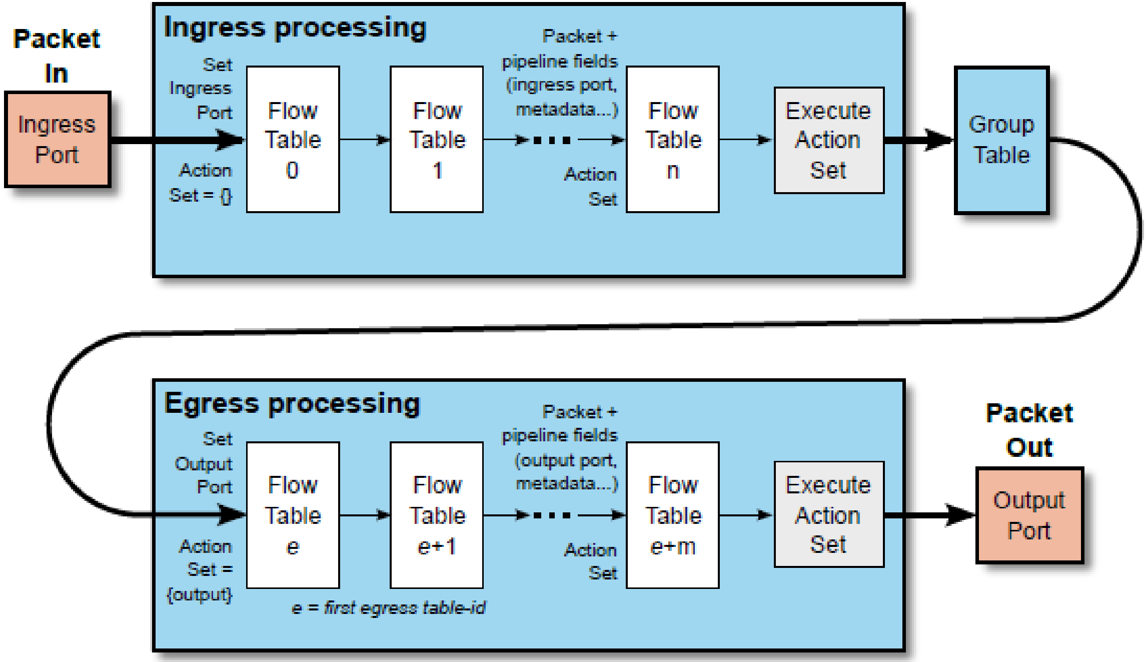
| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Switch port | MAC src | MAC dst | Eth type | VLAN ID | VLAN prio | MPLS label | MPLS traffic cl. | IP src | IP dst | IP proto | IP ToS | src port | dst port |

- Match fields: to match against packets, consisting of ingress port and packet headers, and optionally metadata
- Priority:
  matching precedence of flow entry
- Counters:
  updated when packets are matched
- Instructions: to modify action set or pipeline processing
- Timeouts: maximum amount of time or idle time before flow is expired by switch
- Cookie:
  opaque data value chosen by controller
- Flags: to trigger certain reactions

# 3. OpenFlow

## 3. Packet Processing

≥ 1 ingress flow table



# 3. OpenFlow

## 4. Instructions

- Each flow table entry contains a set of instructions, which are executed when a packet matches the entry.
- Instructions change packet, action set and / or pipeline processing (directing packet to other flow table).

**Required instructions**

- Write-Actions: merges specified action into current action set
- Goto-Table: indicates next table in processing pipeline
- Clear-Actions: clear all actions immediately

**Optional instructions**

- Stat-Trigger: Generate event to controller, if some flow statistics exceed threshold values.
- Apply-Actions: applies specified actions immediately
- Write-Metadata: write masked metadata value into metadata field

# 3. OpenFlow

## 5. Actions and Action Sets

### Actions

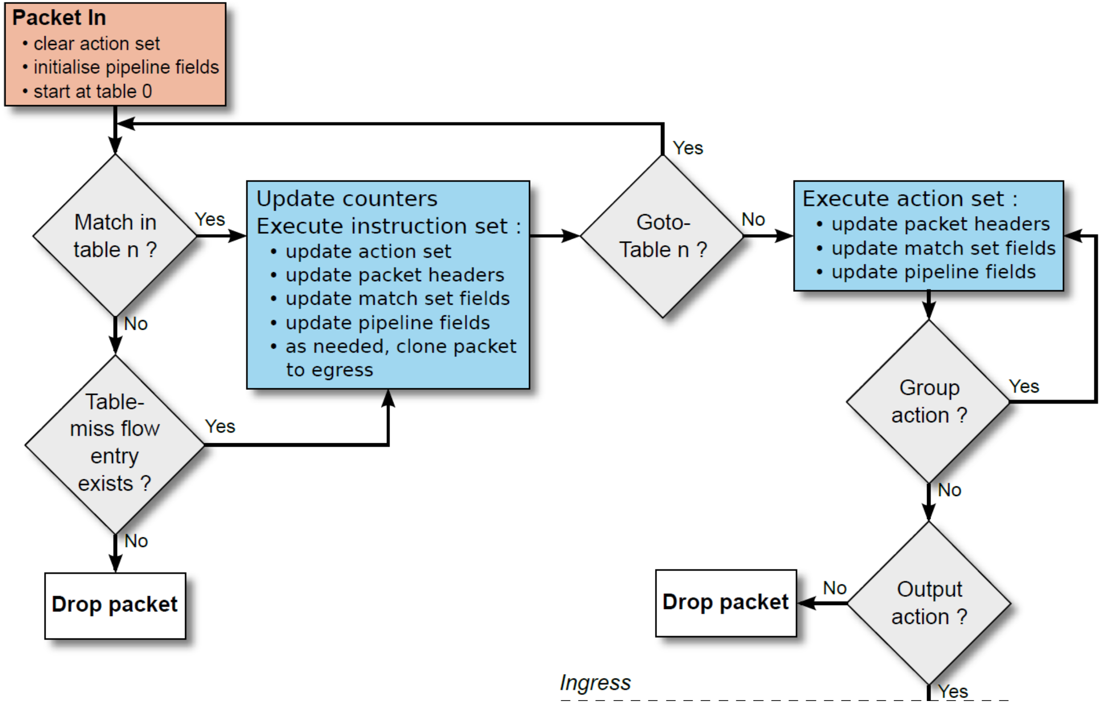
- Decrementing / copying TTL values
- Pop / push tags (MPLS, VLAN headers)
- Set MAC / IP addresses, VLAN IDs, port numbers, IP header bits etc.
- Quality-of-service actions, e.g., assign queues or meters to a packet
- Group actions
- Output of packets

### Action Sets

- are associated with each packet and are empty at the beginning for ingress processing and an output action for output processing.
- Flow entry instructions modify action set.
- Execution of actions at the end of pipeline, exception: Apply-Actions

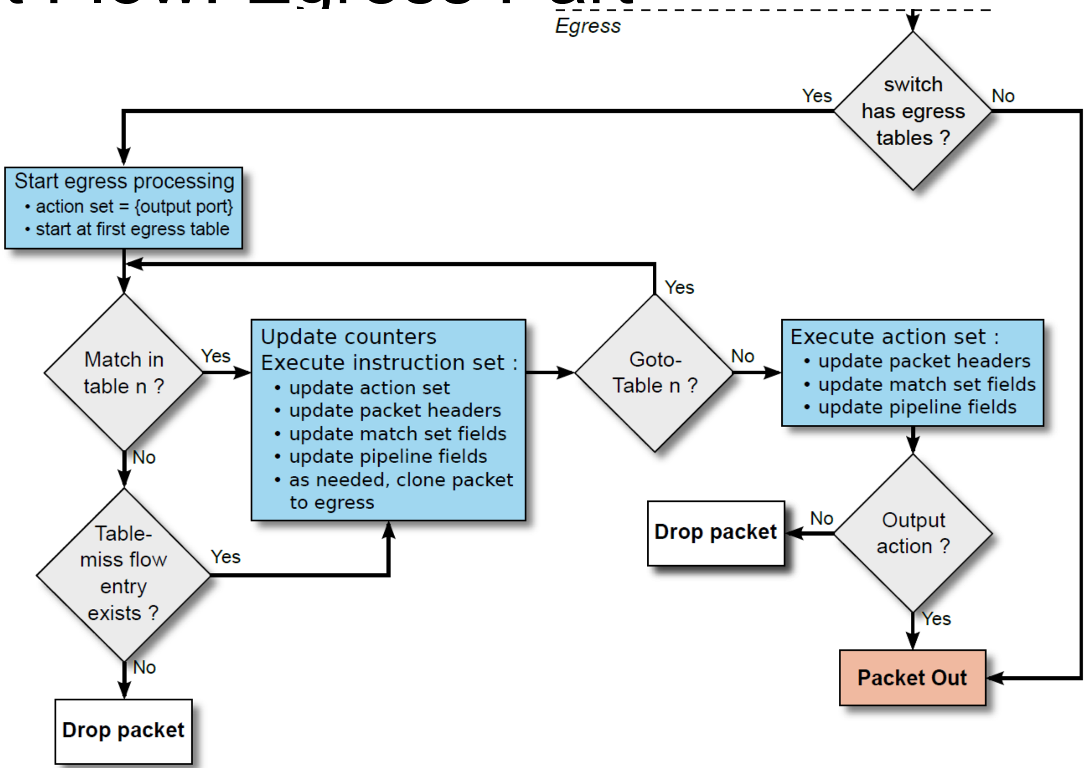
# 3. OpenFlow

## 6.1 Packet Flow: Ingress Part



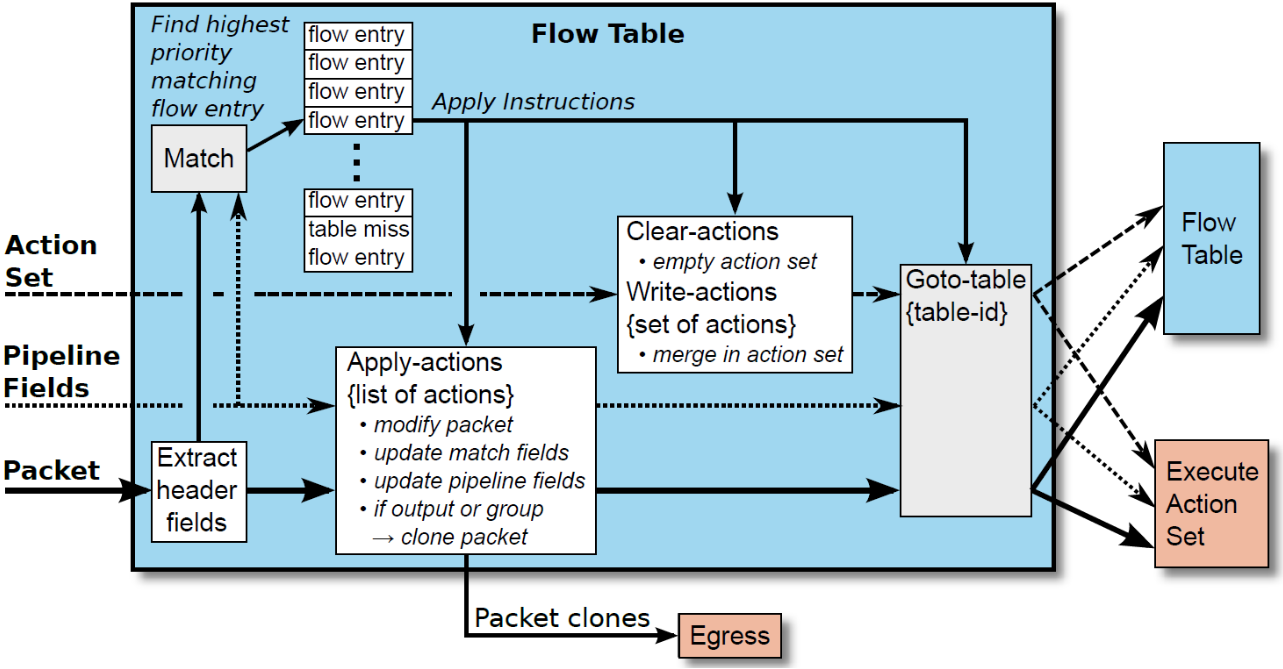
# 3. OpenFlow

## 6.2 Packet Flow: Egress Part



# 3. OpenFlow

## 7. Matching and Instruction Execution



# 3. OpenFlow

## 8. Group Table

- Group: list of Action Buckets and some means to choose one or more buckets per packet

| Group Identifier | Group Type | Counters | Action Buckets |
|---|---|---|---|

- Group table entry
  - Group Identifier: 32 bit unsigned integer uniquely identifying the group
  - Group Type: to determine group semantics
    - All: execute all buckets, multicast or flooding (required)
    - Indirect: one bucket, simple indirection (required)
    - Select: execute one bucket, for multipath forwarding and load balancing (optional)
    - Fast failover: execute first live bucket (optional)
  - Counters: updated when packets are processed by a group
  - Action Buckets: an ordered list of action buckets,
    where each Action Bucket contains a set of actions to execute and associated parameters

# 3. OpenFlow

# 9. Meter Table

## Meter Table Entry

- Meter bands define how packets are processed if target rate is exceeded.

| | | |
|---|---|---|
| Meter Identifier | Meter Bands | Counters |

## Applications

- Quality of Service operation
- Rate limiting and policing
- Metering
- Packet classification

# 3. OpenFlow

## 10. Protocol

**Message types**

- Controller-to-switch to
  - request switch features
  - set and query configuration parameters
  - add, modify, delete flow / group / meter table entries
  - read switch statistics
  - output packets to switch
  - set role of controller in case of multiple controllers (master / slave)
  - set filter for asynchronous messages
- Asynchronous: unsolicited messages from switch to controller to
  - receive packets
  - indicate flow removal after timeout
  - indicate port, role, table, controller status
- Symmetric (without solicitation in either direction)
  - Hello messages
  - Echo messages
  - Error messages

**Connection setup**

- TLS/TCP connection

# 4. SDN Summary

## 1. Key Issues

- Separation of control plane from data plane
- Centralized controller and centralized view of the network
- Open interfaces between devices in control plane (controllers) and those in data plane (network devices)
- Network programmability by external applications

# 4. SDN Summary

## 2. Benefits

- Simpler centralized network management and control
  - increases network reliability
  - minimizes inconsistent configuration
- Improved automation and management by common APIs
- Abstraction of underlying network details from orchestration / provisioning systems and applications
- Rapid innovation independent from network device manufacturers
- Programmability by operators and users
- More fine-granular network control
- Simpler, faster, cheaper network devices

# 5. SDN Applications

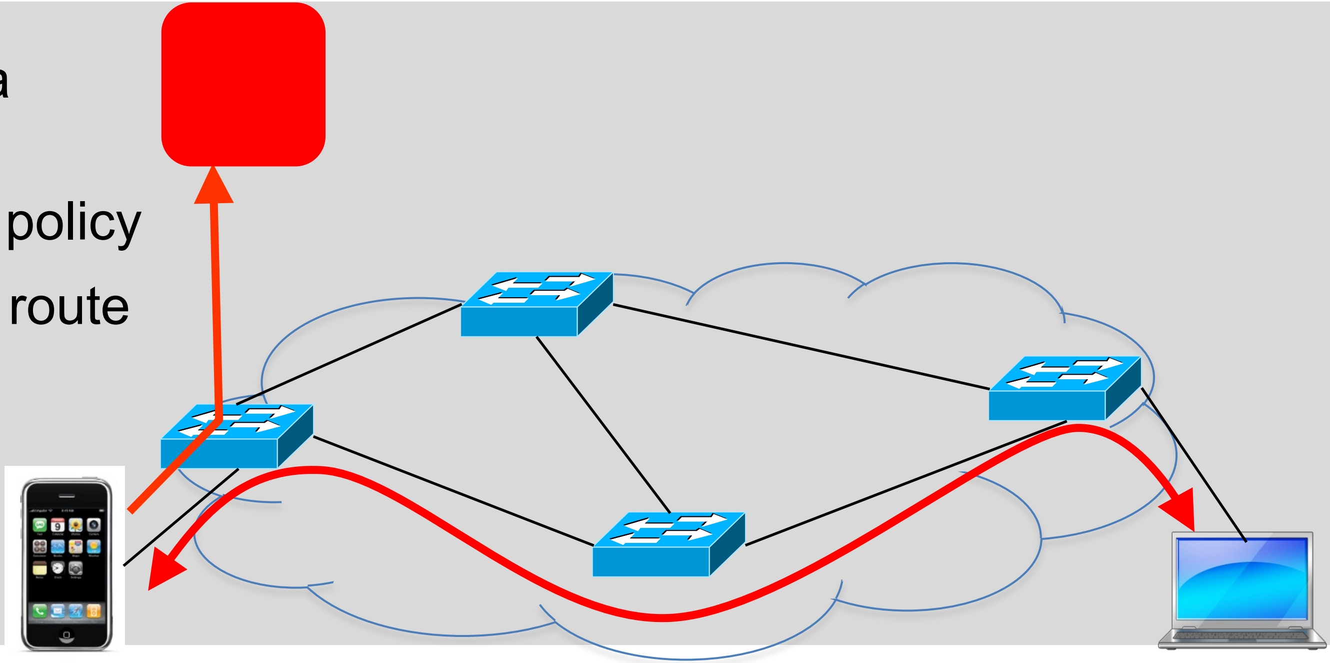
1. *Dynamic access control*
2. *Seamless mobility/migration*
3. *Server load balancing*
4. *Network virtualization*

- Link failure recovery
- Data centre networking
- Multiple wireless access points
- Energy-efficient networking
- Adaptive traffic monitoring
- Denial-of-Service attack detection
- Network management and control
- Virtual networks
- Non-IP networks, Future Internet protocols
- Deep-packet inspection, intrusion detection

# 5. Applications

## 1. Dynamic Access Control

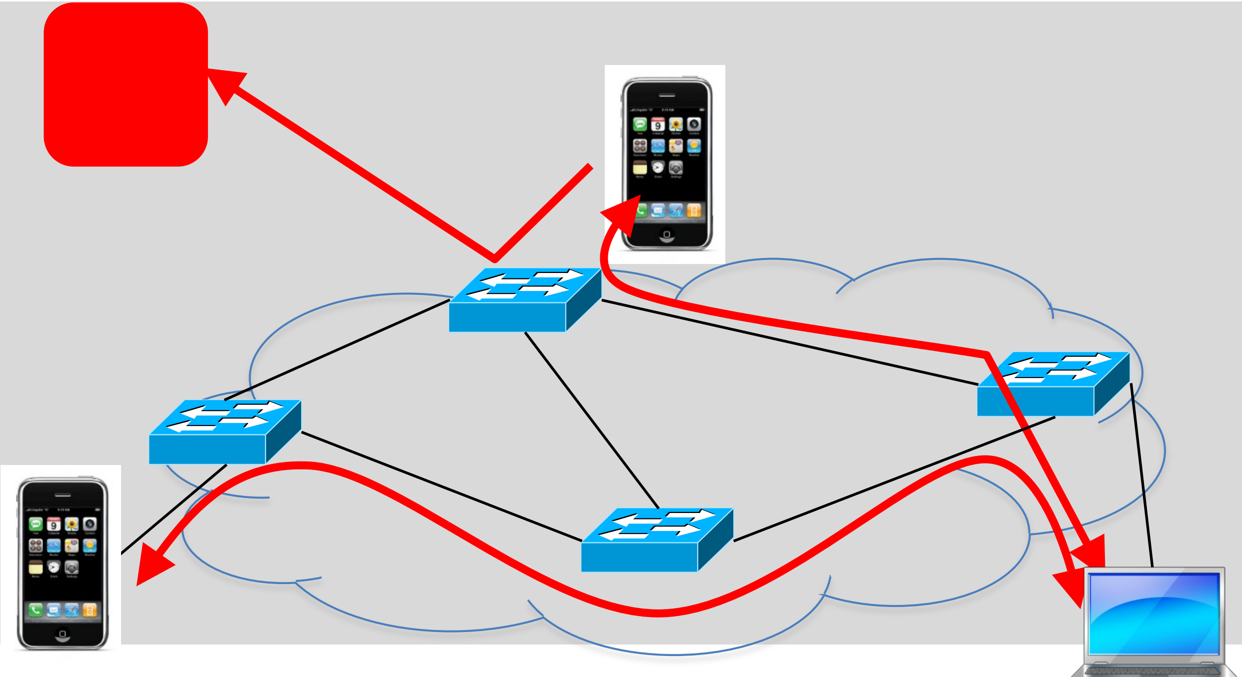
- inspect first packet of a connection
- consult access control policy
- install rules to block or route traffic



# 5. Applications

## 2. Seamless Mobility / Migration

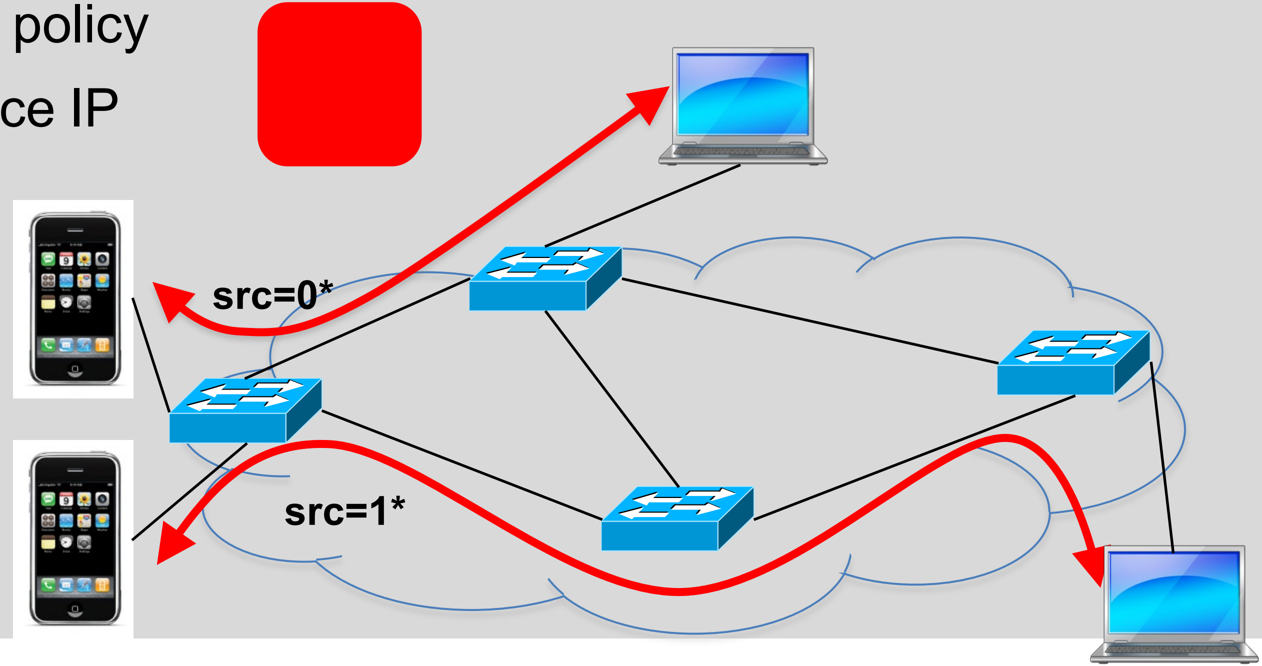
- see host to send traffic at new location
- modify rules to reroute traffic



# 5. Applications

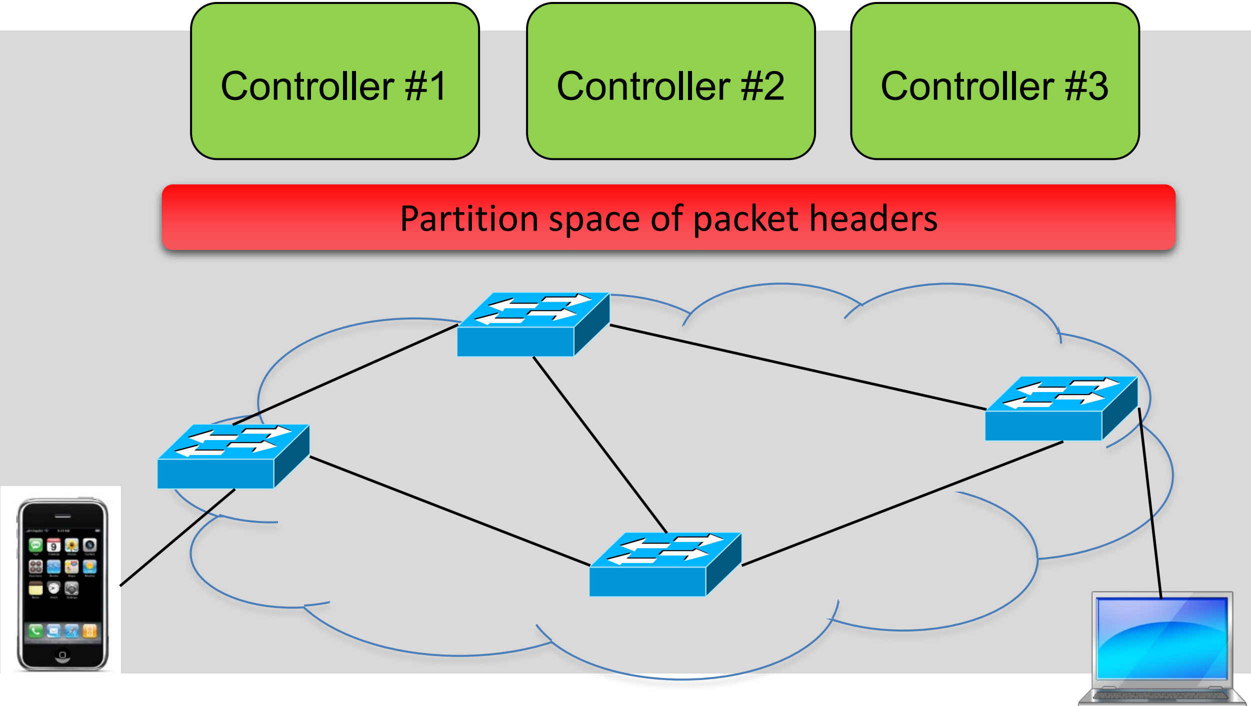
## 3. Server Load Balancing

- Pre-install load-balancing policy
- Split traffic based on source IP



# 5. Applications

## 4. Network Virtualization



# Thanks

## for Your Attention

**Prof. Dr. Torsten Braun, Institut für Informatik**

Bern, 12.10.2020