Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Institute of Computer Science
Chair of Communication Networks
Prof. Dr.-Ing. P. Tran-Gia

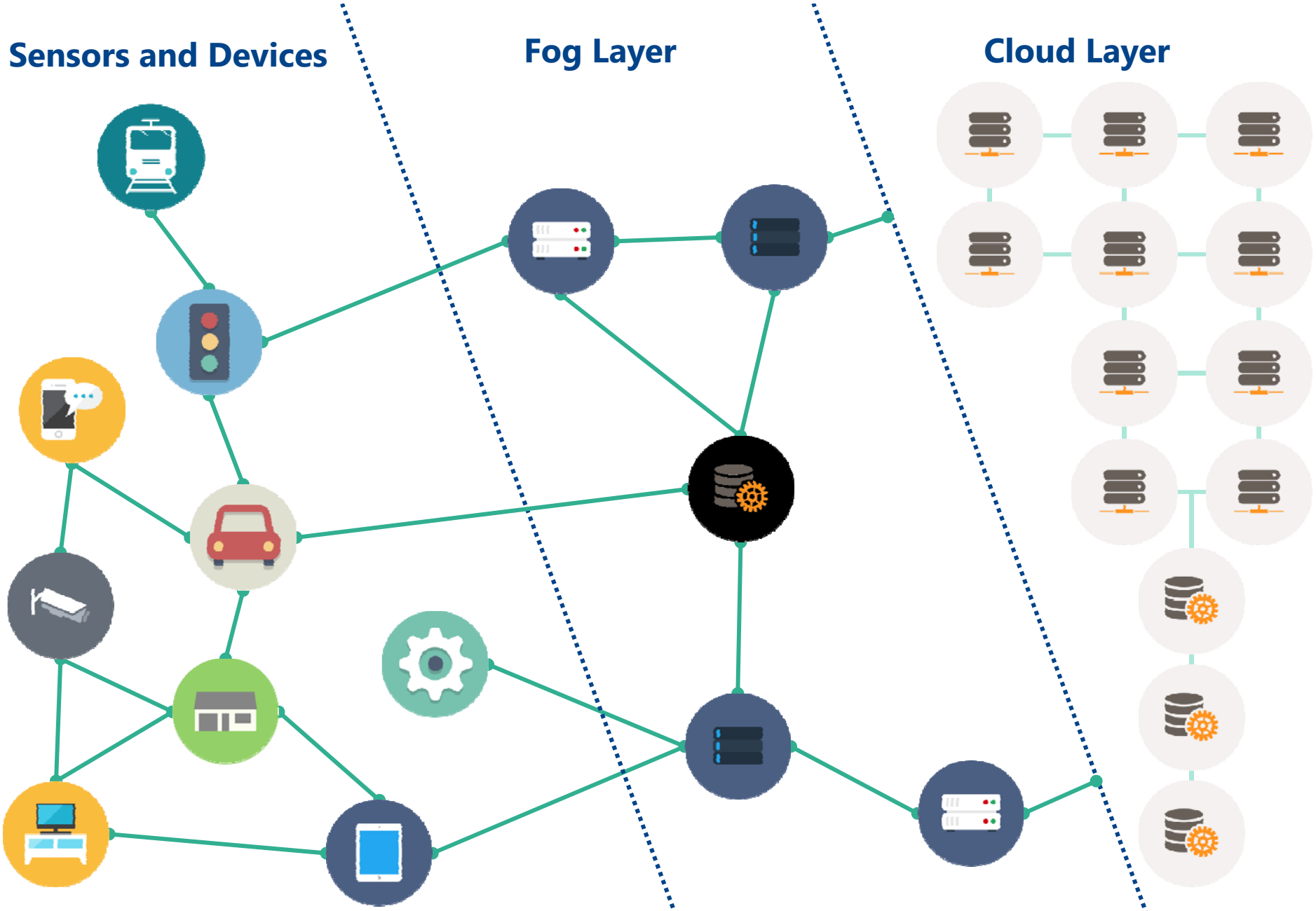
SENDATE
PLANETS

TableVisor 2.0

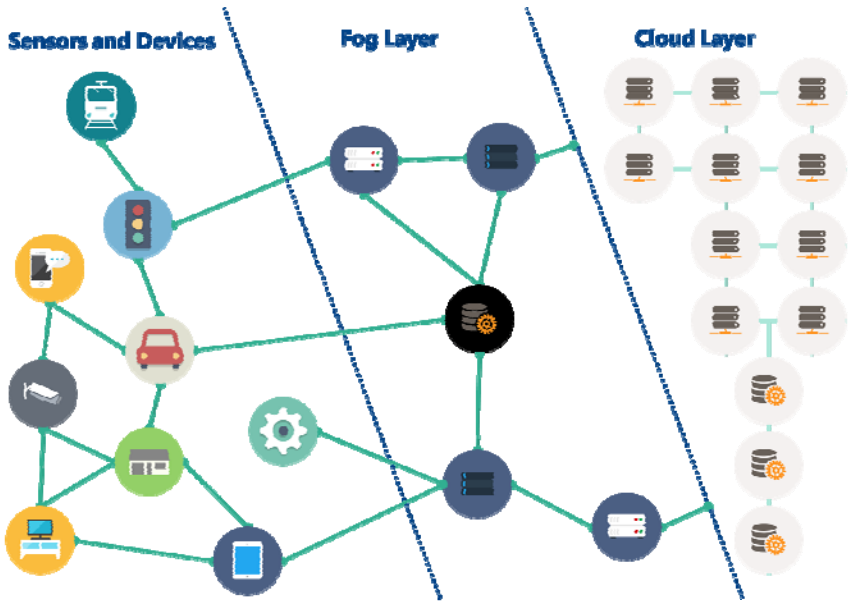
Hardware-independent Multi Table Processing

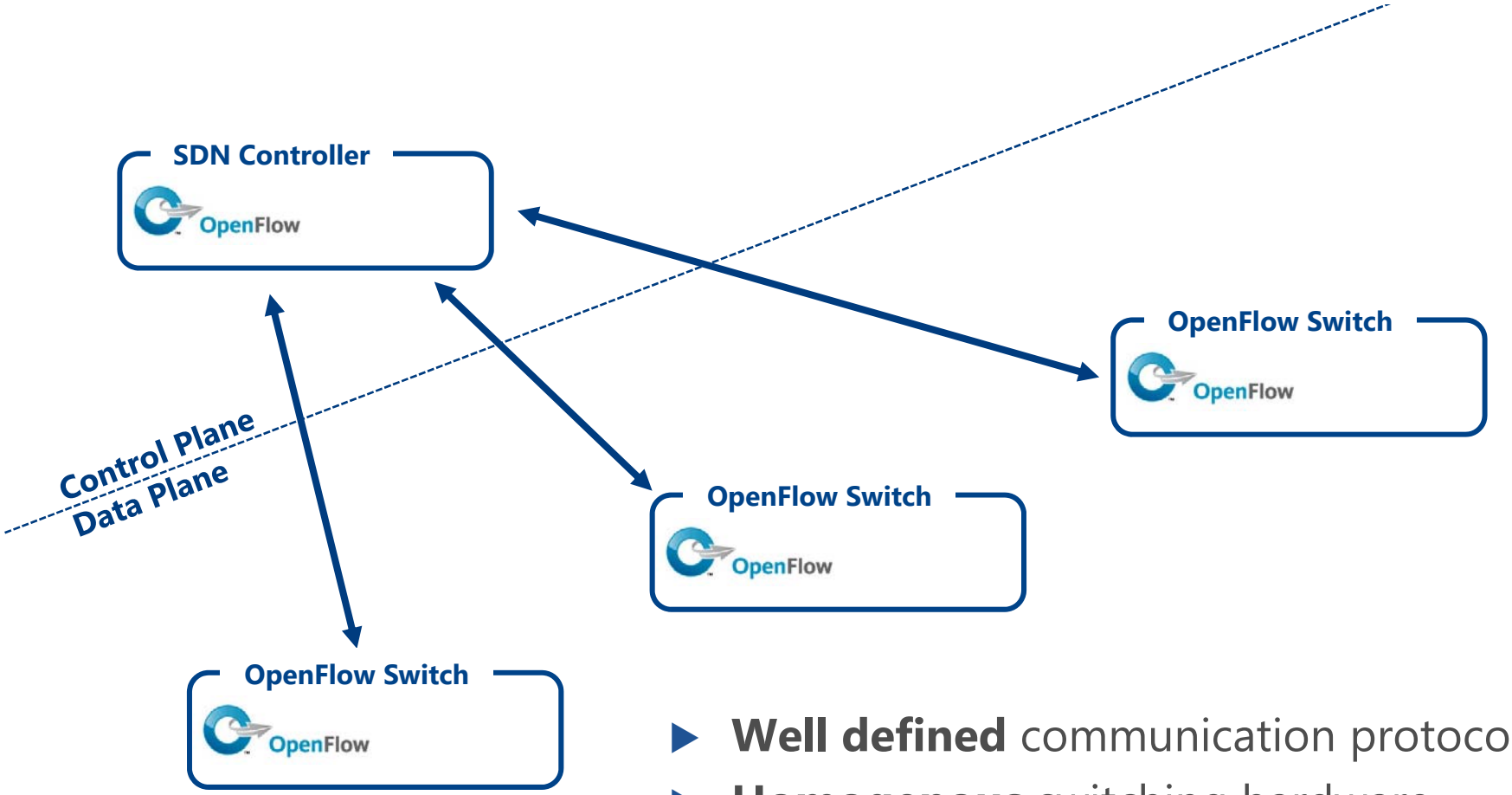
Stefan Geißler, Thomas Zinner (University of Wuerzburg)

comnet.informatik.uni-wuerzburg.de

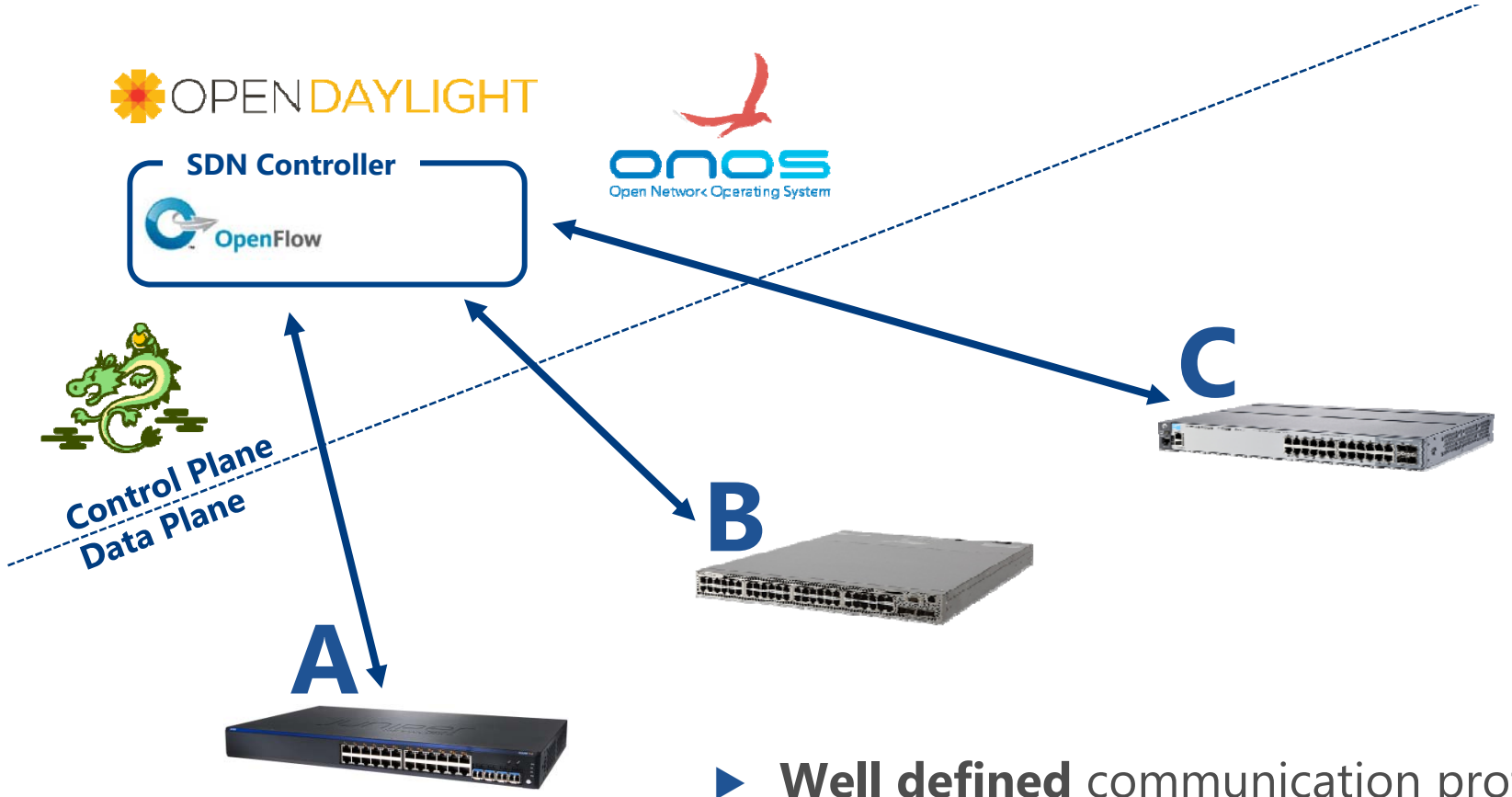

Sensors and Devices
Fog Layer
Cloud Layer

- Sensors and Devices
  - Highly **heterogeneous, shifting environment** of devices
  - Different applications with **different requirements**
  - Constantly **changing topology**
- Fog Layer
  - **Distributed** compute, storage and network **resources**
  - **Heterogeneous** environment
  - **Limited resource** availability
- Cloud Layer
  - Centralized, **well structured environment**
  - Basically **unlimited resources**

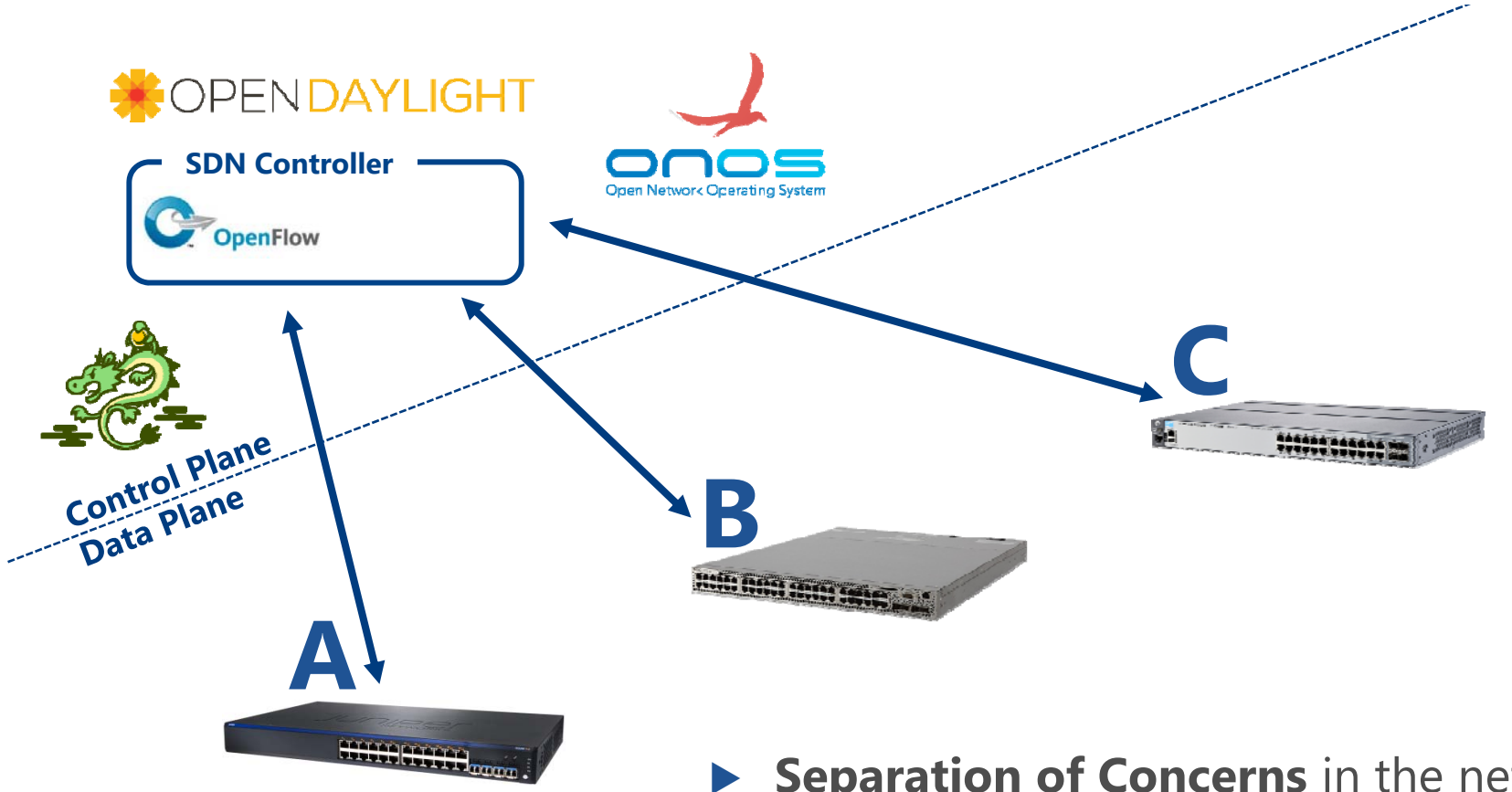




- **Well defined** communication protocol
- **Homogenous** switching hardware
  - Provide **full** OpenFlow support
  - **Respect** OpenFlow specification
  - Support **newest** OpenFlow version



- **Well defined** communication protocol
- **Heterogeneous** vendor specific devices
  - Provide **partial** OpenFlow support
  - **Sometimes** respect OpenFlow specification
  - Support **different** OpenFlow **versions** or a work with a **completely different protocol**



- **Separation of Concerns** in the network
  - Controller
  - Switches
- **Feature Limitation** of switching hardware
  - Unsolvable use cases
  - Vendor dependence

OPENDAYLIGHT

SDN Controller

OpenFlow

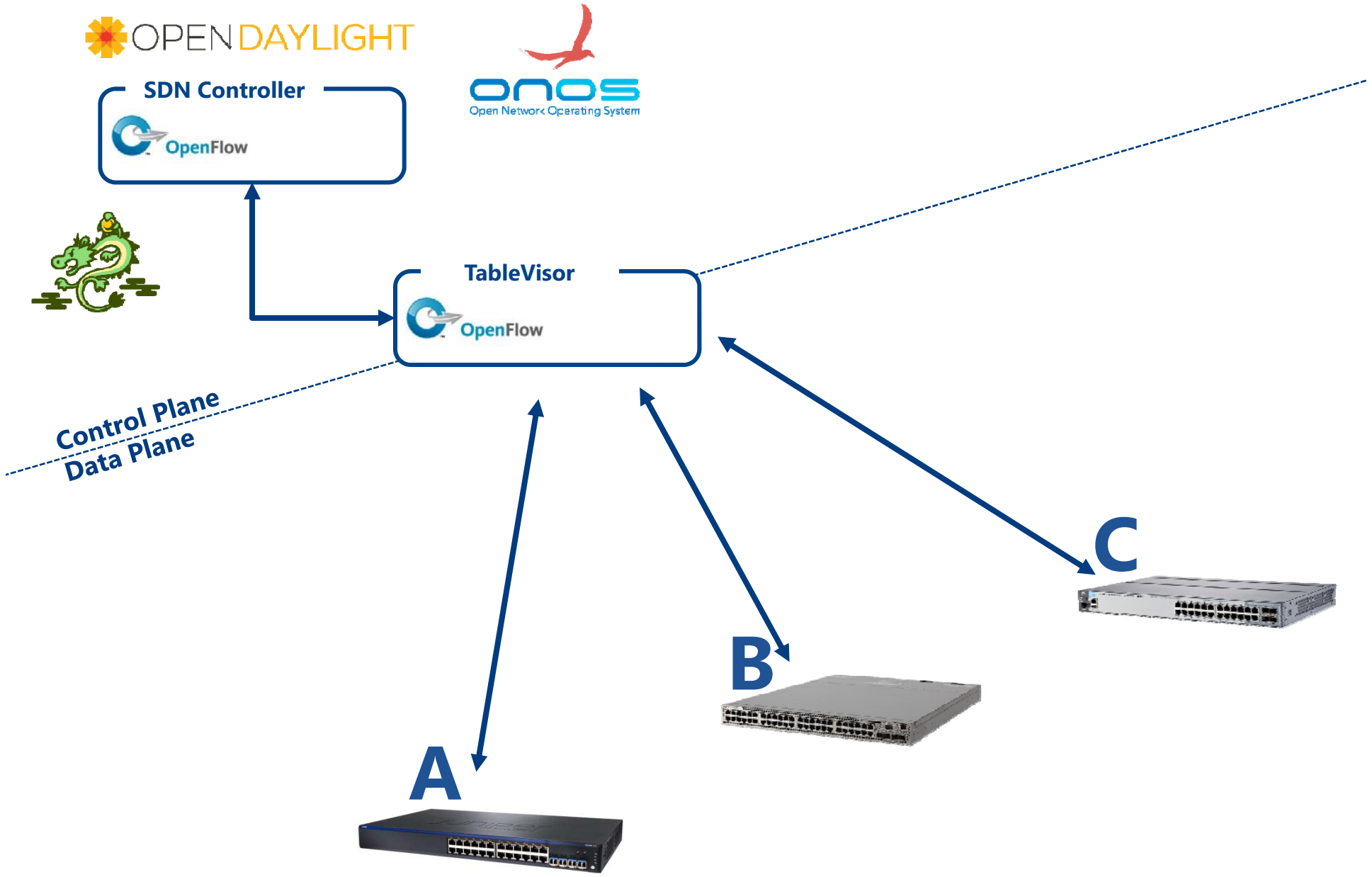
ONOS
Open Network Operating System

Control Plane
Data Plane

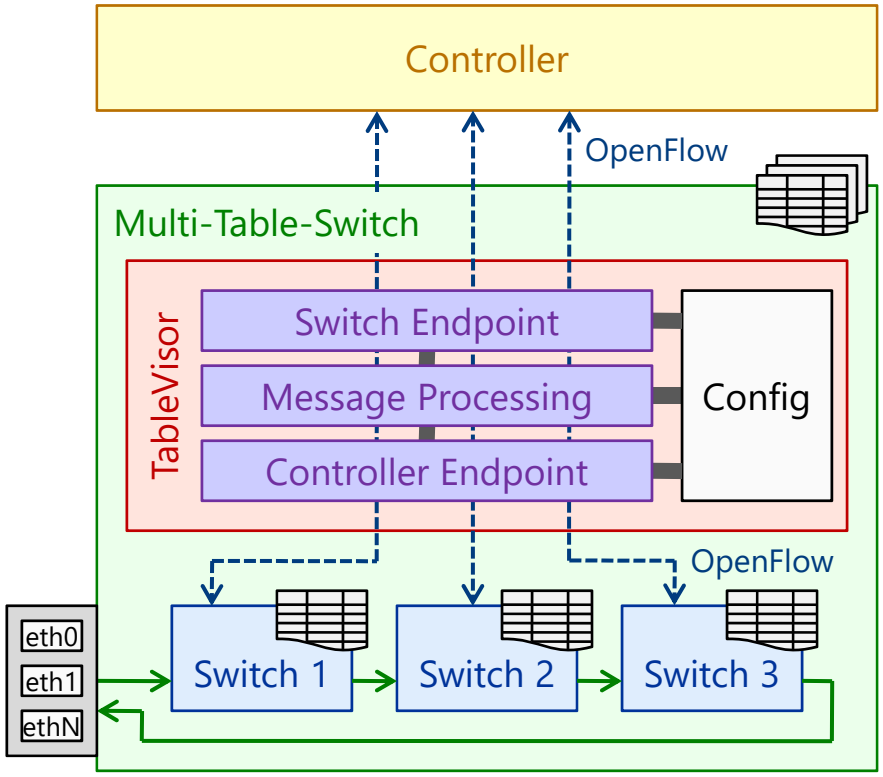
C

B

A


OPENDAYLIGHT
SDN Controller
OpenFlow
ONOS
Open Network Operating System
TableVisor
OpenFlow
Control Plane
Data Plane
A
B
C

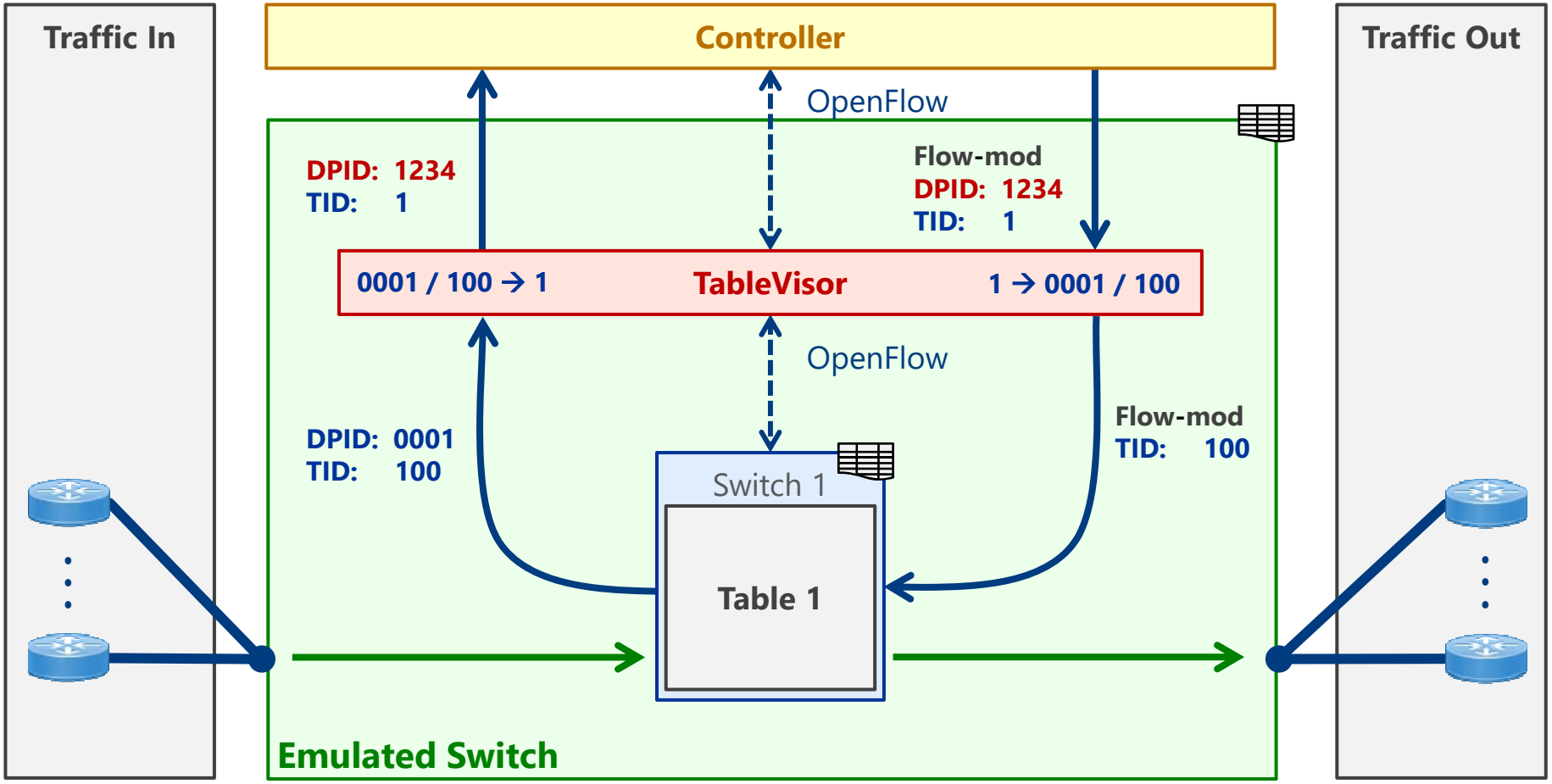
TableVisor



GitHub
lsinfo3/TableVisor

- **Stateless** OpenFlow Proxy Layer
- **No modifications** required
- Architecture
  - Switch Endpoint
  - Message Processing
  - Controller Endpoint
- **Translates** OpenFlow Messages
- **Emulated** hardware switch
  - Multi table forwarding pipeline
  - Combined hardware functionality
  - Extension of flow table capacity

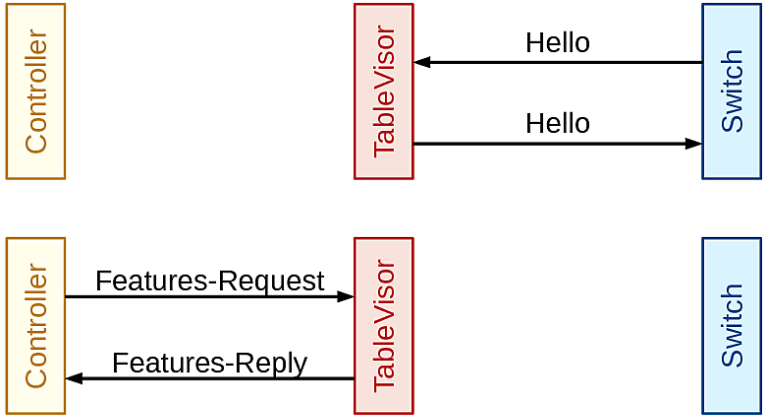
Single Switch Example



Message Processing

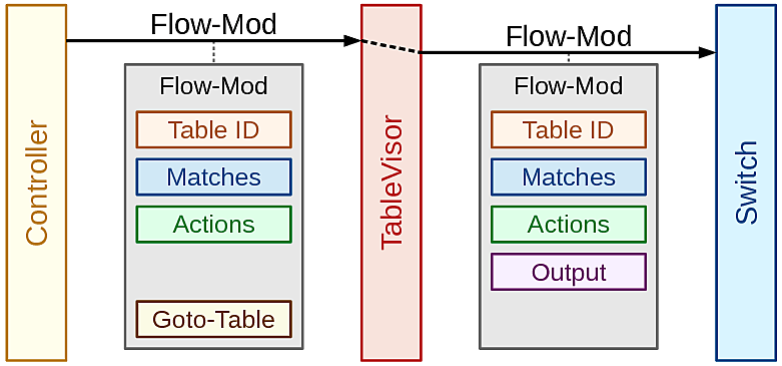
Type I Messages

- Simple **Request-Response**
  - Hello
  - Feature-Request
- No message forwarding

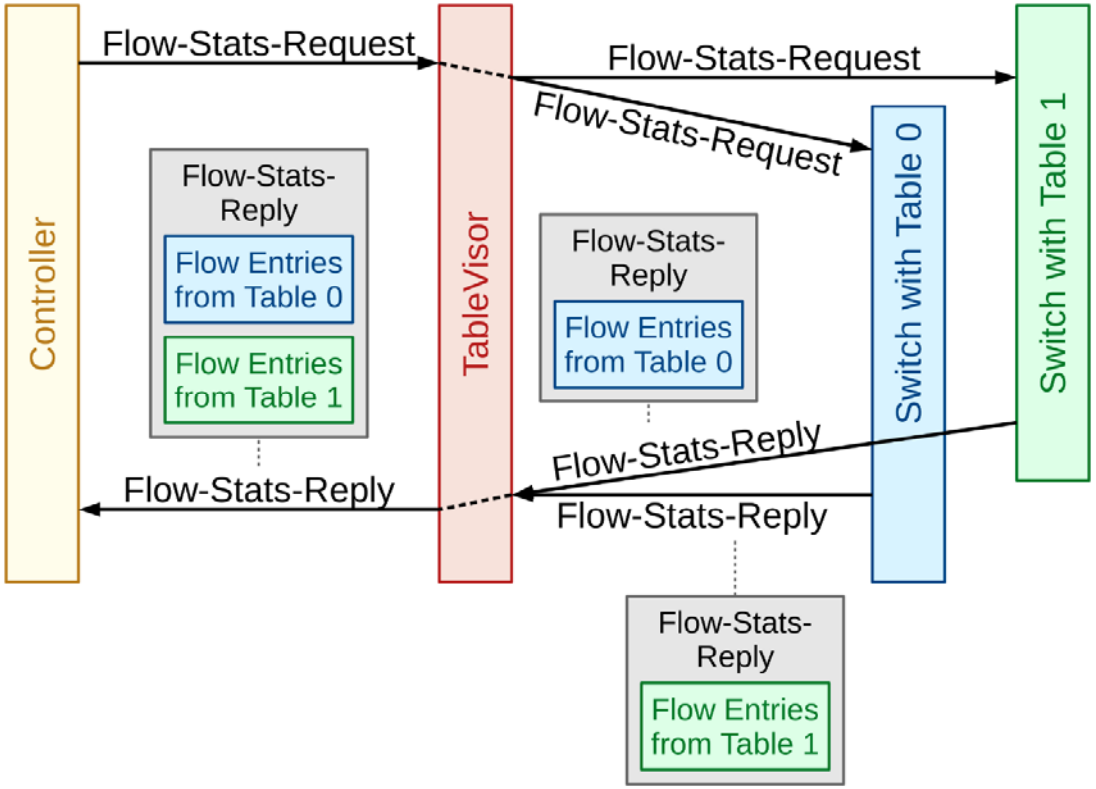


Type II Messages

- Message **modification** required
  - Flow-Mod
  - Flow-Stats
- Modified message is forwarded

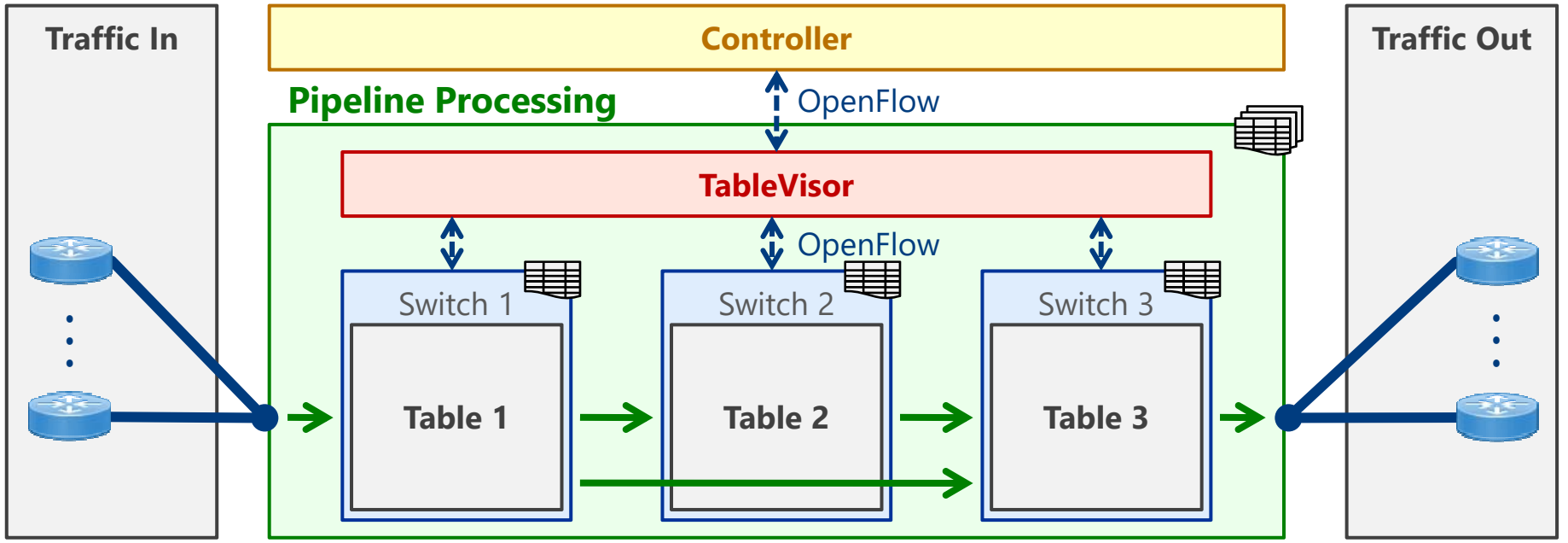


Processing Flow-Stats-Requests



- **Distribution** of Flow-Stats-Requests
  - One new message per involved switch
  - Table to switch mapping
- **Aggregation** of Flow-Stats-Replies

Multi Table Pipeline Processing

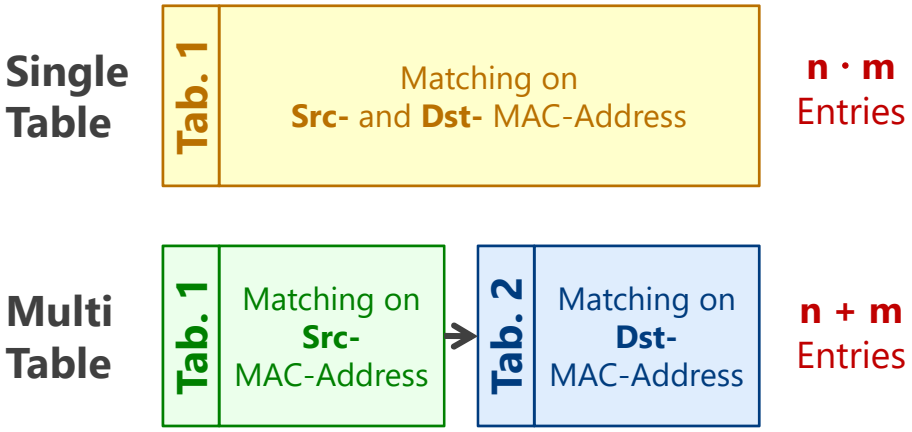


- Combination of **functionalities from different switches**
  - Enables complex use cases
  - Exploits device heterogeneity
- Multi-table switch through **concatenation of hardware devices**
  - Alleviates flow rule explosion
  - Allows multi stage processing

Multi-Table Use Cases

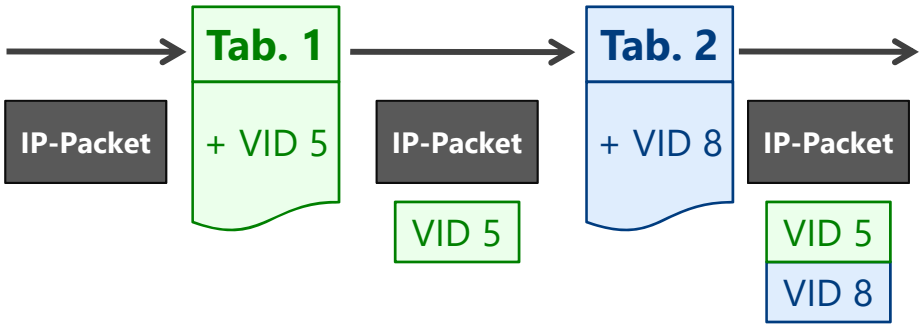
Independent Action

- Single Table leads to rule explosion
- Example: MAC-Address Learning
  - Matching on Src-MAC to learn
  - Matching on Dst-MAC to determine output port



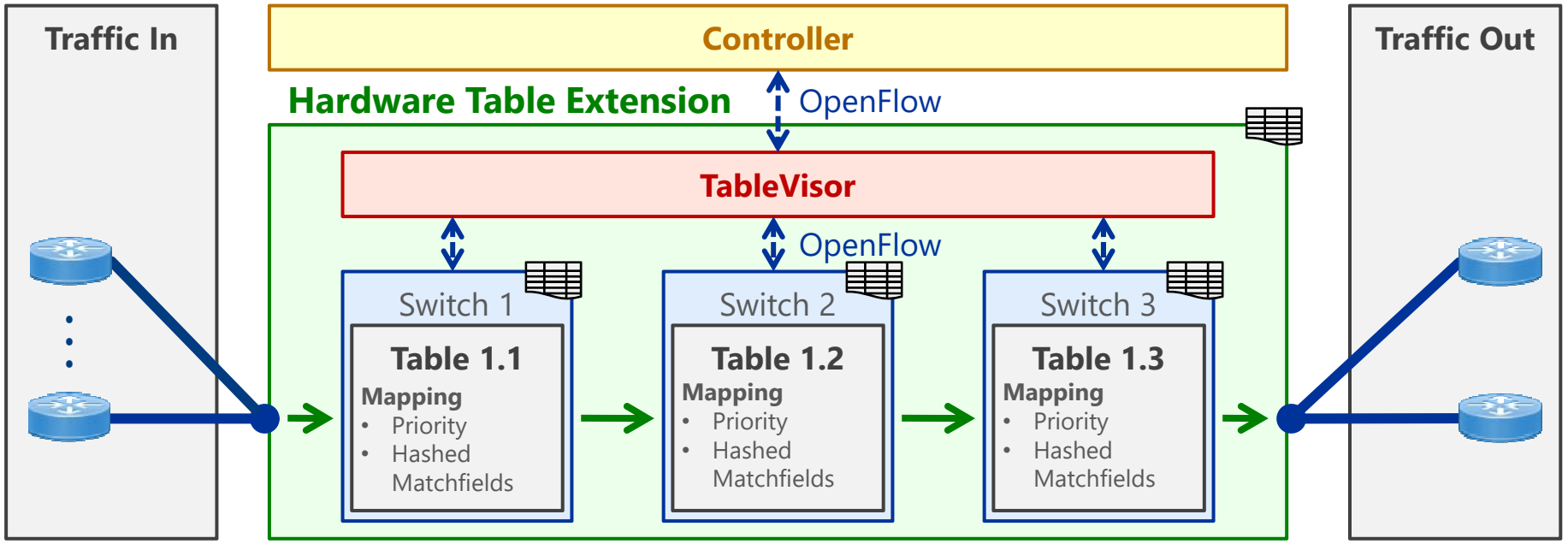
Multi-Stage Processing

- Single action per table
- Example: Stacked VLAN
  - Customer identification
  - Support multiple VLANs per customer



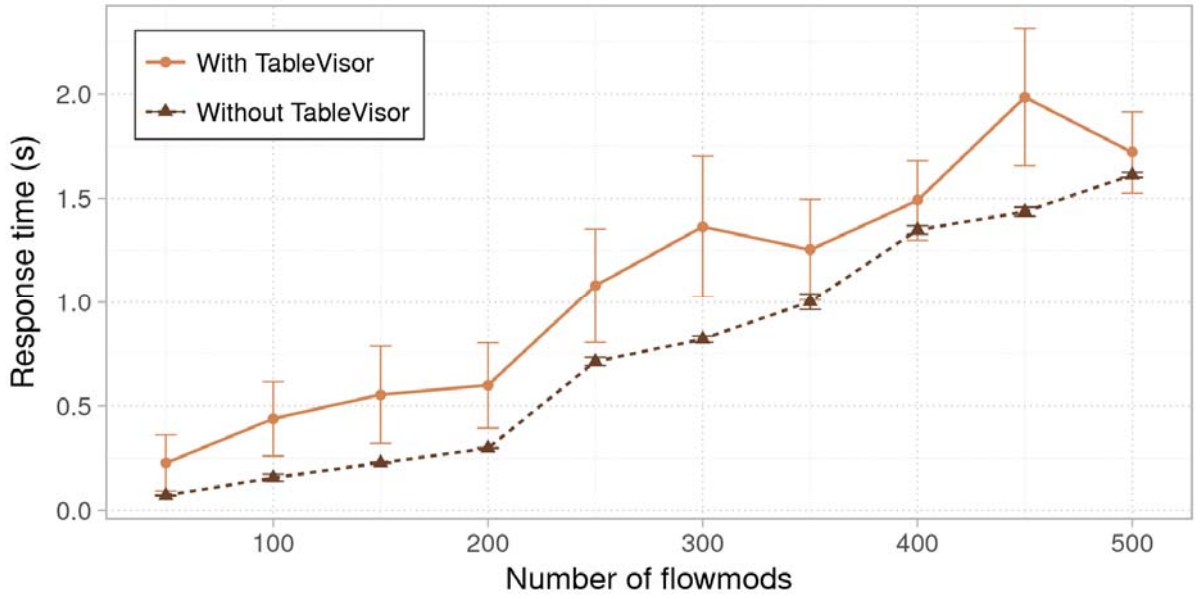
VID = VLAN ID

Hardware Table Extension



- Aggregation of TCAM storage
- **Stateless architecture** allows consistent updates and deletions
  - Split by **priority**
  - Assigned by **hashed match fields**
- Avoid multimatching through **inter-device metadata**

Control Plane Impact



- Measured **flow-mod setup time** using HP 2920 hardware switch
  - Send 1-500 flow mods followed by barrier request
  - Setup time between barrier request and reply
- Control path delay increased by a constant offset

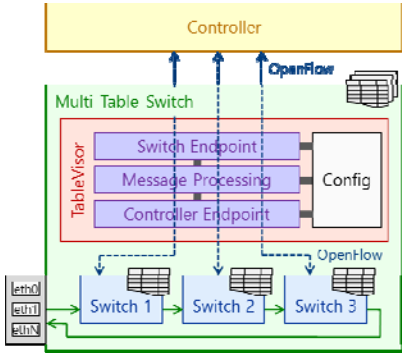
Current and Future Work

- Port of prototype software from Erlang to Java
  - Standalone software tool
  - Loxigen OpenFlow libraries
- Assessment of usability in P4 context
- Evaluation of further use cases
  - Device specific message processing
  - OpenFlow version translation
  - Dynamic resource and feature pooling
  - Local Repair

Any use case ideas?
I would really like to discuss them with you!

Conclusion

- **Stateless and transparent** OpenFlow proxy layer
  - Feature aggregation
  - Hardware table extension



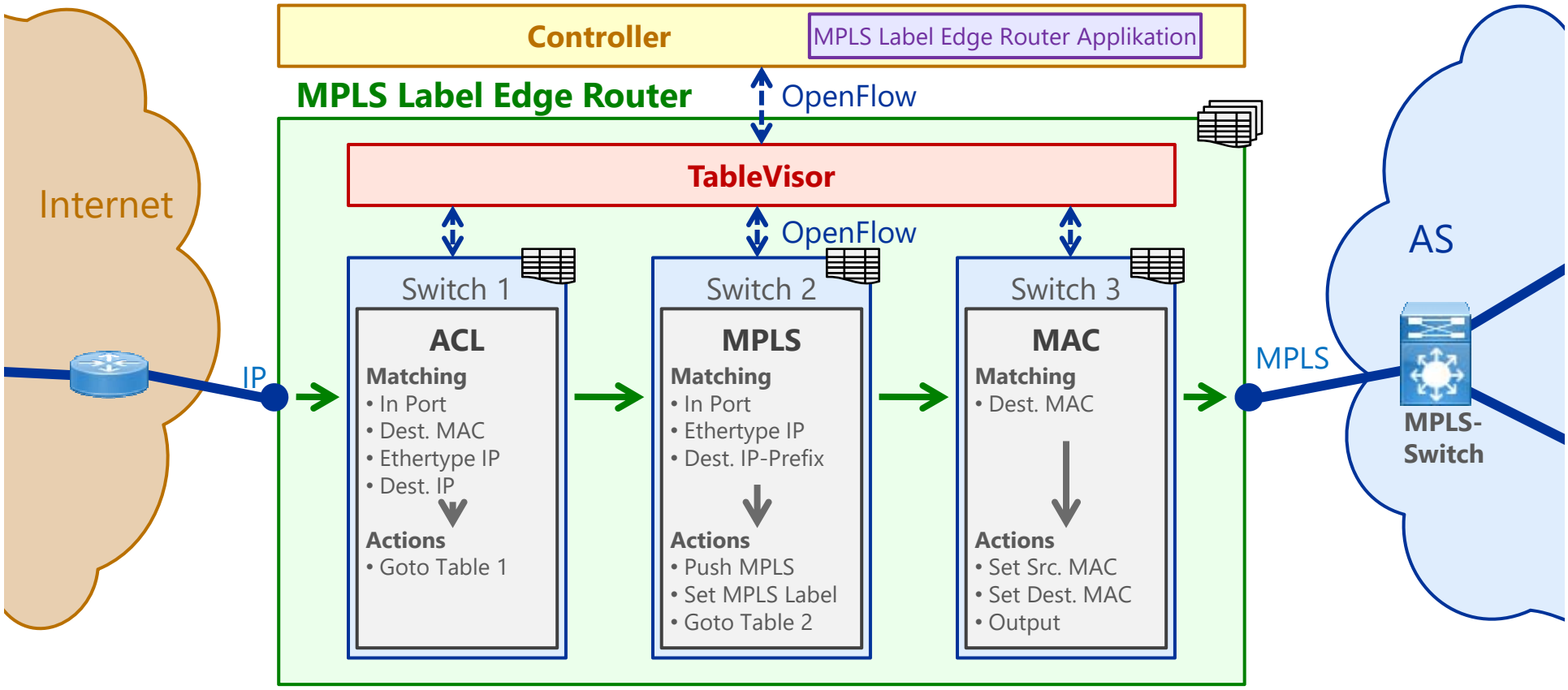
- Control plane **requirements** vs. data plane **capabilities**
  - Exploitation of device heterogeneity
  - Enables more complex use cases

GitHub
lsinfo3/TableVisor

- Clear **separation of concerns**
  - Controller handles management
  - Switches provide specialized functionalities

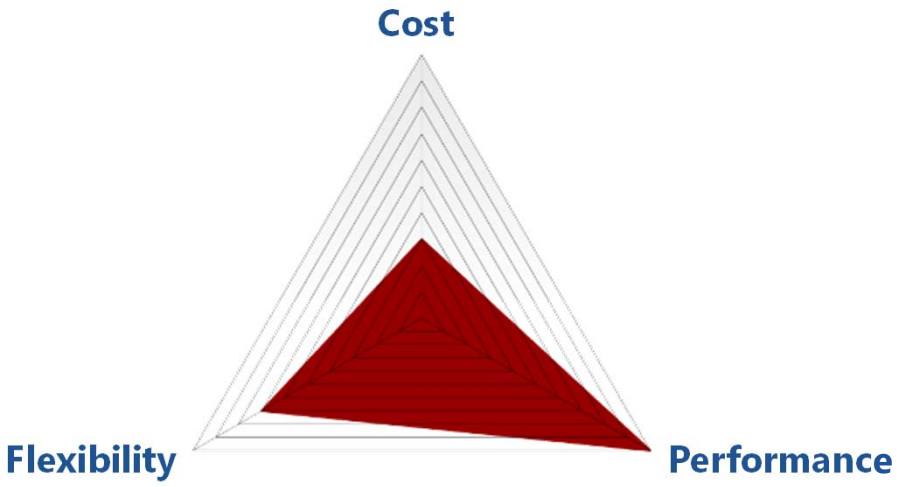
- Future support for **dynamic resource pooling** and **local repair**

Pipeline Processing

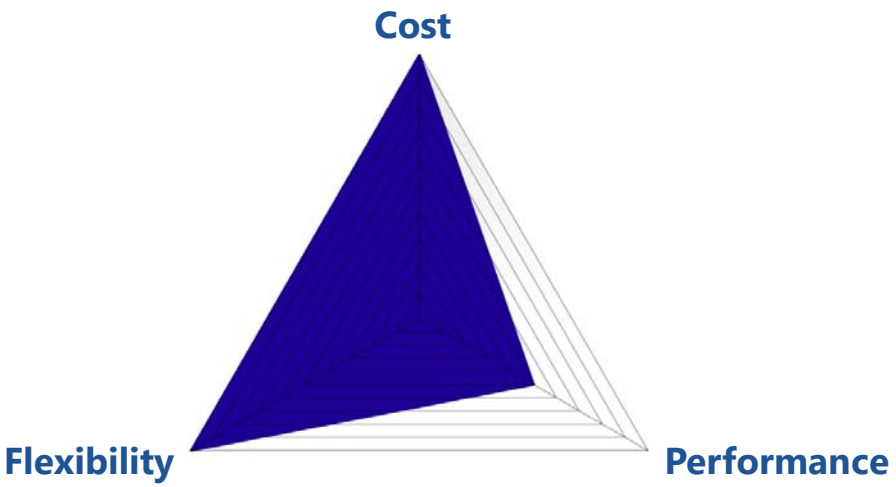


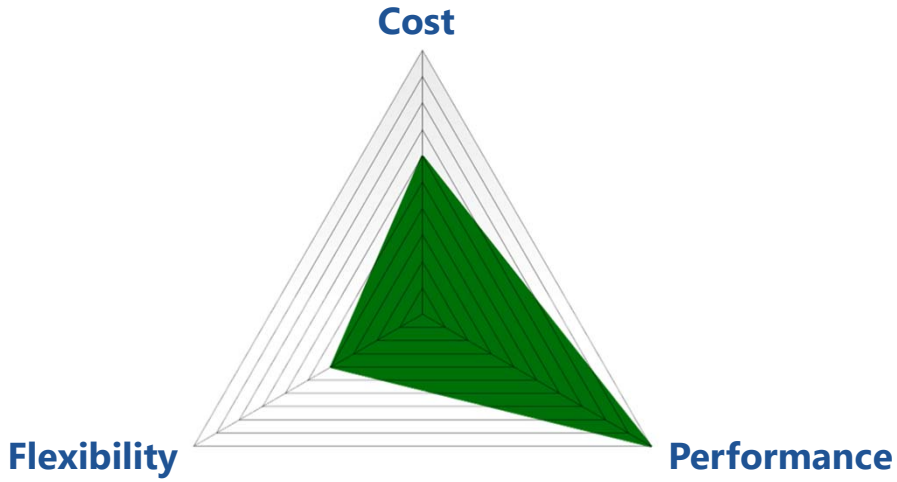
S. Gebert et al. "TableVisor: An Emulation Layer for Multi-Table OpenFlow Switches". European Workshop on Software Defined Networks (EWSDN), 2015.

Hardware Switches

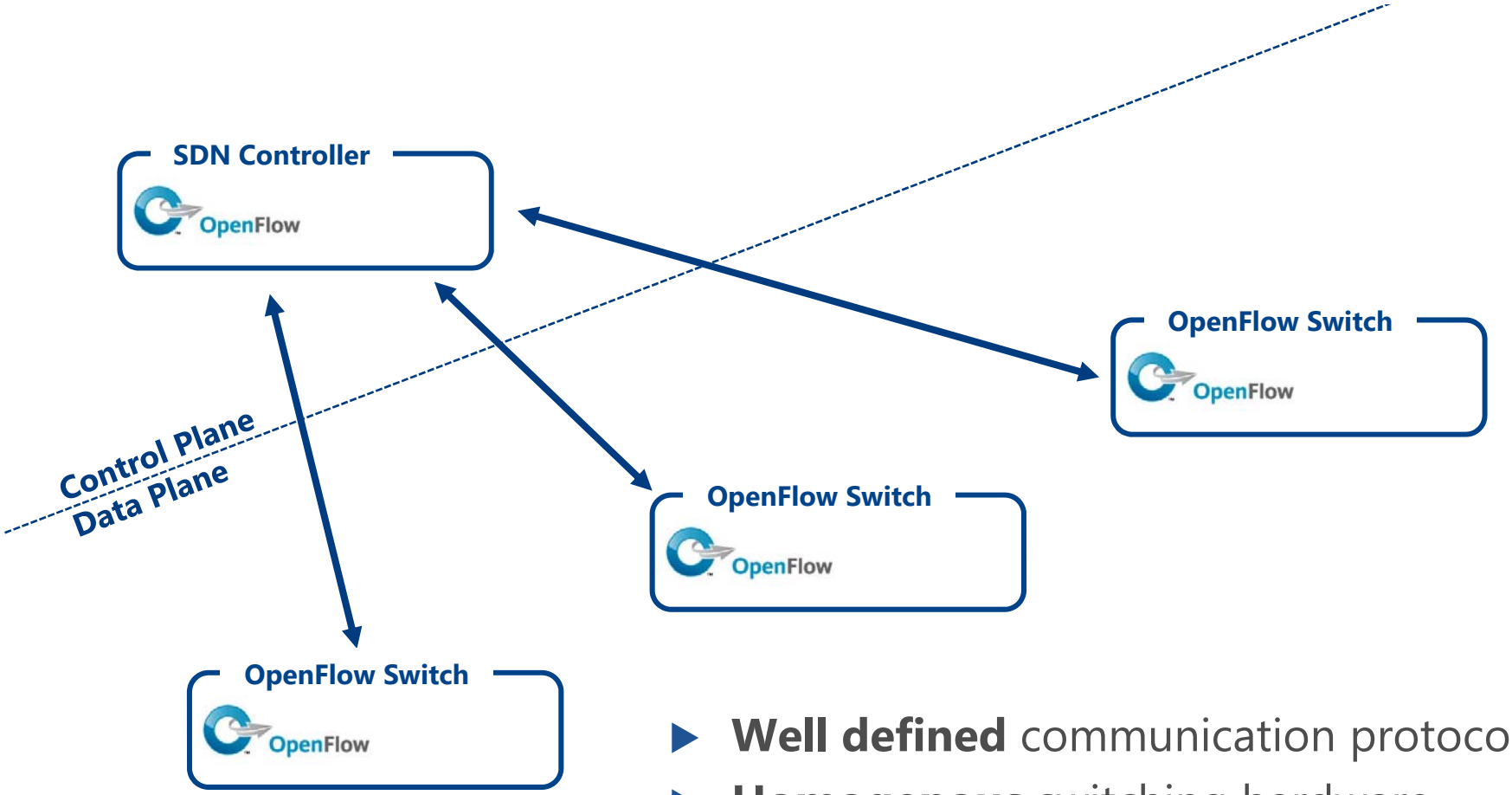


Software Switches



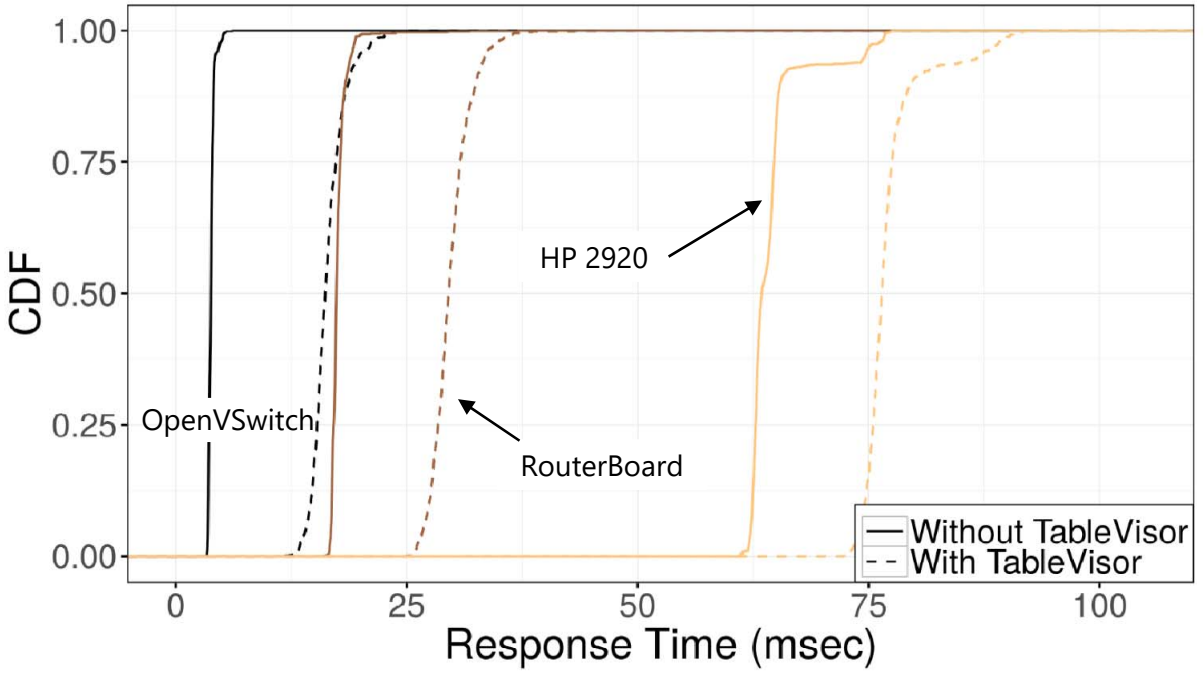


- **Cost-efficient, flexible** software
- **High performance** hardware
- Heterogeneity results in **Trade-off**
  - Data plane performance
  - Initial cost
  - Flexibility and programmability



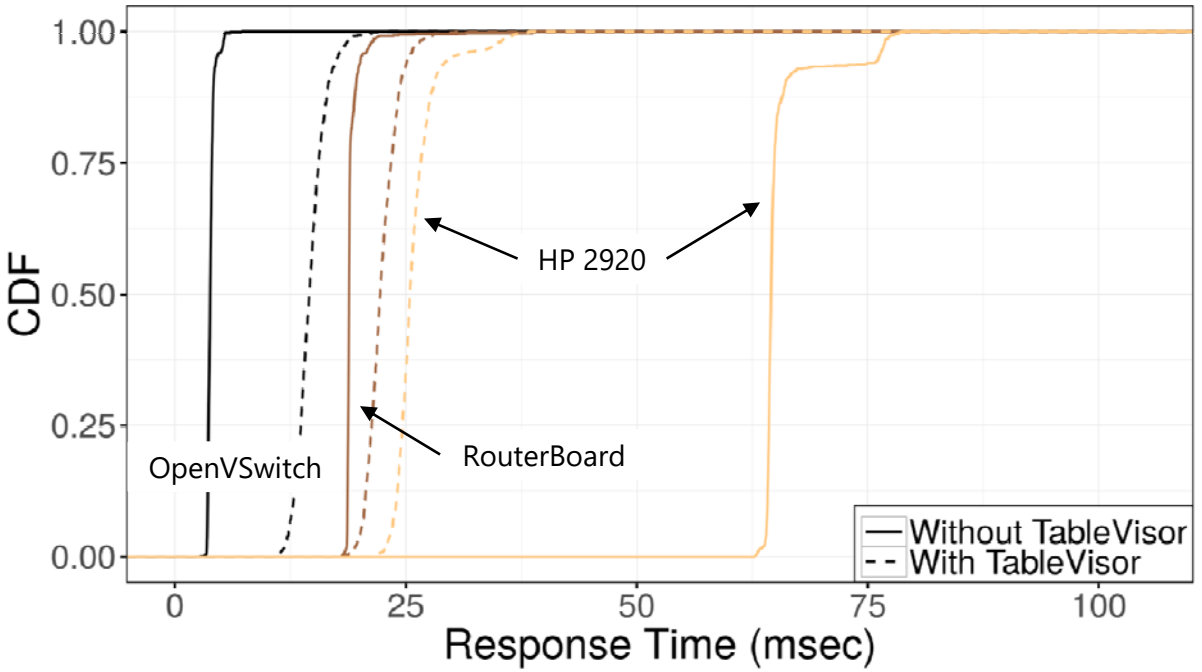
- **Well defined** communication protocol
- **Homogenous** switching hardware
  - Provide **full** OpenFlow support
  - **Respect** OpenFlow specification
  - Support **newest** OpenFlow version

Control Plane Impact



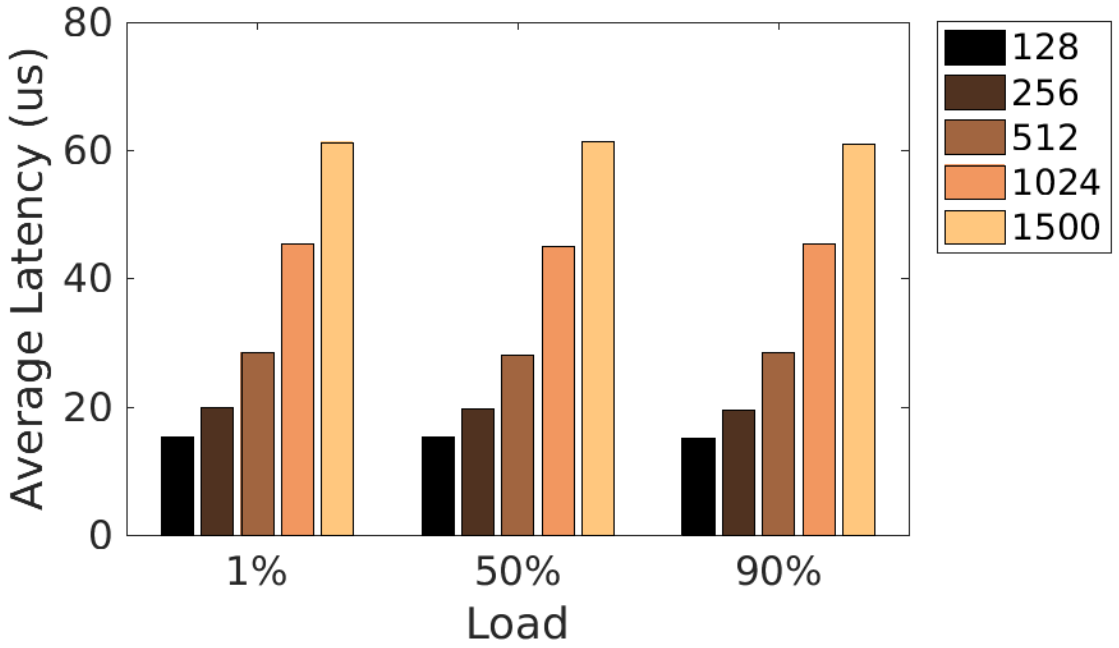
- Measured **flow-stats-request response time**
  - 300 flows
  - Single table
- Roughly 10 msec additional delay

Control Plane Impact



- Measured **flow-stats-request response time**
  - 3 x 100 flows
  - Multiple devices
- **No additional overhead** due to message processing

Data Plane Impact



- Measured **average delay** using Spirent C1
- 4 switch pipeline of HP 2920 switches