Skoltech
Skolkovo Institute of Science and Technology

Lomonosov Moscow State University

Lecture 2: OpenFlow Protocol

Vasily Pashkov
pashkov@lvk.cs.msu.su

Skoltech
Skolkovo Institute of Science and Technology

What is software defined networking?

- Software-defined networking (SDN) is an approach to computer networking that allows network administrators to manage network services through abstraction of lower-level functionality.
  - Abstractions for three problems: constrained forwarding model, distributed state, detailed configuration
- SDN is
  - Directly programmable: network control is programmable because it is decoupled from forwarding functions
  - Agile: administrator can dynamically adjust network-wide traffic flow to meet changing needs.
  - Centrally managed: network intelligence is logically centralized.
  - Programmatically configured
  - Open standards-based and vendor-neutral

Forwarding abstraction

- Control plane needs flexible forwarding model
  - With behavior specified by control program applications
    - Use a generic “flow” concept that is inclusive and forward based on flows.
    - Historically the hardware’s capability for forwarding is vendor dependent
      - e.g. forwarding based on L2 address, L3 address
  - This abstracts away forwarding hardware
  - Flexibility and vendor-neutrality are both valuable

Skoltech
Skolkovo Institute of Science and Technology

State Distribution Abstraction

- Shield control mechanisms from state distribution while allowing access to the state
  - Split global consensus-based distributed algorithms into two independent components: a distributed (database) system and a centralized algorithm.
    - We know how to deal with both.
- Natural abstraction: global network view
- Implemented with a network operating system.
- Control (configuration) mechanism is now abstracted as a function of the global view using API
  - Control is now based on a centralized graph algorithm instead of a distributed protocol.

Network Operating System(NOS)

- NOS: a distributed system that creates and maintains a network view
- Communicates with forwarding elements
  - Get state information from forwarding elements
  - Communicates control directives to forwarding elements
    - Using forwarding abstraction

- NOS plus forwarding abstraction = SDN (v1)

Configuration abstraction

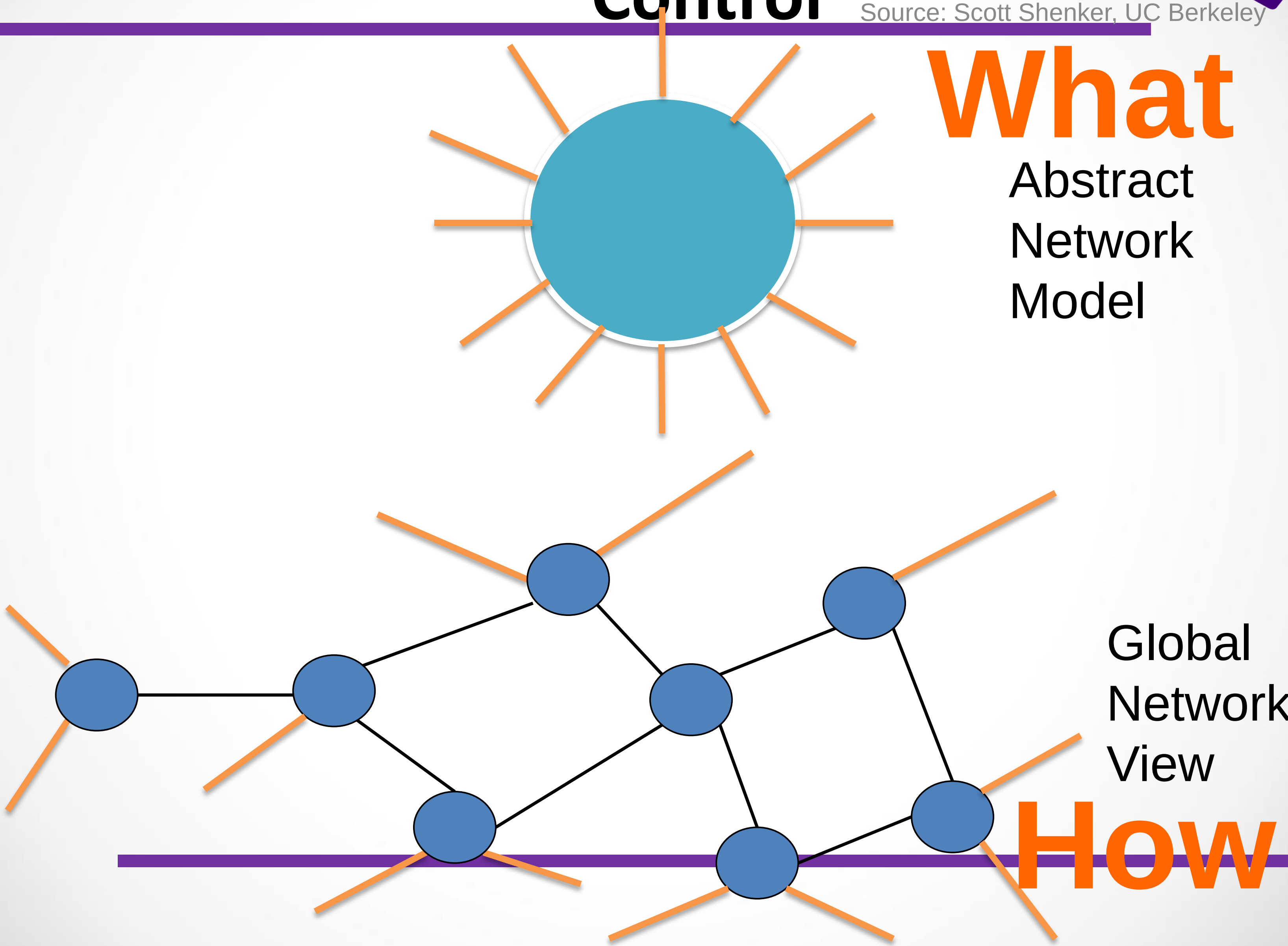
- Application should not configure each individual network device.
- The NOS provides consistent global view of the network
- Configuration is a function of the global view
- NOS eases the implementation of functionality
  - Does not help specification of functionality
- Need a specification abstraction

Specification abstraction

- Give control programs an abstract view of network
  - Abstract view is a function of global view. The abstract view could be just a giant switch connecting all ports, or individual logical topology for each application.
- Control program is abstract mapping
  - Abstract configuration = Function (abstract view)
- Abstraction models should have just enough detail to specify goals
  - Don't provide information needed to implement goals.

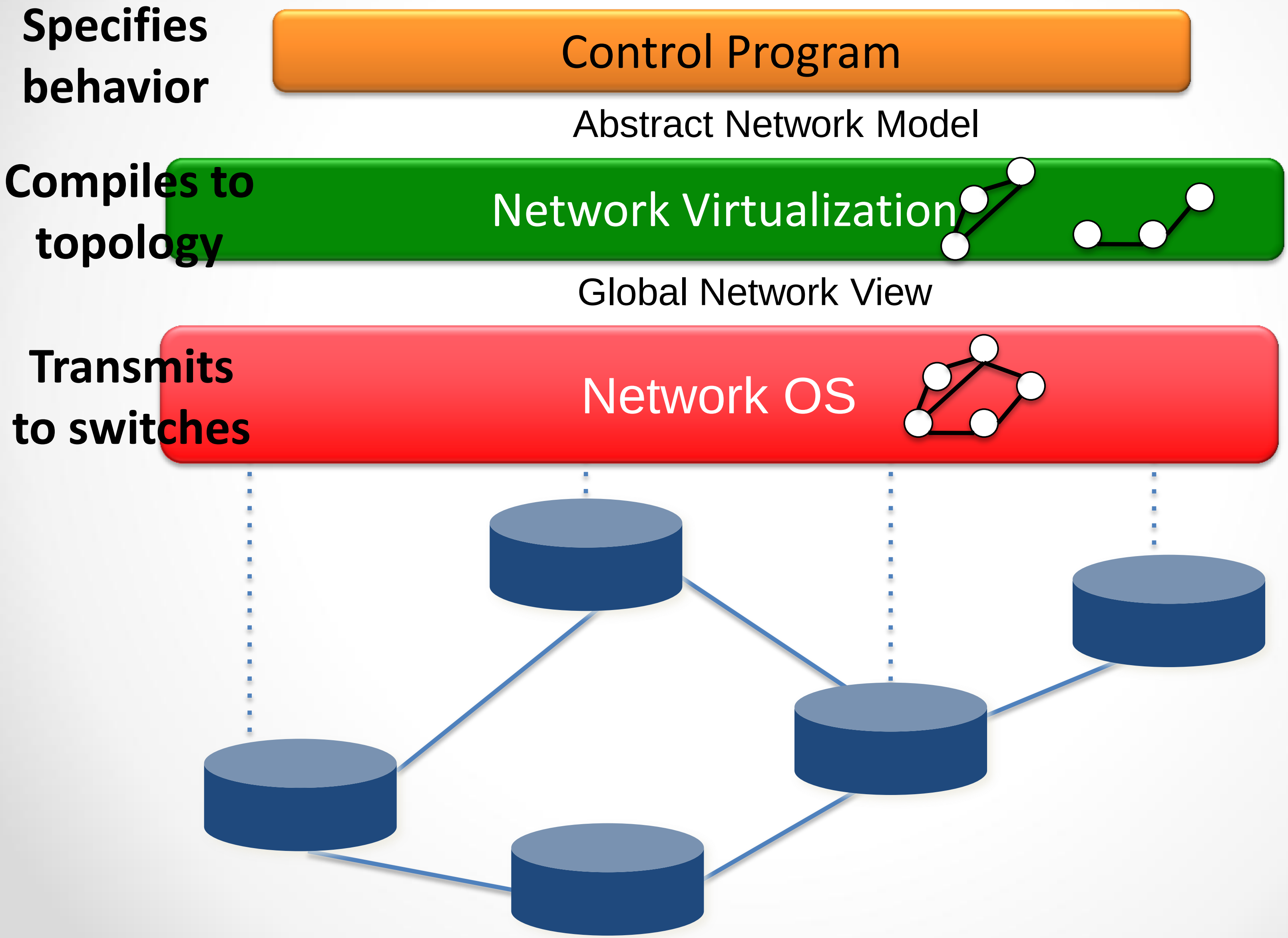
Simple Example: Access Control

Source: Scott Shenker, UC Berkeley



Software Defined Networks

Source: Scott Shenker, UC Berkeley



What Does This Picture Mean?

Source: Scott Shenker, UC Berkeley

- Write a simple program to configure a simple model
  - Configuration is merely a way to specify what you want
- Examples
  - ACLs: who can talk to who
  - Isolation: who can hear my broadcasts
  - Routing: only specify routing to the degree you care
    - Some flows over satellite, others over landline
  - TE: specify in terms of quality of service, not routes
- Virtualization layer “compiles” these requirements
  - Produces suitable configuration of actual network devices
- NOS then transmits these settings to physical boxes

Openflow: Simplifying the control

Routing, management, mobility management, access control, VPNs, …

Feature · · · · · · Feature

Operating System

Specialized Packet Forwarding Hardware

Million of lines of source code	5400 RFCs	Barrier to entry
Billions of gates	Bloated	Power Hungry

Many complex functions baked into the infrastructure

OSPF, BGP, multicast, differentiated services,
Traffic Engineering, NAT, firewalls, MPLS, redundant layers, …

Ossified networks today

OpenFlow: a pragmatic compromise

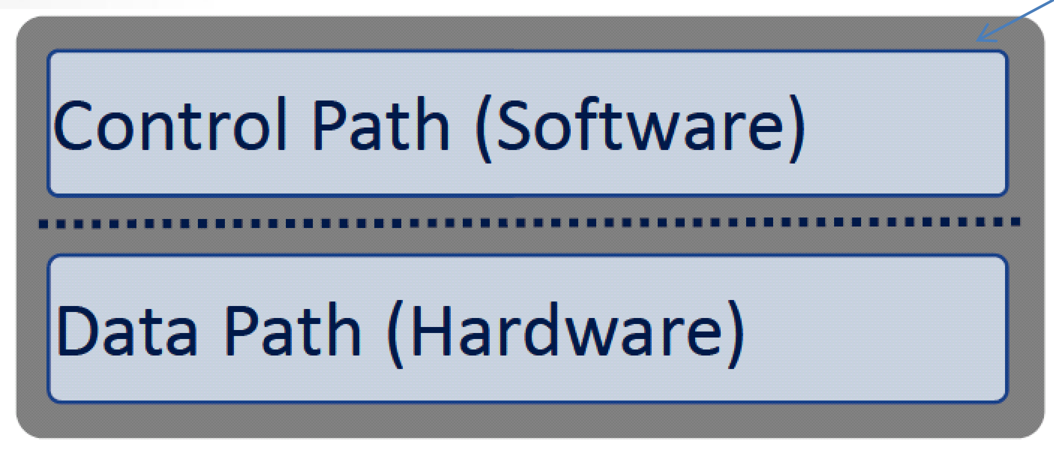
- + Speed, scale, fidelity of vendor hardware
- + Flexibility and control of software and simulation
- Vendors don’t need to expose implementation
- Leverages hardware inside most switches today (ACL tables)

Skoltech
Skolkovo Institute of Science and Technology

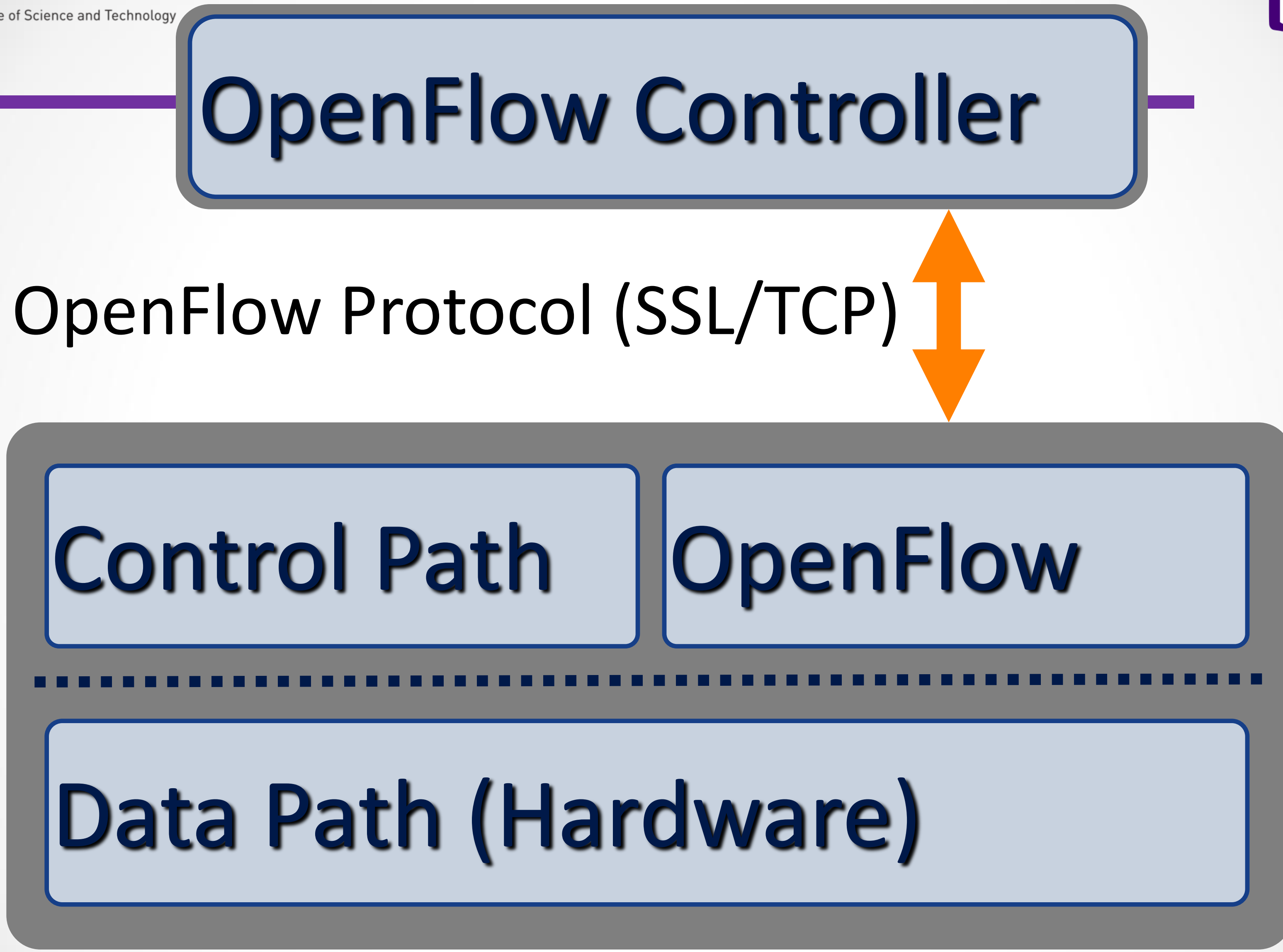
How does OpenFlow work?

Ethernet switch

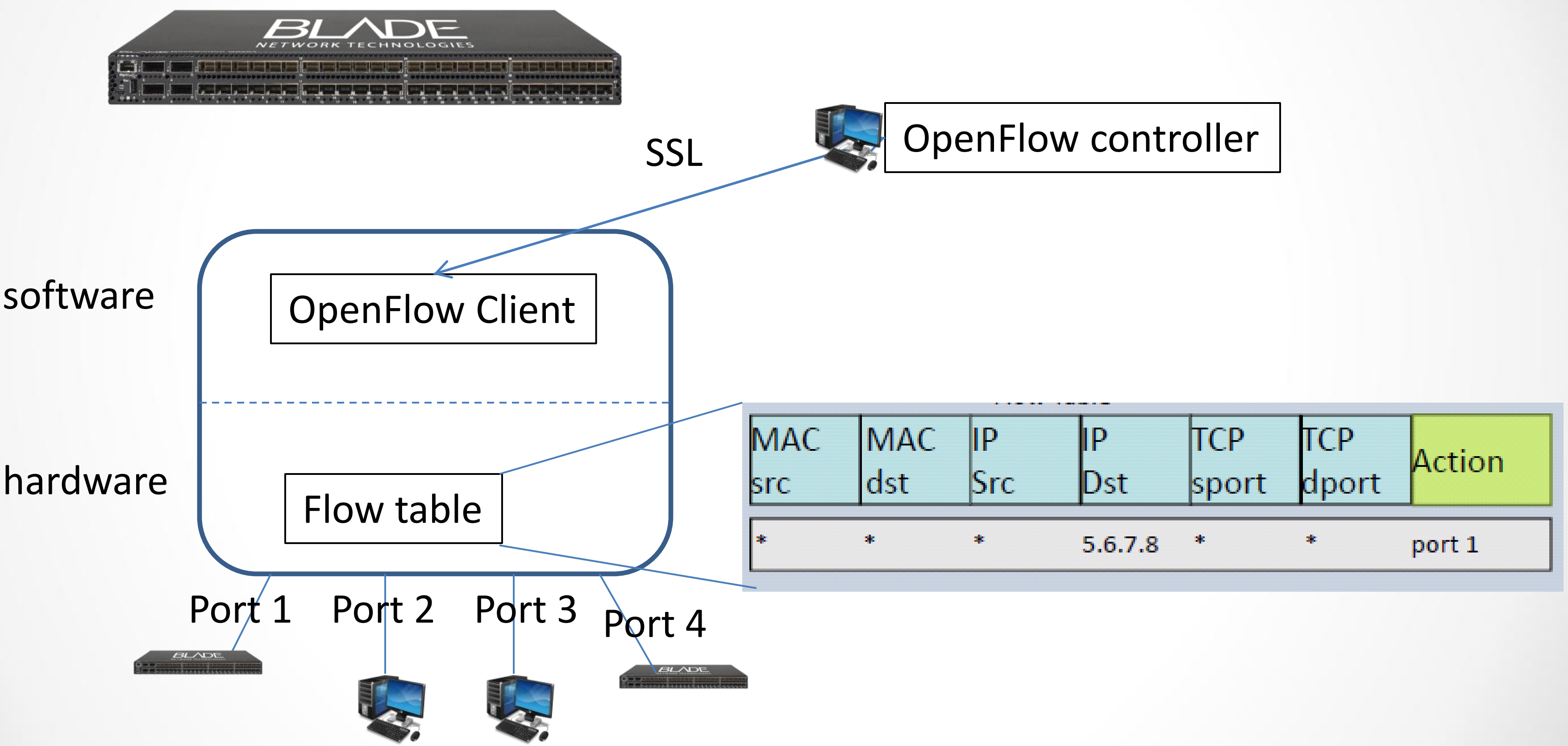
What sets the forwarding
Table in Ethernet?



Forwarding table:
12:12:12:12:12:12 port 1
3f:13:33:ef:ff:ff port 2


OpenFlow Controller
OpenFlow Protocol (SSL/TCP)
Control Path
OpenFlow
Data Path (Hardware)

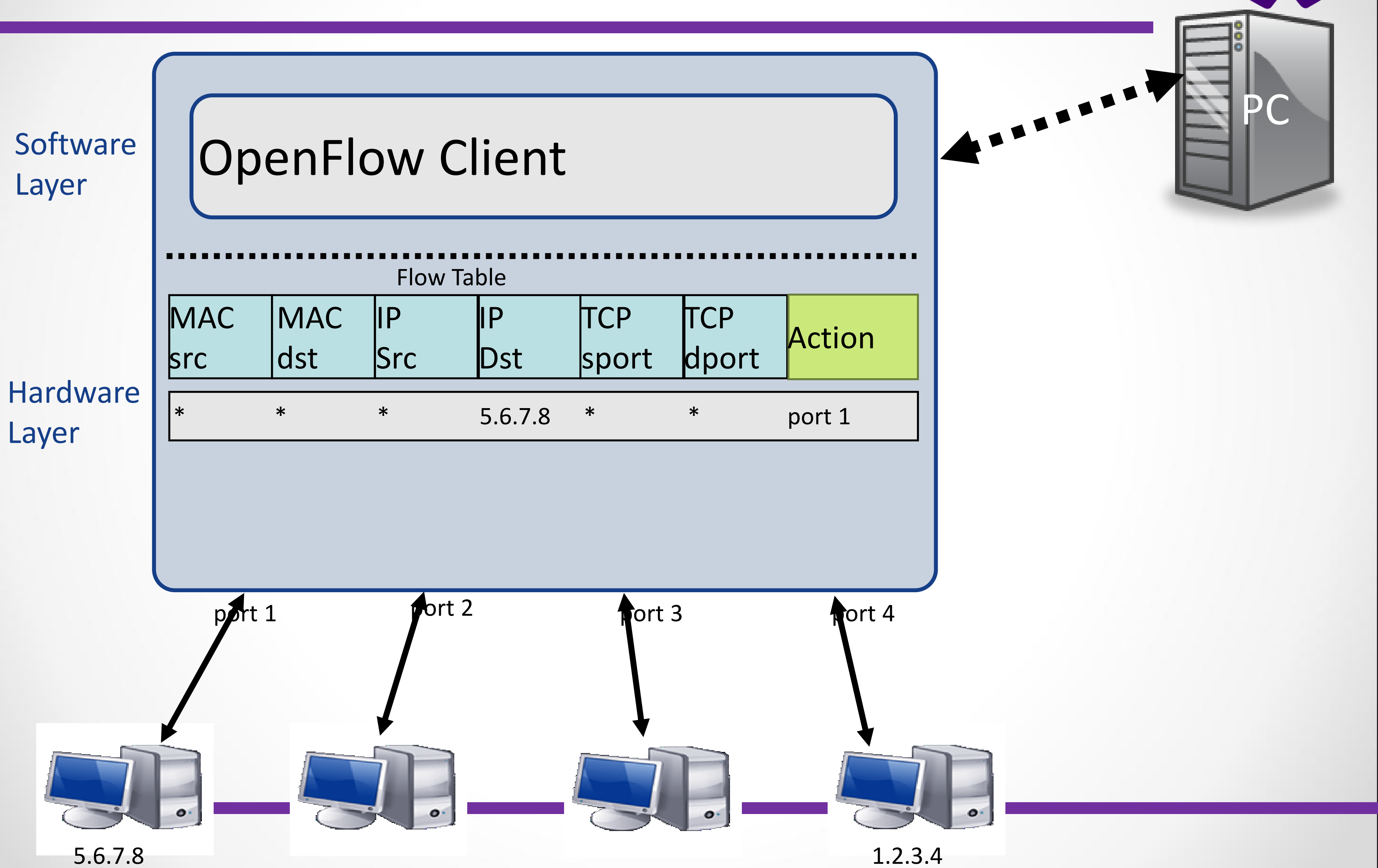
OpenFlow switch



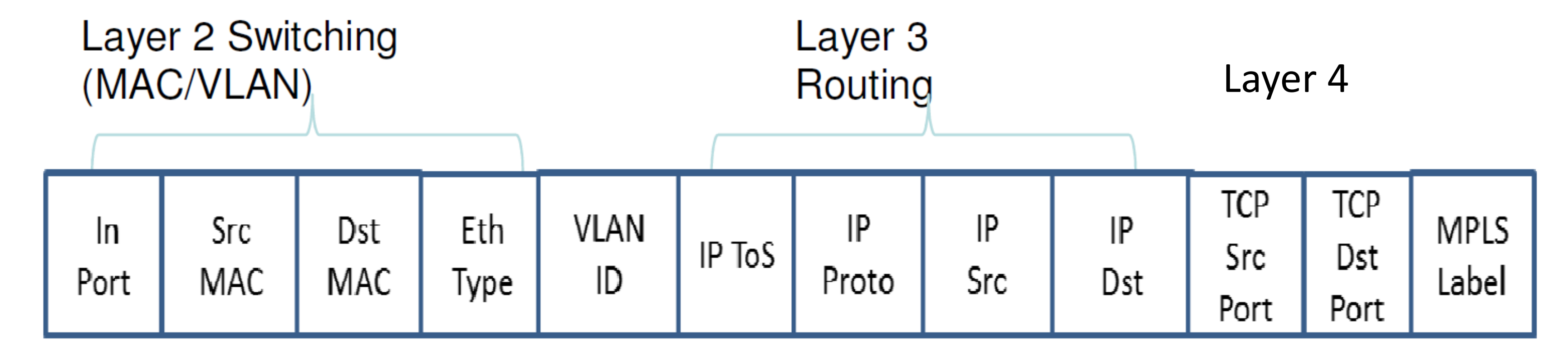
Openflow switch

- An Openflow switch (Ethernet switch) has an internal flow table.
  - If a packet matches an entry in the flow table, perform the actions (e.g. forward to port 10) according to the flow table.
  - If a packet does not match any entry in the flow table. Send it to the Openflow controller
    - The controller will figure out what to do with such packet
    - The controller will then respond to the switch, informing how to handle such a packet so that the switch would know how to deal with such packets next time.
    - For each flow, ideally the controller will be queried once.
- Openflow defines the standard interface to add and remove flow entries in the table.

OpenFlow Example



Flow switching and routing



- Each individual field + meta data
- Wild Card aggregation
  - E.g. IP-subnet: 192.168.*/24

OpenFlow Basics

Flow Table Entries

Rule	Action	Stats

Packet + byte counters

1. Forward packet to zero or more ports
2. Encapsulate and forward to controller
3. Send to normal processing pipeline
4. Modify Fields
5. Any extensions you add!

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP ToS	IP Prot	L4 sport	L4 dport

+ mask what fields to match

Examples

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

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

Examples

Routing

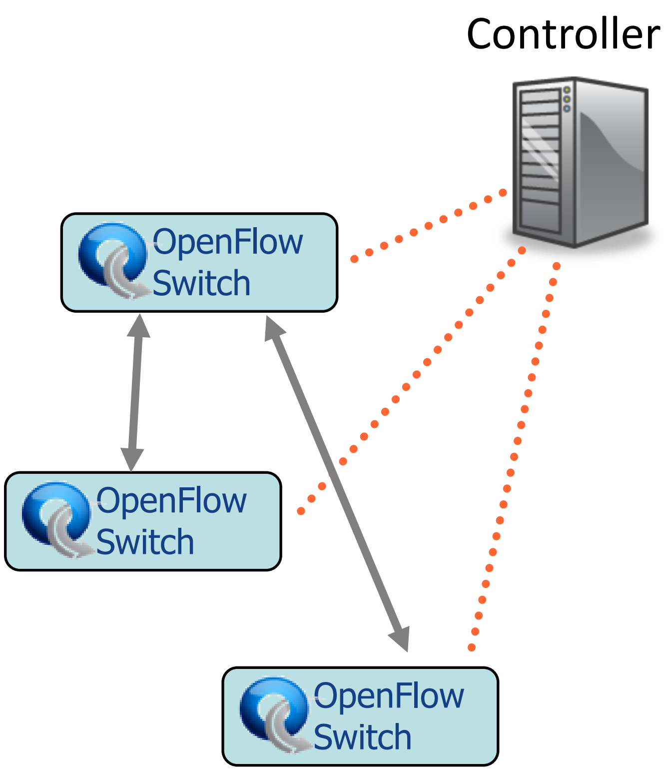
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

VLAN Switching

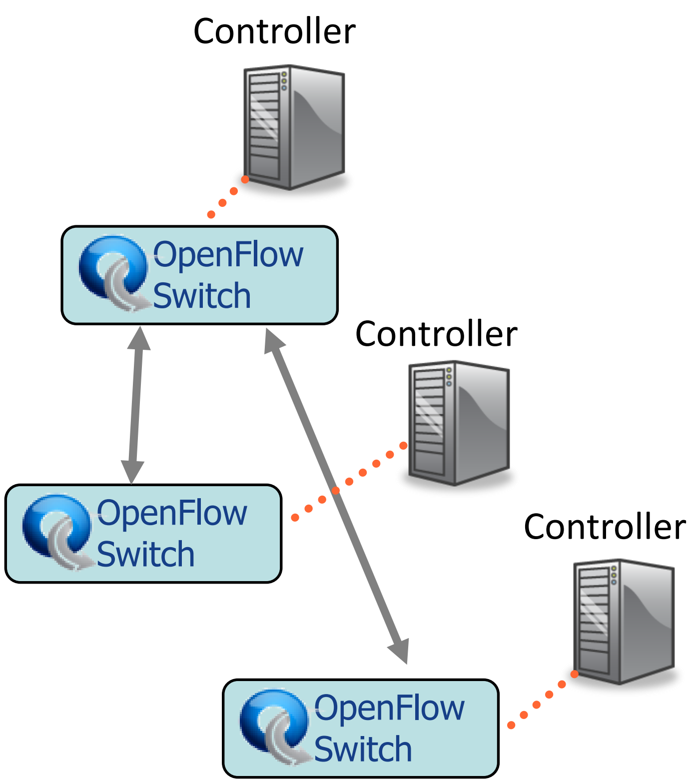
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

Centralized vs Distributed Control

Centralized Control



Distributed Control



Skoltech
Skolkovo Institute of Science and Technology

Flow Routing vs. Aggregation

Flow-Based

- Every flow is individually set up by controller
- Exact-match flow entries
- Flow table contains one entry per flow
- Good for fine grain control, e.g. campus networks

Aggregated

- One flow entry covers large groups of flows
- Wildcard flow entries
- Flow table contains one entry per category of flows
- Good for large number of flows, e.g. backbone

Reactive vs. Proactive (pre-populated)

Reactive

- First packet of flow triggers controller to insert flow entries
- Efficient use of flow table
- Every flow incurs small additional flow setup time
- If control connection lost, switch has limited utility

Proactive

- Controller pre-populates flow table in switch
- Zero additional flow setup time
- Loss of control connection does not disrupt traffic
- Essentially requires aggregated (wildcard) rules

Openflow specifications

- From 1.0.0 to 1.5.0 (1.6 not public yet)
- Briefly introduce concepts in versions 1.0.0 to 1.2.0

Openflow 1.0 concepts

- Ports and Port queues
- Flow table
- Packet matching
- Actions and packet forwarding
- Messaging between controller and switch

Skoltech
Skolkovo Institute of Science and Technology

Open Flow Protocol Messages

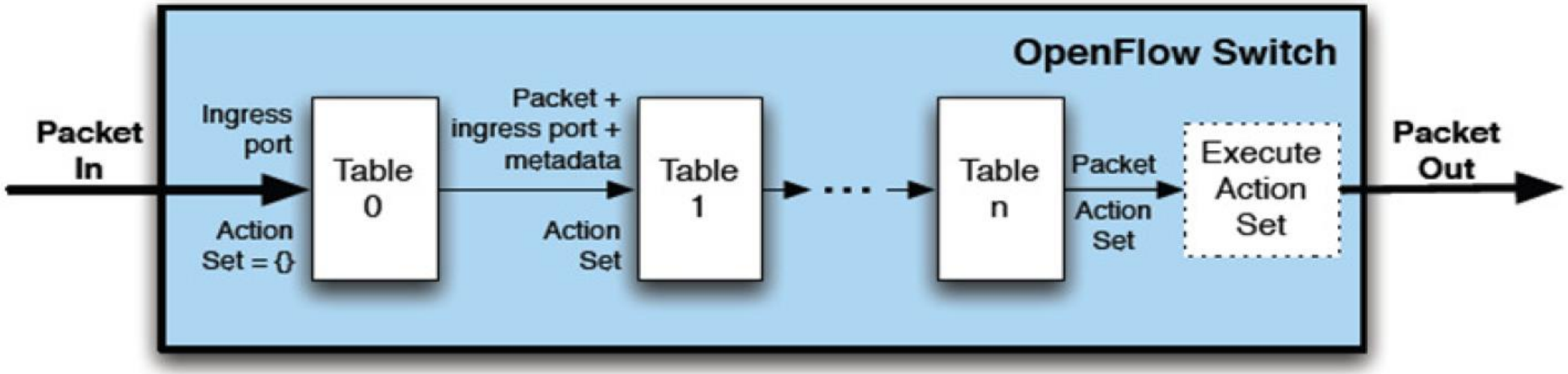
- Controller-to-switch: from the controller to manage or inspect the switch state
  - Features, config, modify state, read state, packet-out, etc
- Asynchronous: send from switch without controller soliciting
  - Packet-in, flow removed/expired, port status, error, etc
- Symmetric: symmetric messages without solicitation in either direction
  - Hello, Echo, etc.

Openflow 1.1 concepts

- Multiple flow tables
- Groups
- MPLS and VLAN tag support
- Virtual ports
- Controller connection failure

Pipeline processing (in 1.1)

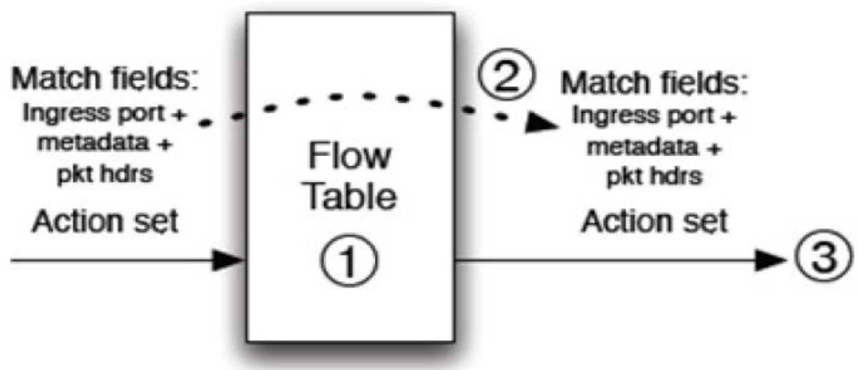
- A switch can have multiple flow tables that are matched in a pipeline fashion.



(a) Packets are matched against multiple tables in the pipeline

Skoltech
Skolkovo Institute of Science and Technology

Per table packet processing



① Find highest-priority matching flow entry

② Apply instructions:
  i. Modify packet & update match fields (apply actions instruction)
  ii. Update action set (clear actions and/or write actions instructions)
  iii. Update metadata

③ Send match data and action set to next table

(b) Per-table packet processing

Groups

- Group table: entries and actions
  - To refine flooding
  - Support multicast
  - As a base for rules that apply to multiple flows

1.2.0 concepts

- Extensible match support
- Extensible set_field packet-rewrite support
- IPv6
- Multiple controller enhancements

- Later versions of Openflow specification supports more necessary functions.

Skoltech
Skolkovo Institute of Science and Technology

Thanks for your attention!

Vasily Pashkov
pashkov@lvk.cs.msu.su