

Computer Networks

COL 334/672

Software Defined Networking

Tarun Mangla

Slides adapted from KR

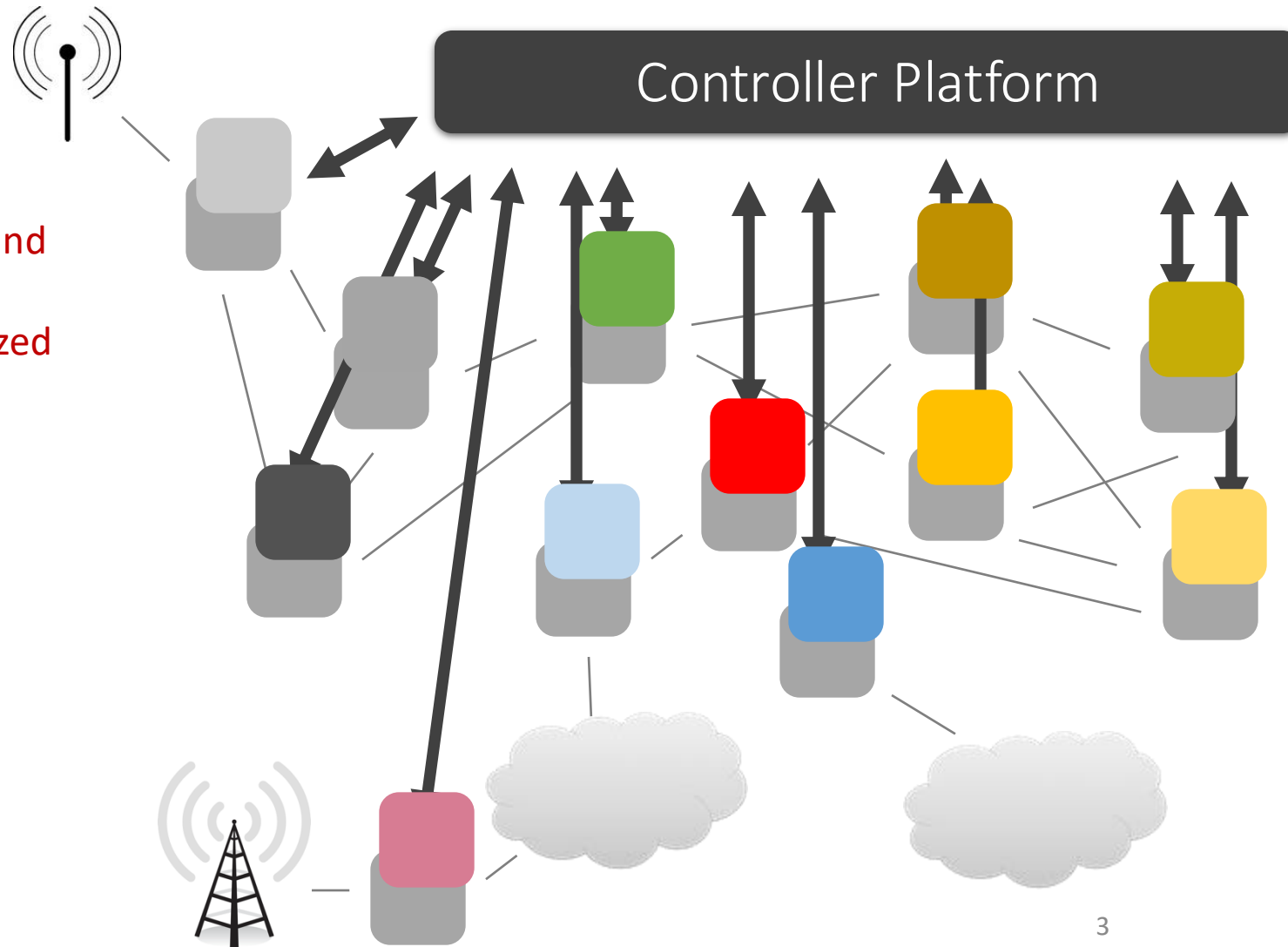
Sem 1, 2024-25

Quiz

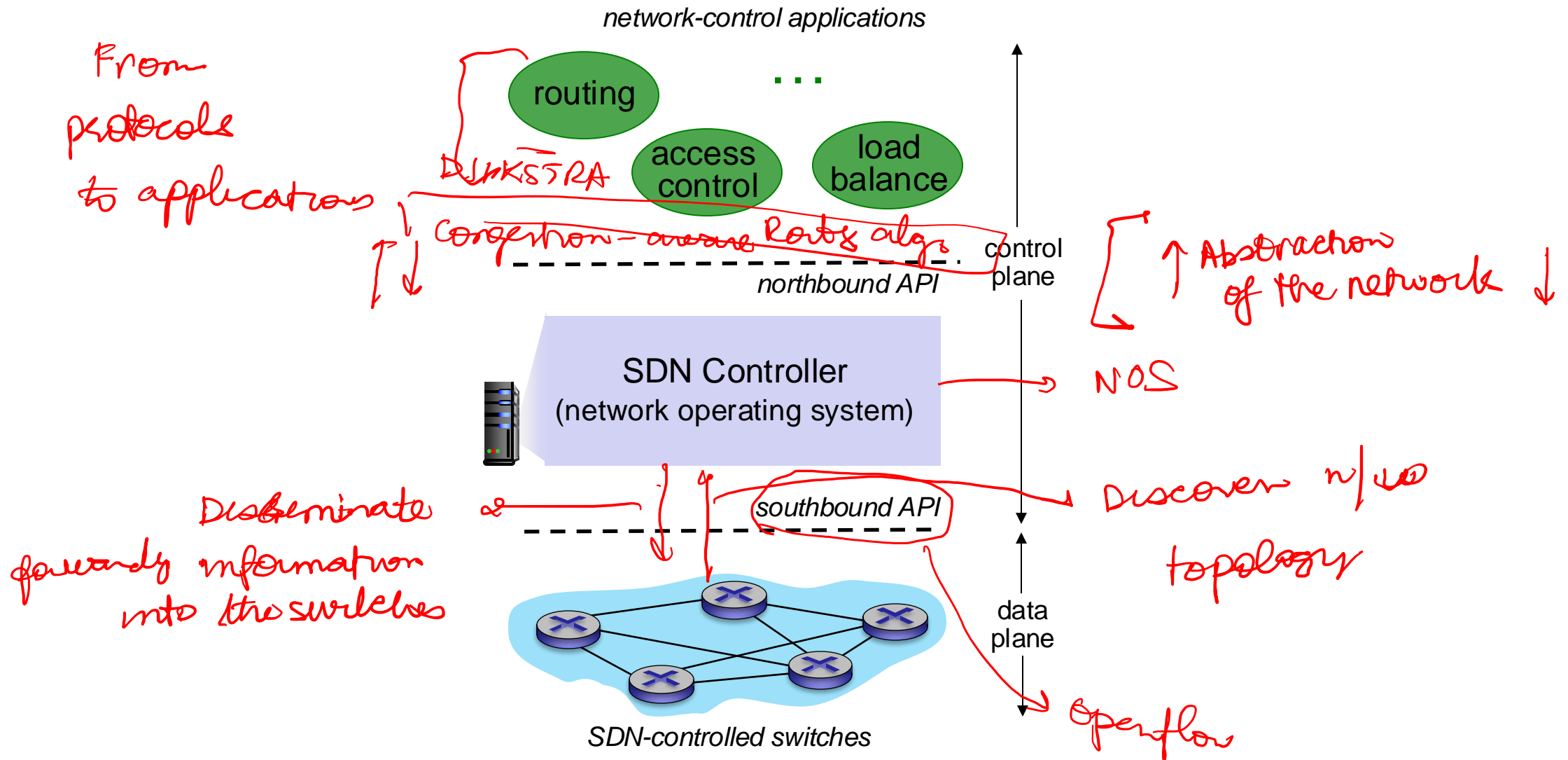
Password: openflow

Software-defined Networking

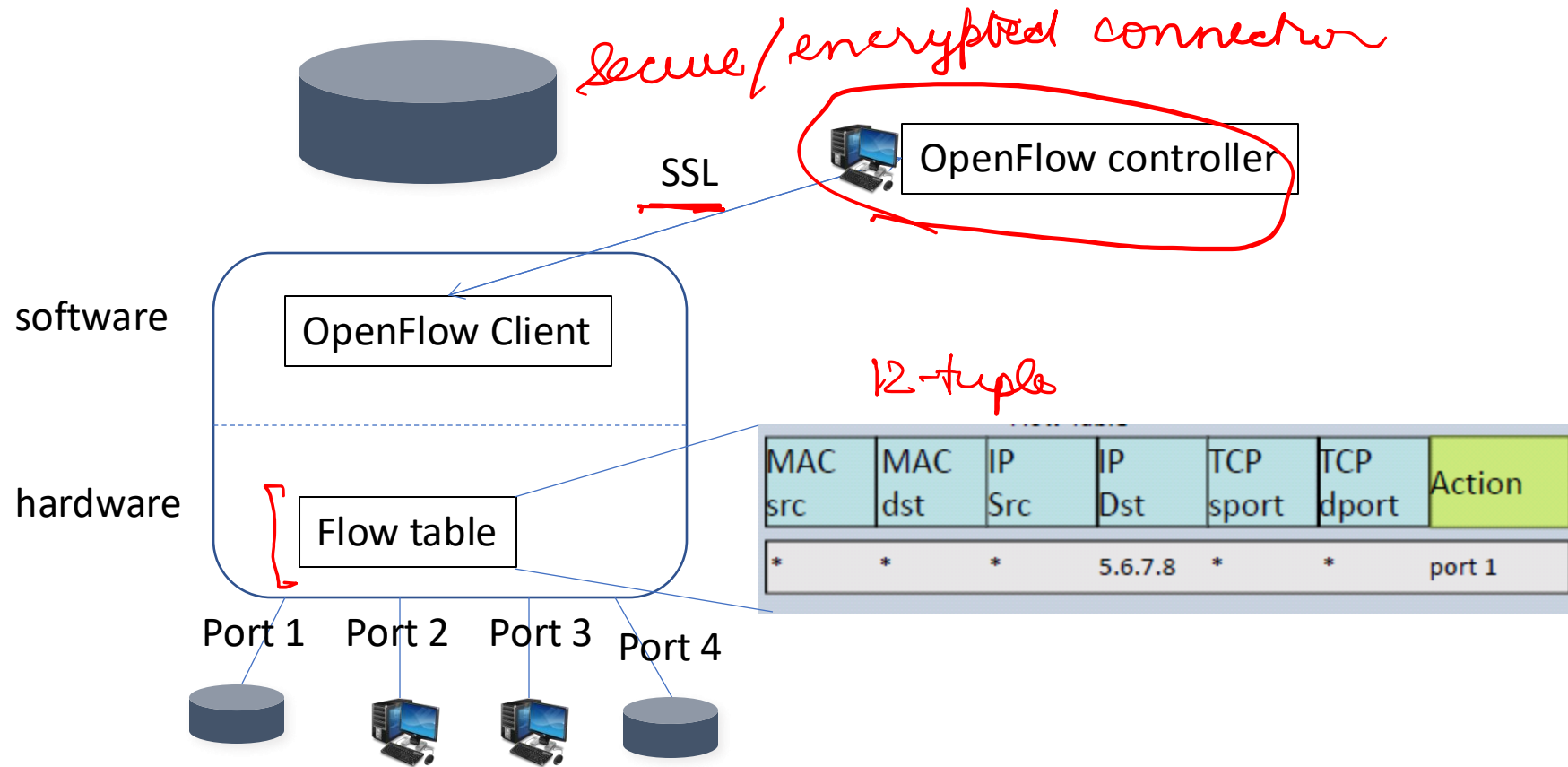
- Decouple control and data plane
- (Logically) Centralized Controller



Software-defined Network Architecture



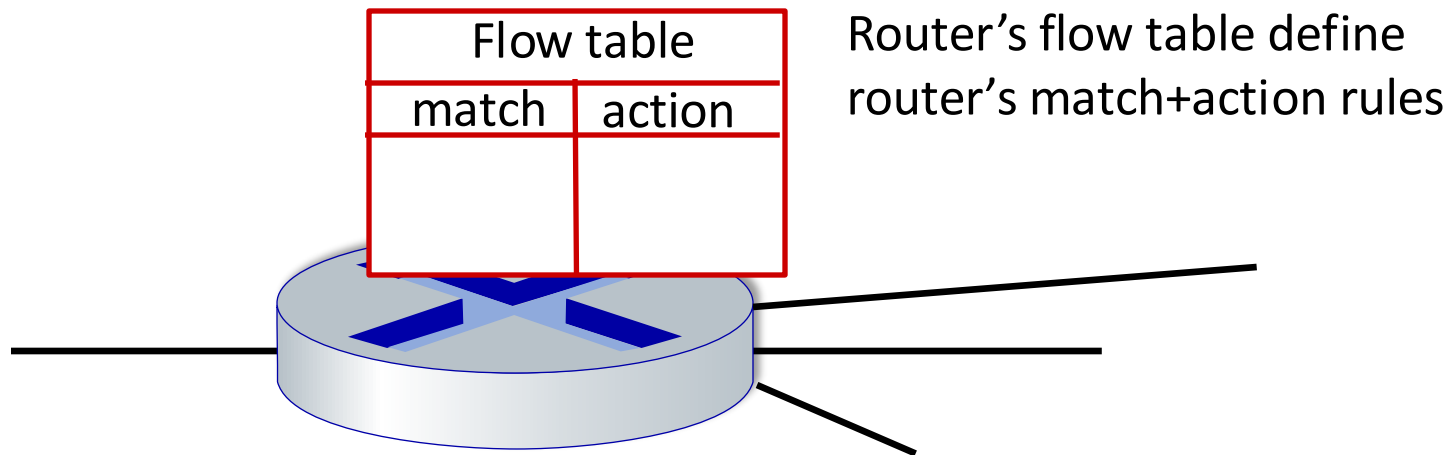
OpenFlow switch



OpenFlow: Flow table abstraction

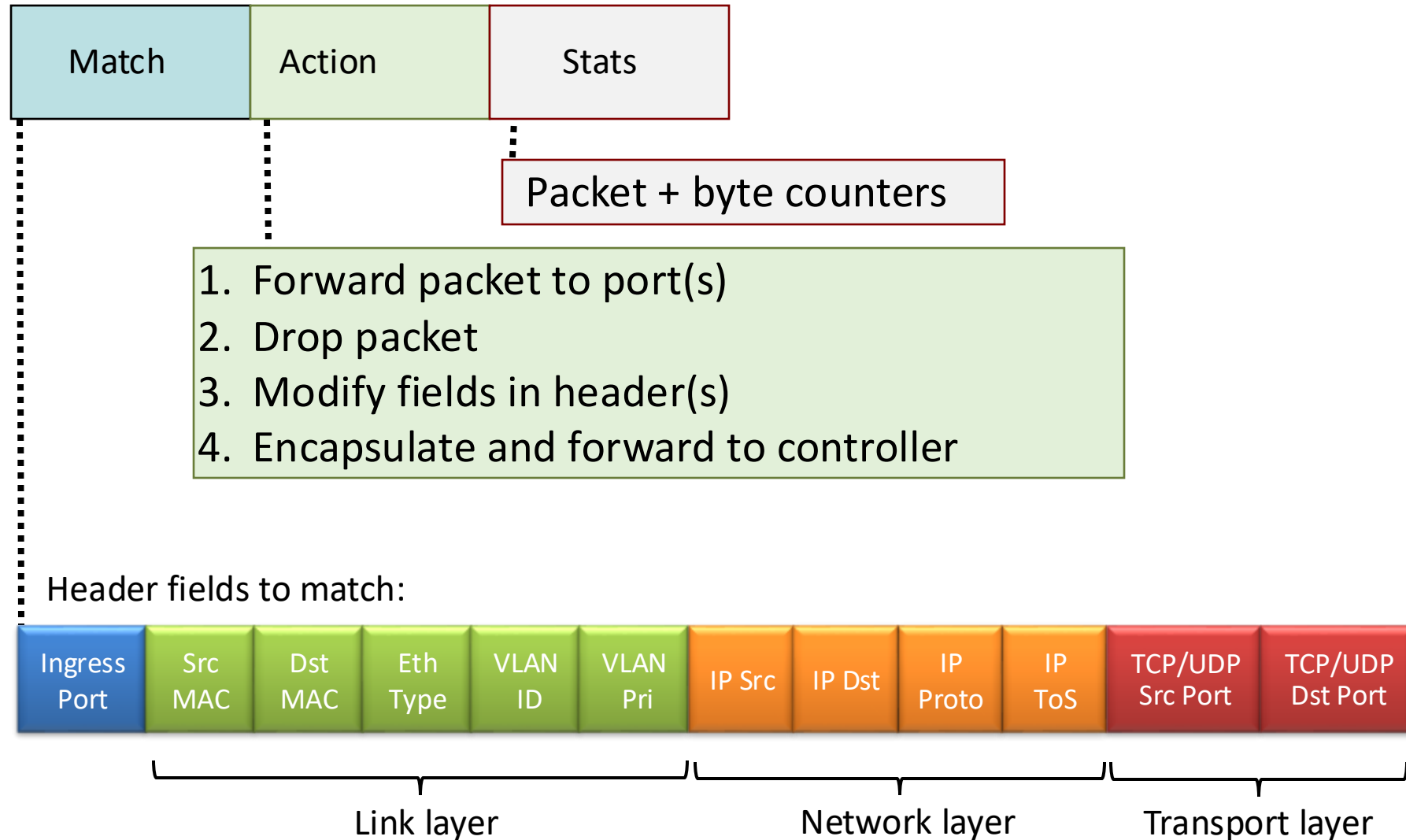
- **flow**: defined by header field values (in link-, network-, transport-layer fields) *12-tuple*
- **generalized forwarding**: simple packet-handling rules
 - **match**: pattern values in packet header fields *optional in first version*
 - **actions**: for matched packet: drop, forward, modify, matched packet or send matched packet to controller
 - **priority**: disambiguate overlapping patterns
 - **counters**: #bytes and #packets

Flow rule:
① dest port : 22 Forward(1)
② dest IP = X, Forward(2)



Deep packet inspection

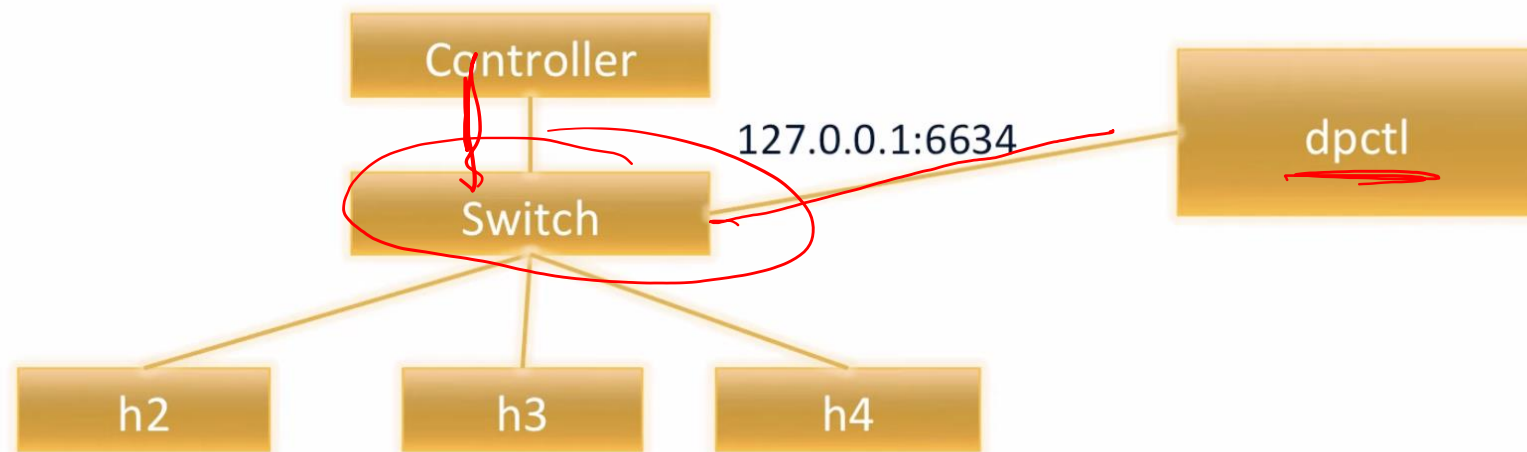
OpenFlow: flow table entries



OpenFlow Demo

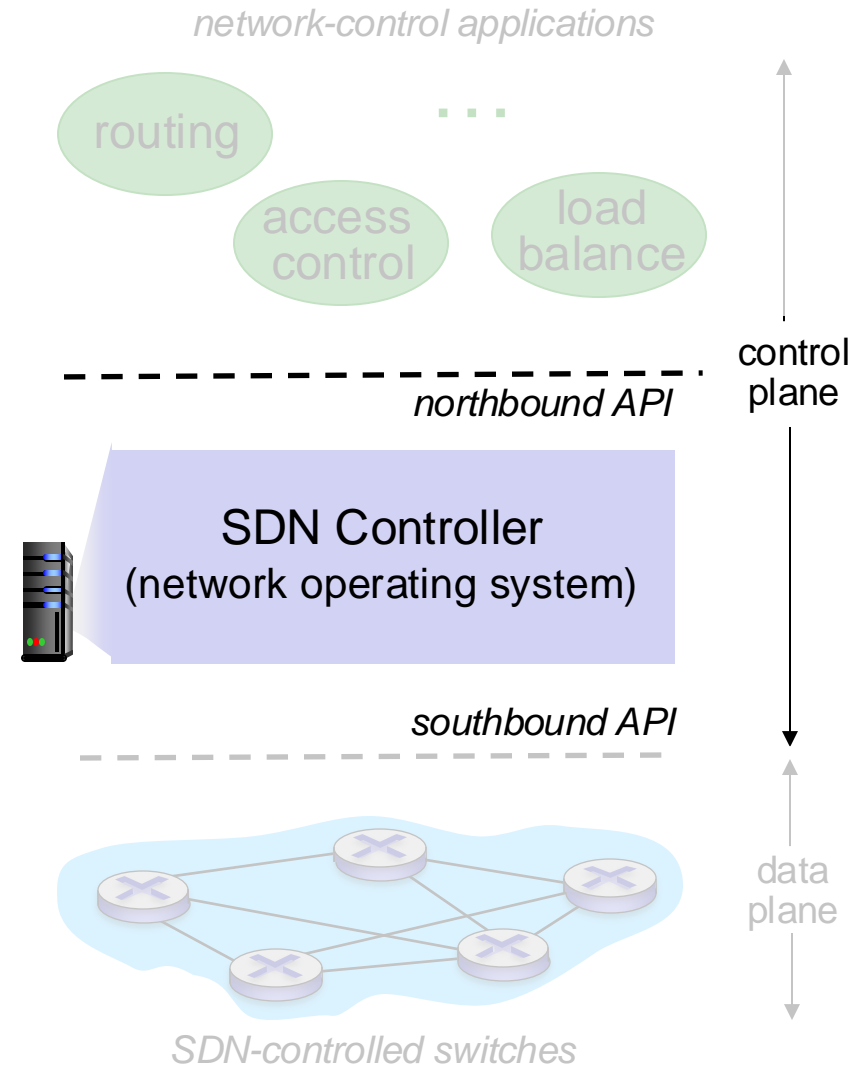
Simulation vs *Real-world testing*

- Mininet: a network emulation and prototyping platform
 - A virtual network environment that can run on a single PC
 - Built in OpenFlow features
- Openflow switch: dpctl control channel



SDN Controller (Network OS)

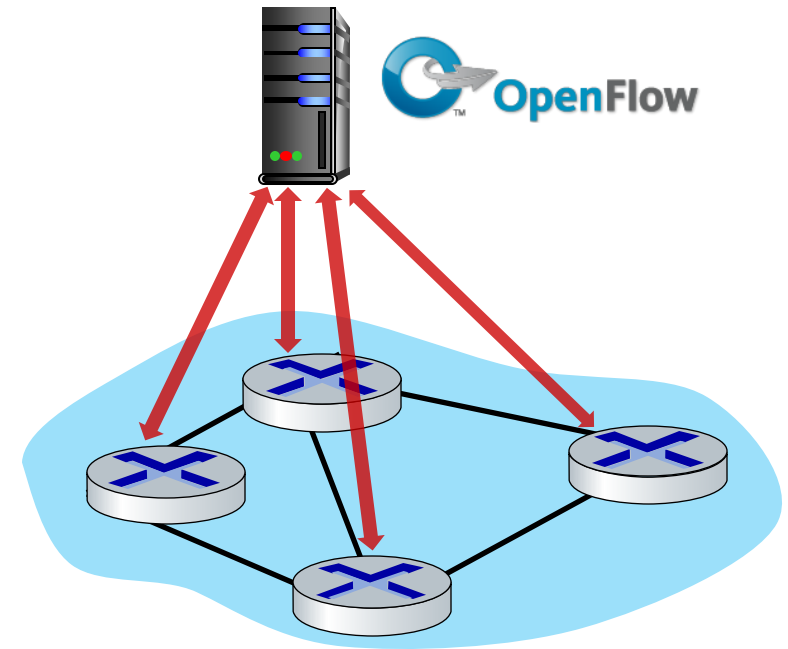
- maintains network state information
- interacts with network control applications “above” via northbound API
- interacts with network switches “below” via southbound API
- implemented as distributed system for performance, scalability, fault-tolerance, robustness



OpenFlow Messages

- TCP used to exchange messages
 - optional encryption
- Three classes of OpenFlow messages:
 - controller-to-switch
 - asynchronous (switch to controller)
 - symmetric (misc.)
- distinct from OpenFlow API
 - API used to specify generalized forwarding actions

OpenFlow Controller

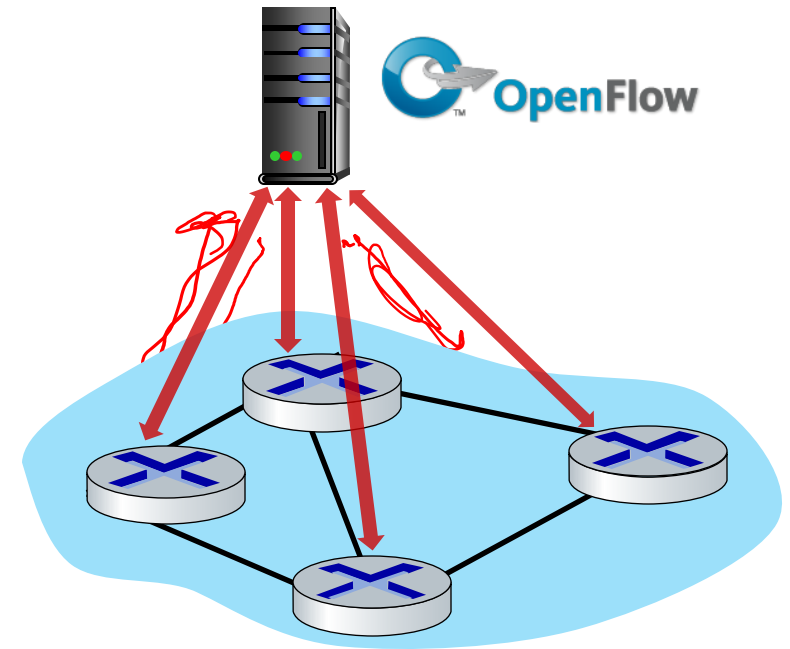


OpenFlow: controller-to-switch messages

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 of specific switch port

OpenFlow Controller

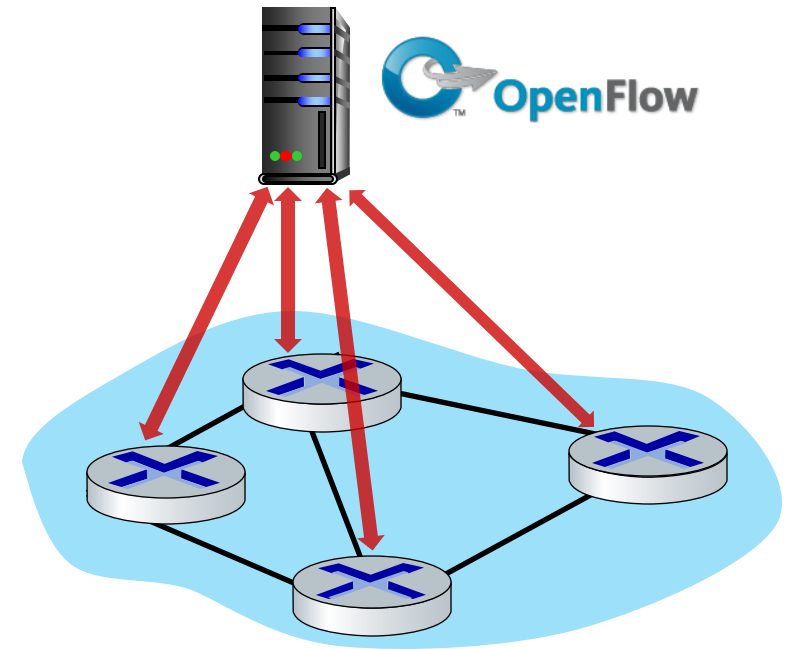


OpenFlow: switch-to-controller messages

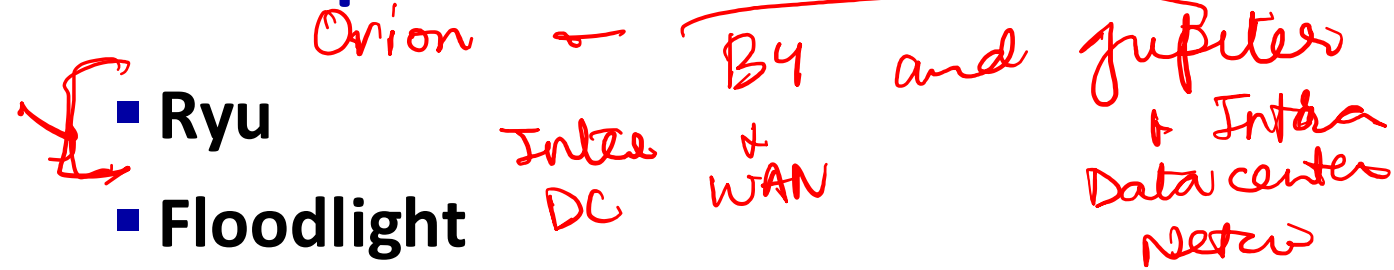
Key switch-to-controller messages

- *packet-in*: transfer packet (and its control) to controller. See packet-out message from controller
- *flow-removed*: flow table entry deleted at switch
- *port status*: inform controller of a change on a port.

OpenFlow Controller



Example of Controllers



Floodlight

- Ryu
- Floodlight
- NOX/POX

- OpenDayLight (ODL)
- Open Network Operating System (ONOS)



OPEN DAYLIGHT

- Pyretic
- Frentic
- Procera

SQL-like abstraction

to control things in the data plane

Choosing the right SDN controller?

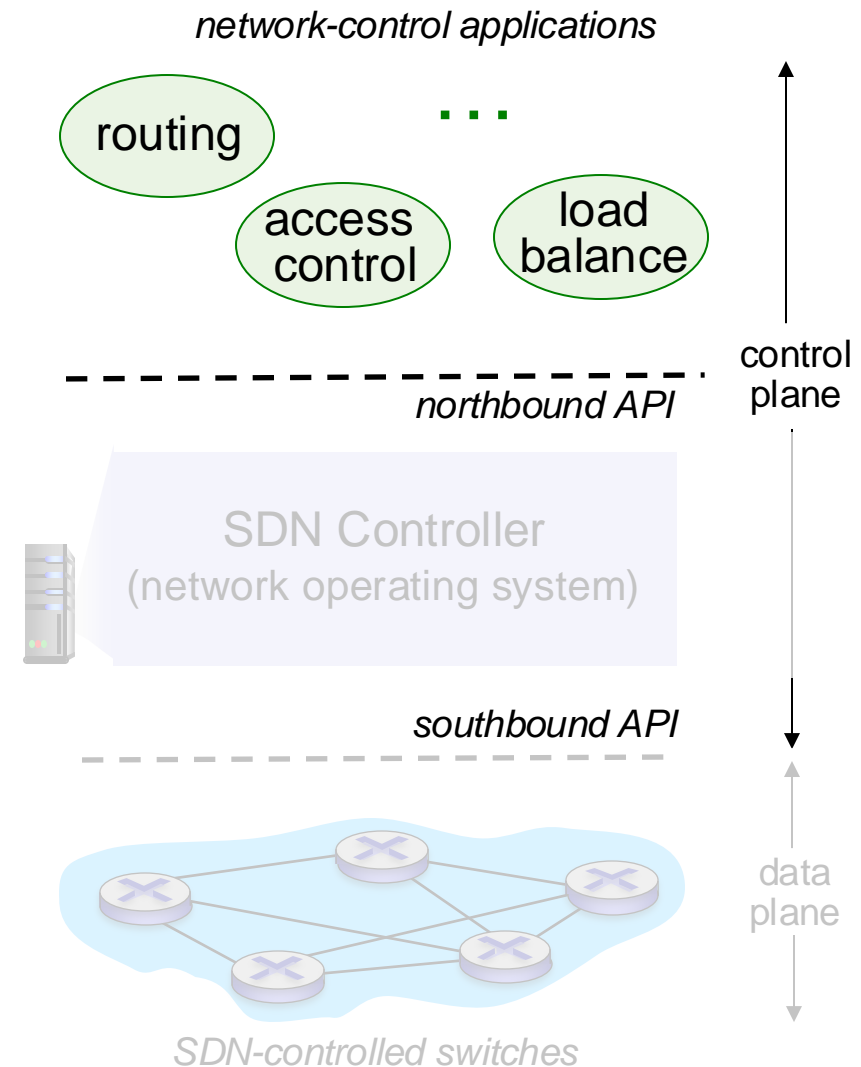
Use case

- Learning curve, programming language
- Focus (Southbound API, Northbound API)
- Community support

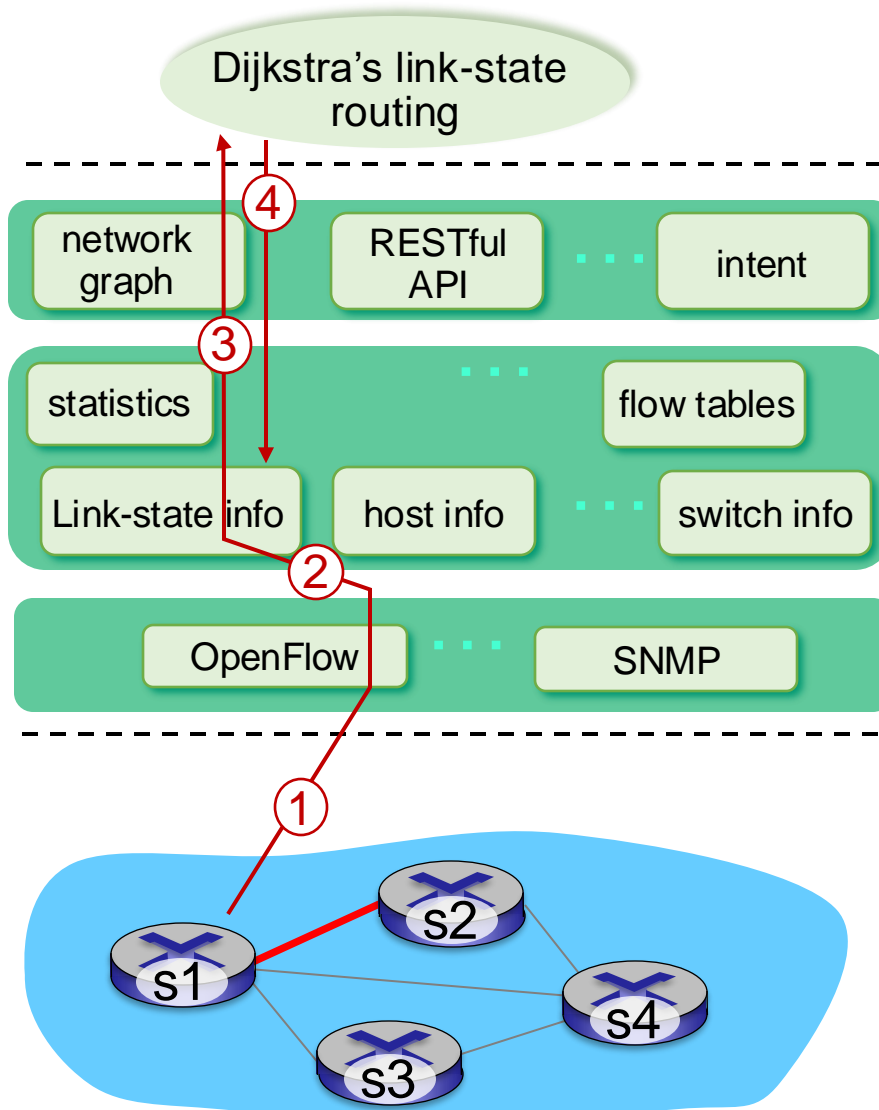
Ryu Demo

Software defined networking (SDN)

- operators don't "program" switches by creating/sending OpenFlow messages directly.
- Instead use higher-level abstraction at controller
- "brains" of control: implement control functions using lower-level services, API provided by SDN controller
- *unbundled*: can be provided by 3rd party: distinct from routing vendor, or SDN controller

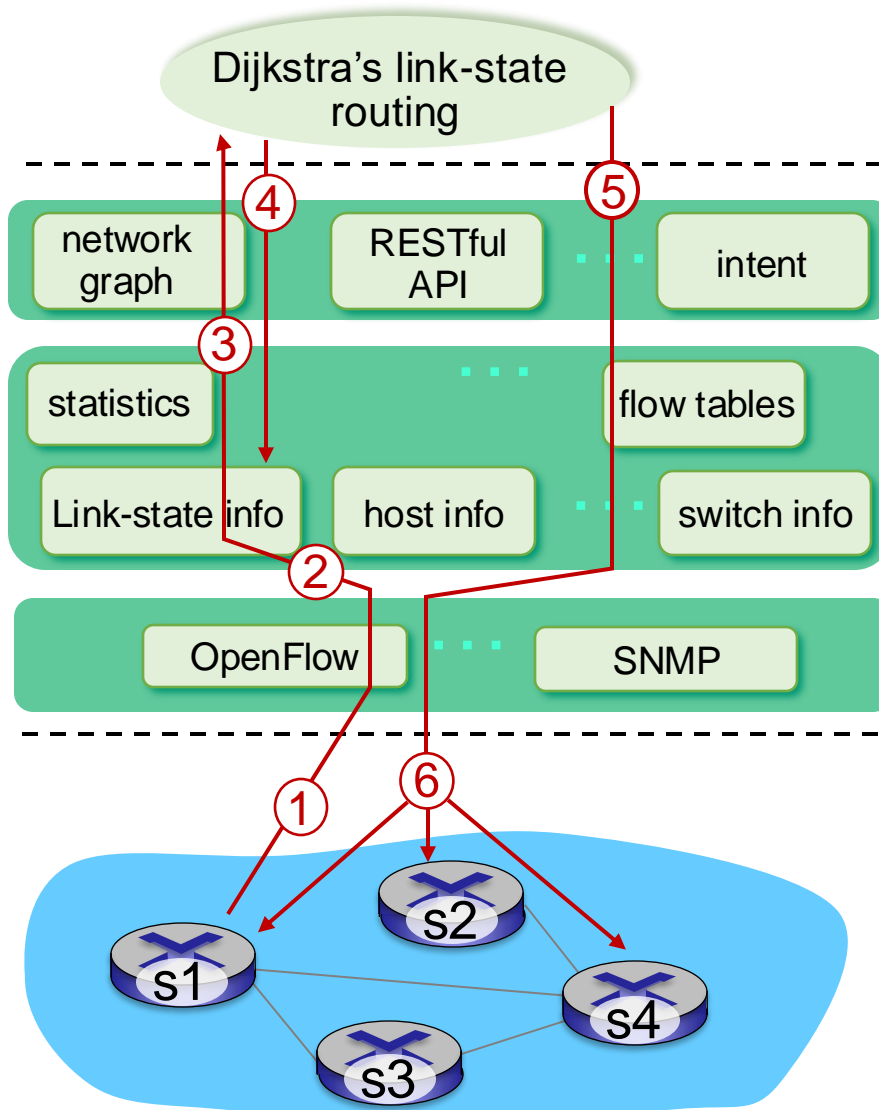


SDN: control/data plane interaction example



- ① S1, experiencing link failure uses 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

SDN: control/data plane interaction example



- 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

SDN: Key Challenges

- Hardening the control plane
 - Scalability
 - Reliability
 - Consistency
 - Security
- Internet-scaling: beyond a single AS (?)

Attendance

