Software-defined networking – Smart Grid

Introduction to SDN, and OpenFlow 2020 March

SDN overview

Explain the background for SDN

Traditional networking appliance

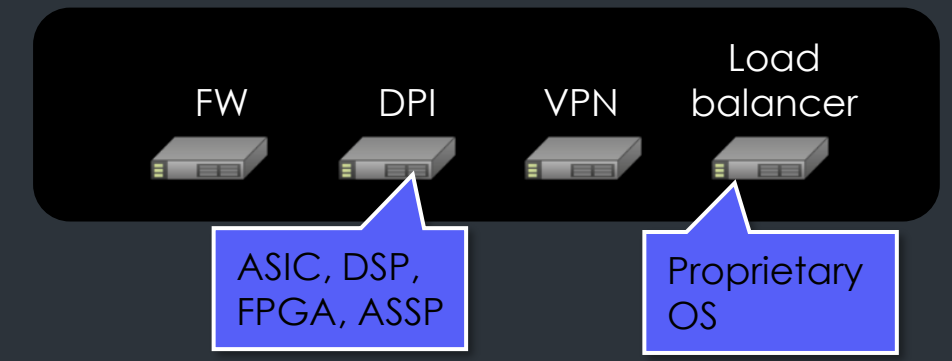
A dedicated proprietary appliance or an ASIC

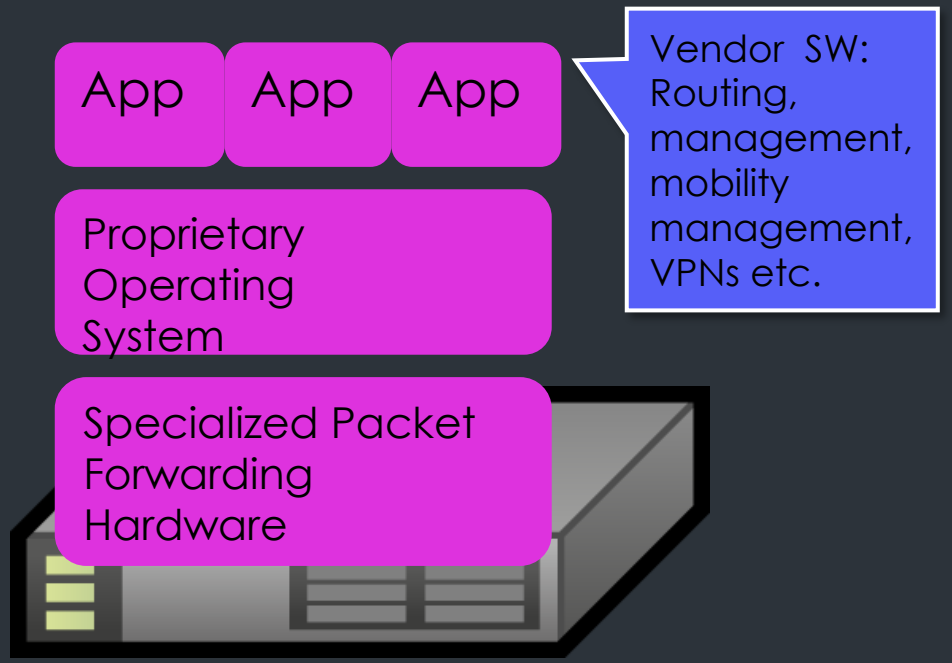
Vendor proprietary SW

- Millions of lines of source code, not disclosed
- Proprietary closed operating systems
- Proprietary development tools

Vendor proprietary HW

Traditional dedicated appliances





The service provider problem

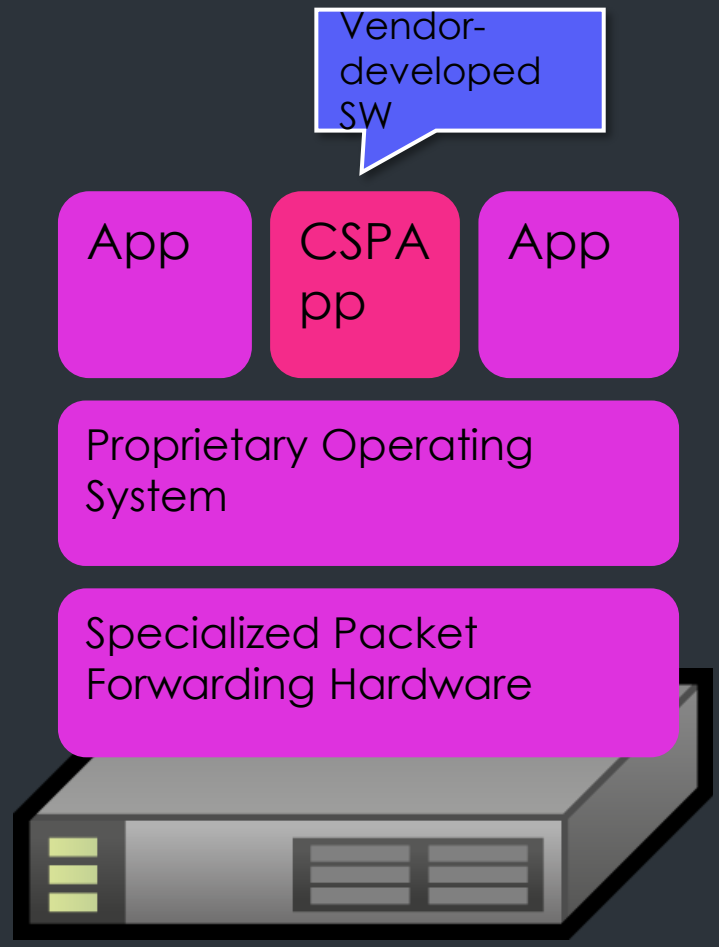
- **Service Providers (SPs) dependent on vendors for adding new services**
  - Proprietary SW in networking appliances

Vendor SW changes slow

- Takes over two years for a vendor to develop required new software!

Network configuration slow and manual

- Provisioning, change, de-provisioning time consuming and error prone



… service not introduced because of

- Lost window of opportunity
  - Service would already be obsolete when finally out
  - Two years to develop required SW (vendor)

    + weeks for testing in network (SP)

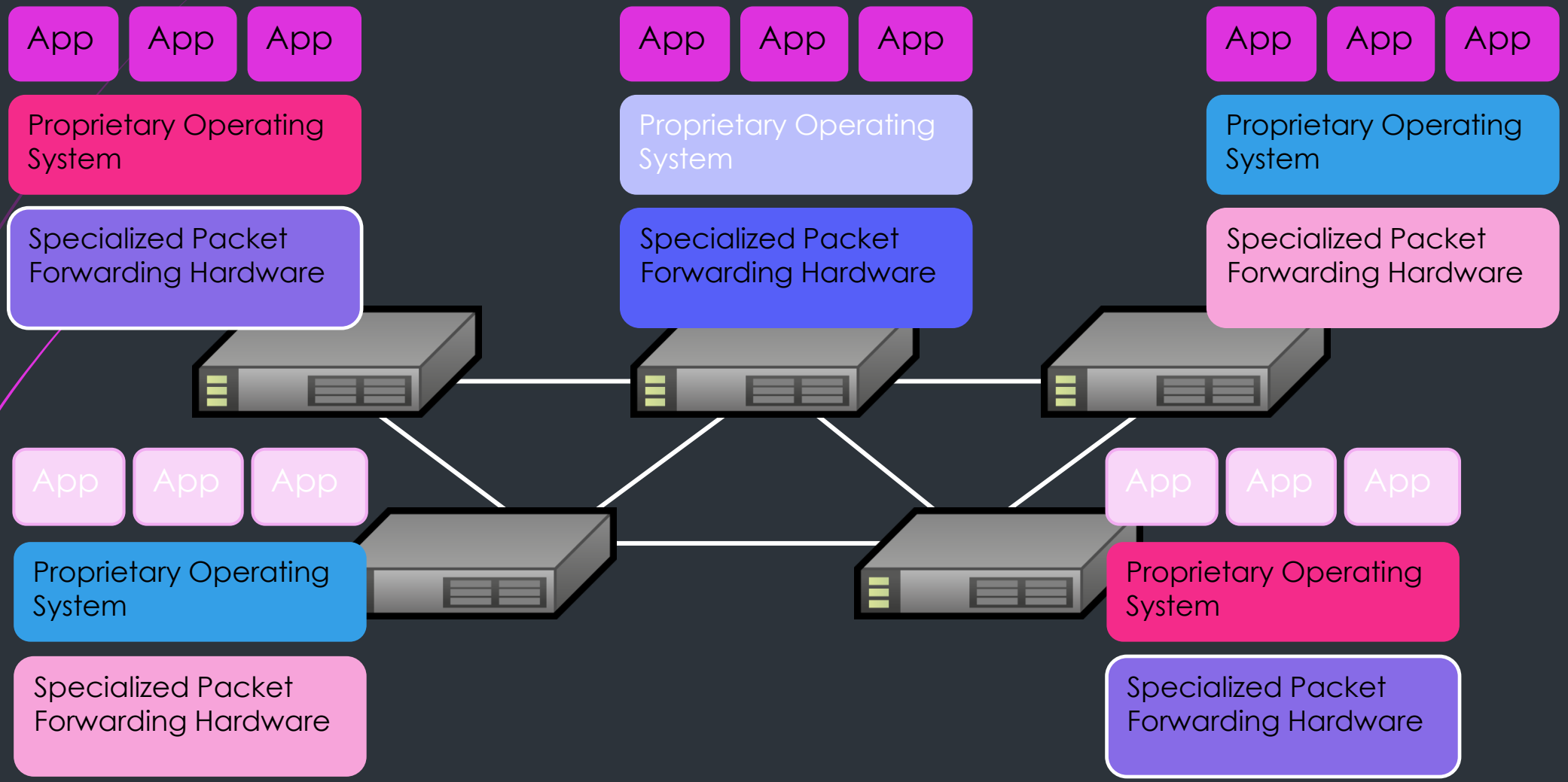
    + months to rollout in network and update OSS/BSS (SP)
- Cost
  - CAPEX: SW development and testing by vendor would make required feature too expensive
  - OPEX: Provisioning required changes (O&M) would be expensive

The solution

From closed and proprietary

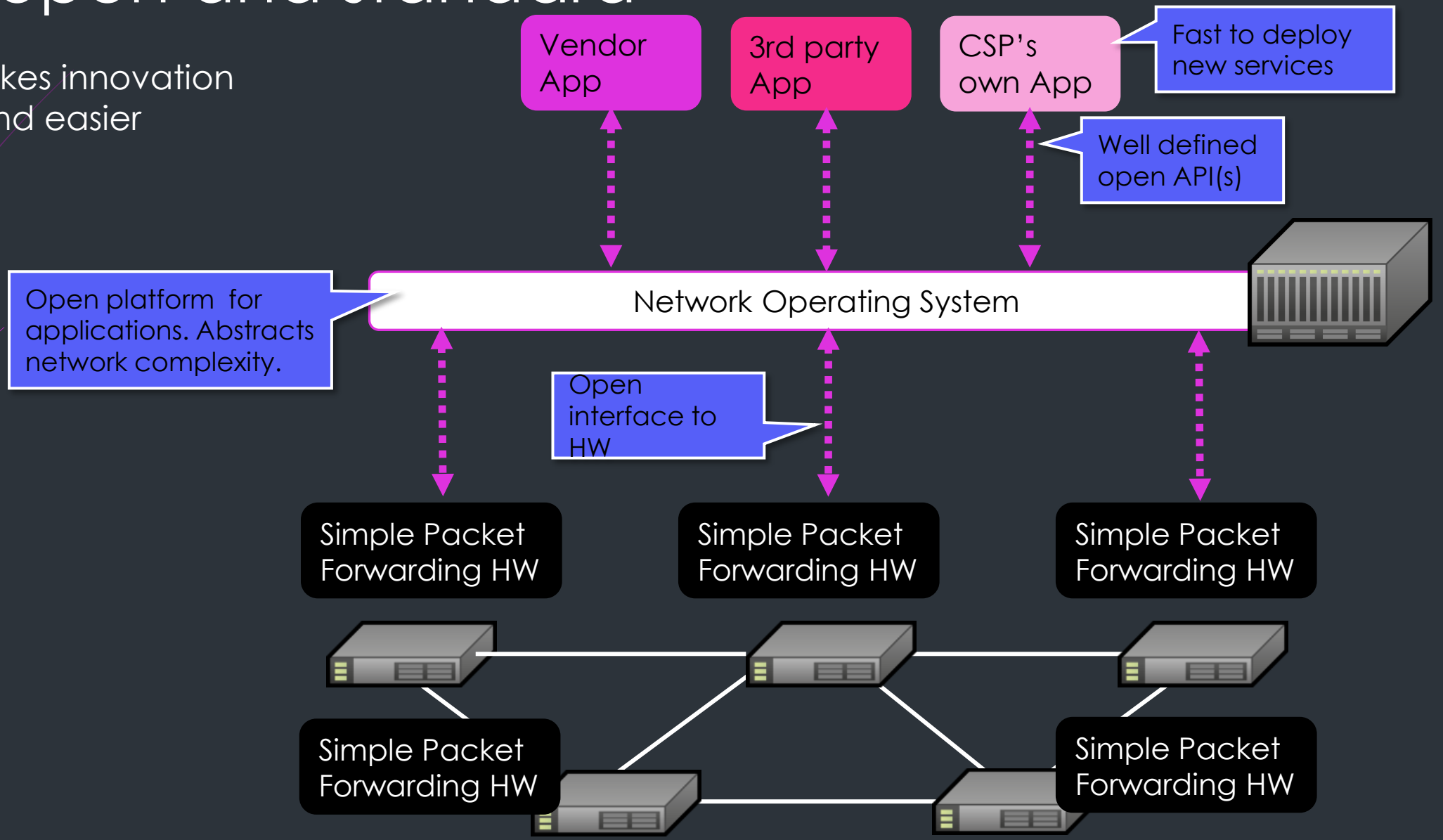
To open and standard

From closed and proprietary…

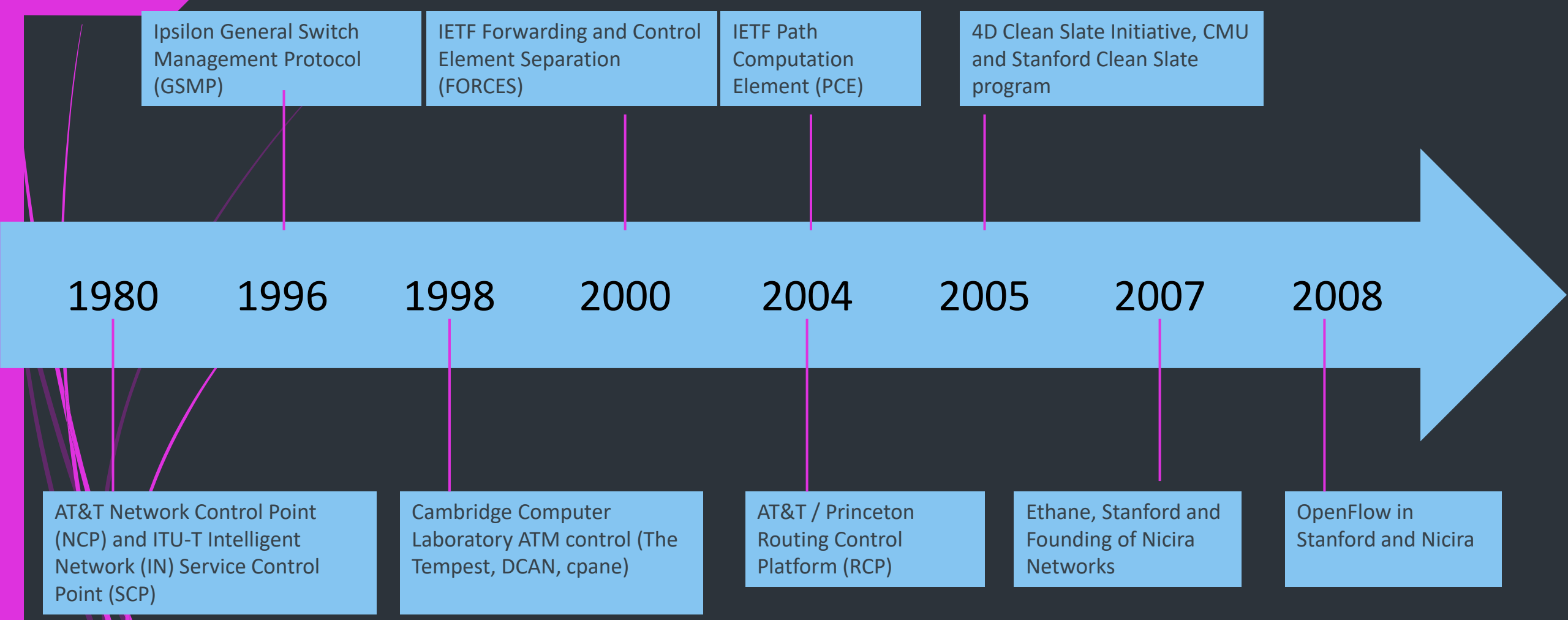


…to open and standard

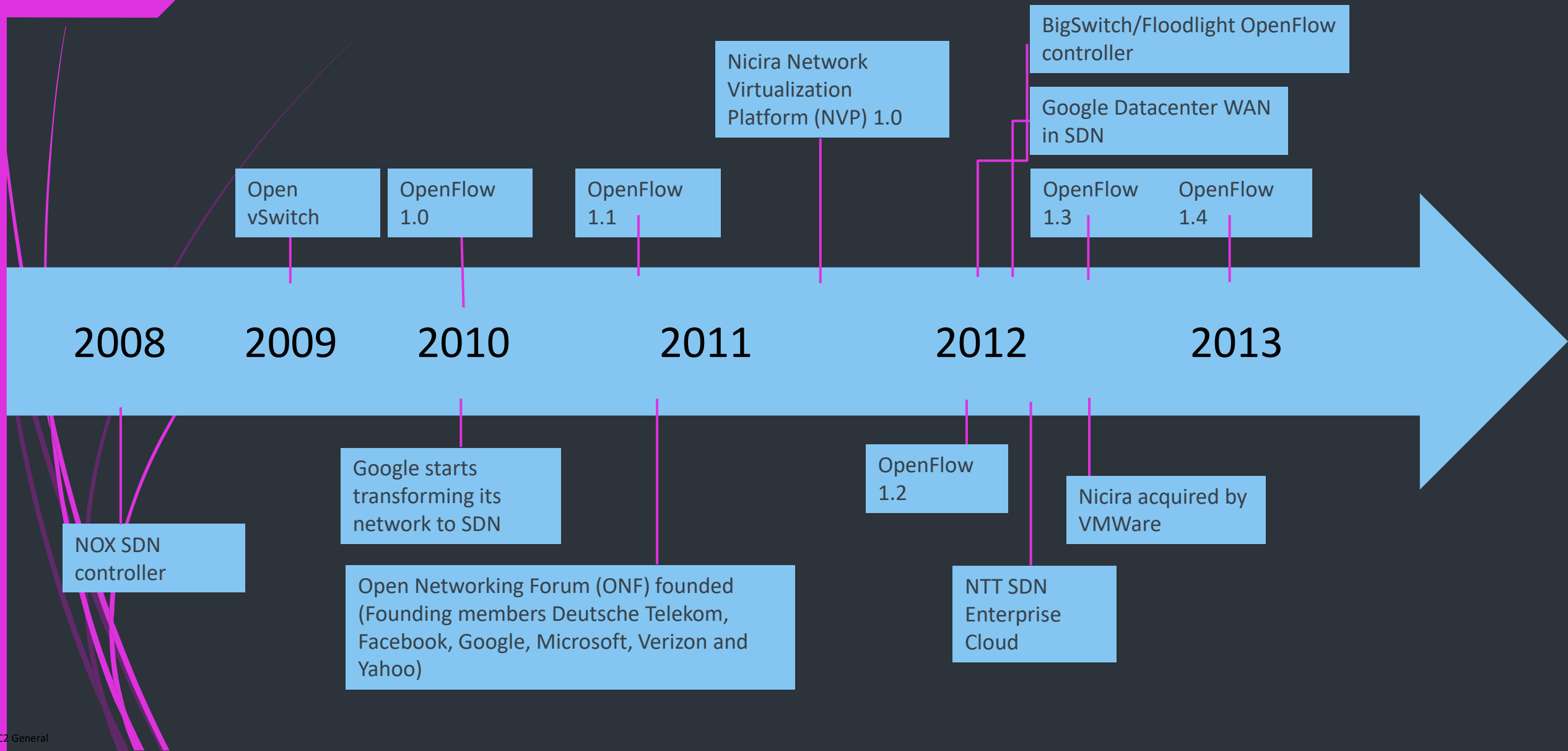
SDN makes innovation faster and easier



SDN HISTORY: the long and winding road



SDN History – the last years

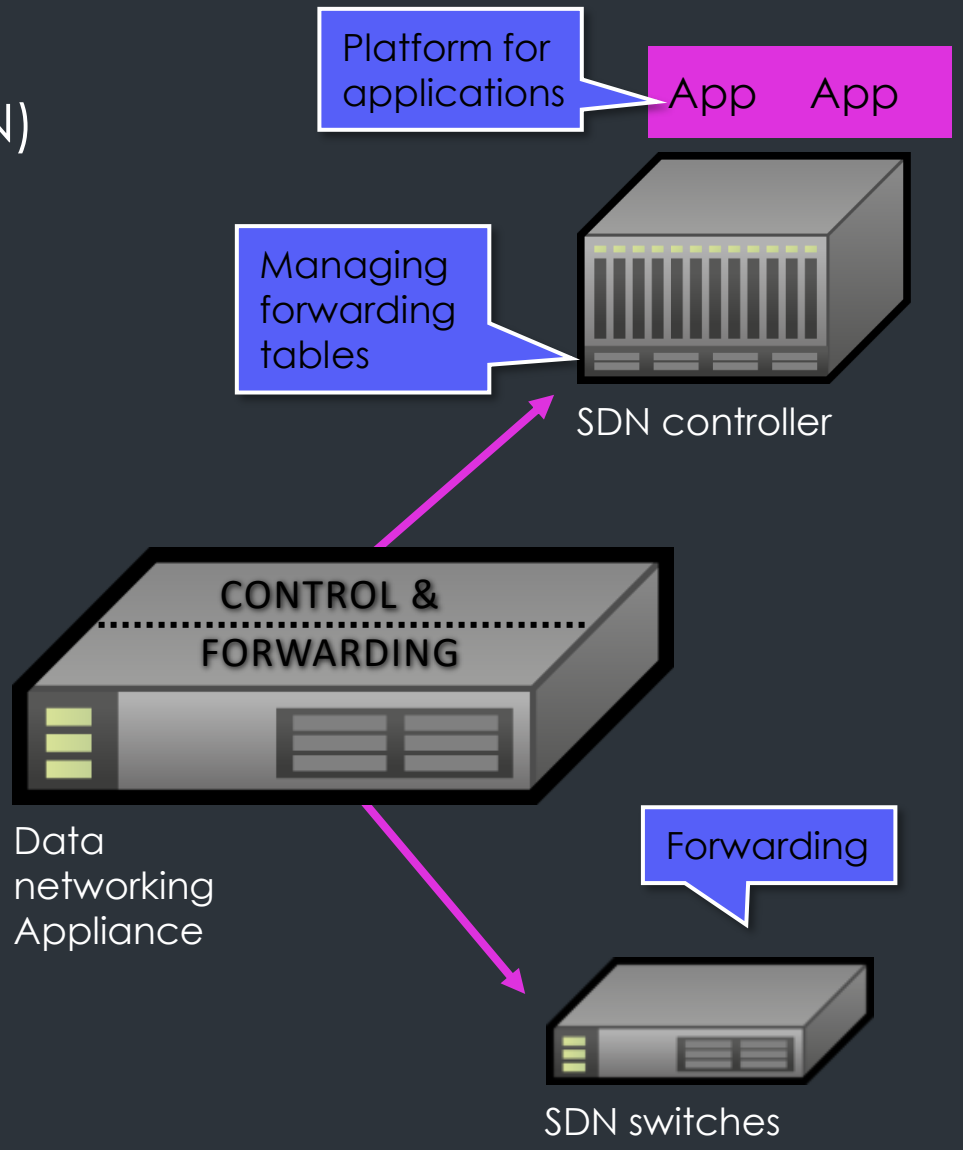


Software Defined Networking (SDN)

Networking control and data plane separation

Abstracting underlying network architecture from applications

Centralized, automated, software driven approach to networking

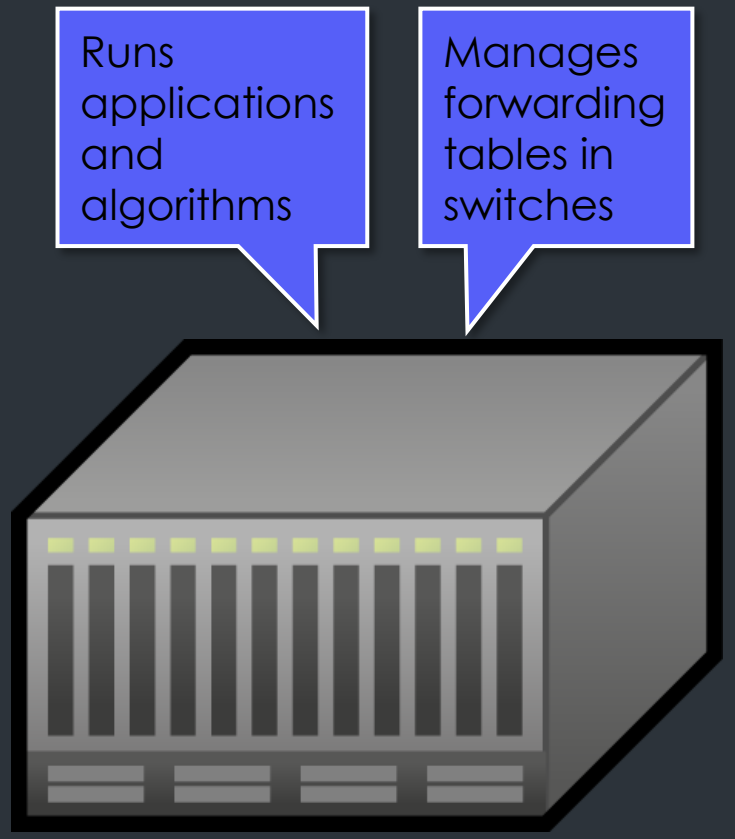


Defining SDN

SDN controller

SDN controller

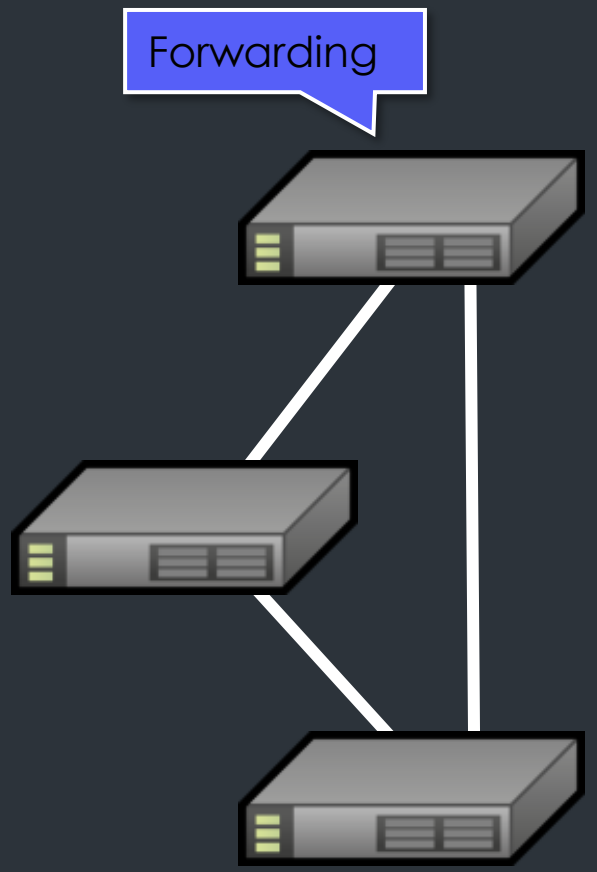
- Network control
- Applies forwarding, QoS and security policies
- Learns network topology, manages forwarding tables in switches
- Platform for network programming via software

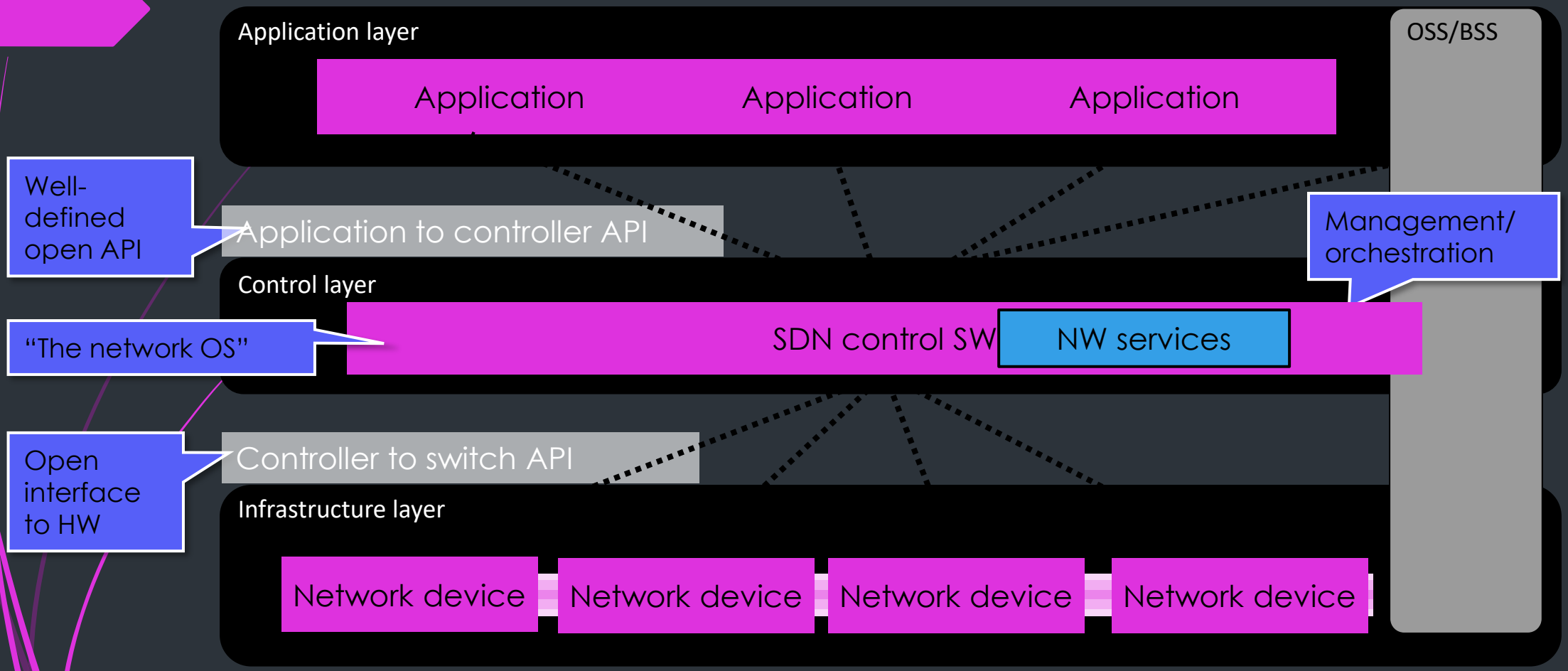


SDN forwarder (”the switch”)

SDN data plane options

- Software router
- OpenFlow SW switch: MiniNet, Open virtual switch (vSwitch)
- Programmable HW: NetFPGA
- Traditional router/switch with OpenFlow or proprietary “control” interface




Application layer
Application
Application
Application
OSS/BSS
Well-defined open API
Application to controller API
Management/ orchestration
Control layer
"The network OS"
SDN control SW
NW services
Open interface to HW
Controller to switch API
Infrastructure layer
Network device
Network device
Network device
Network device

Open Networking Foundation (ONF)

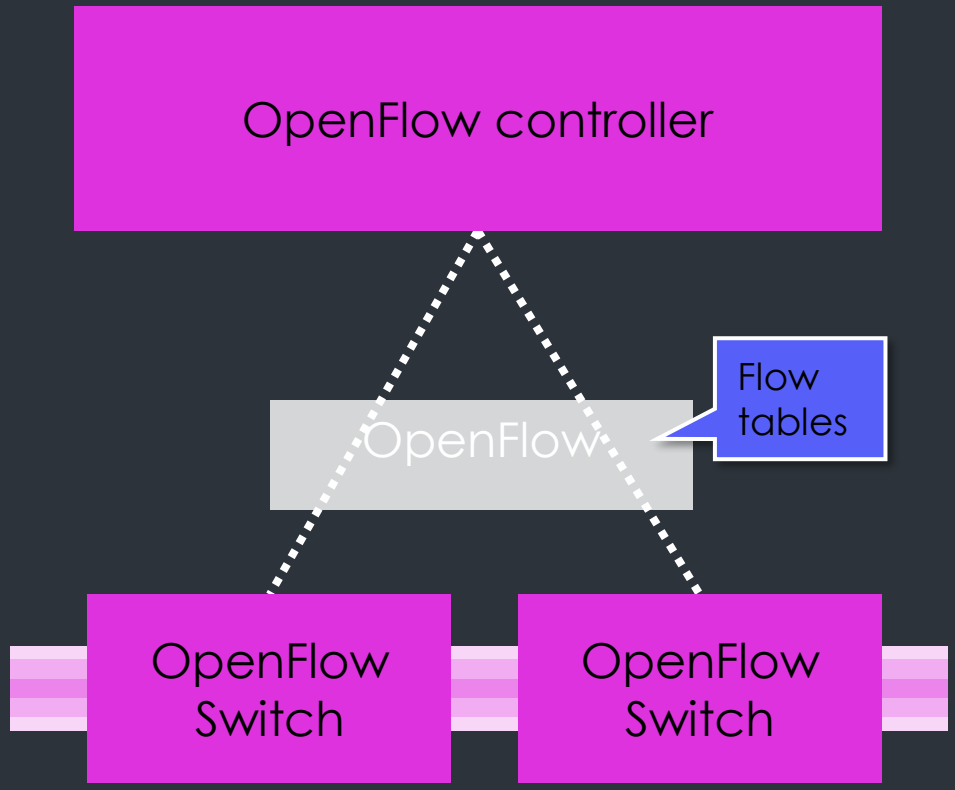
- A user-driven organization dedicated to promotion and adoption of Software-Defined Networking (SDN) through open standards development
- Continues to analyze SDN requirements
- Evolve the OpenFlow standard to address needs of commercial deployments
- Research new standards to expand SDN benefits

OpenFlow

A standard interface enabling SDN

Dynamically programs internal flow tables in switches

Flow tables alter traffic flows (similar to ACLs or Firewall rules)



Floodlight

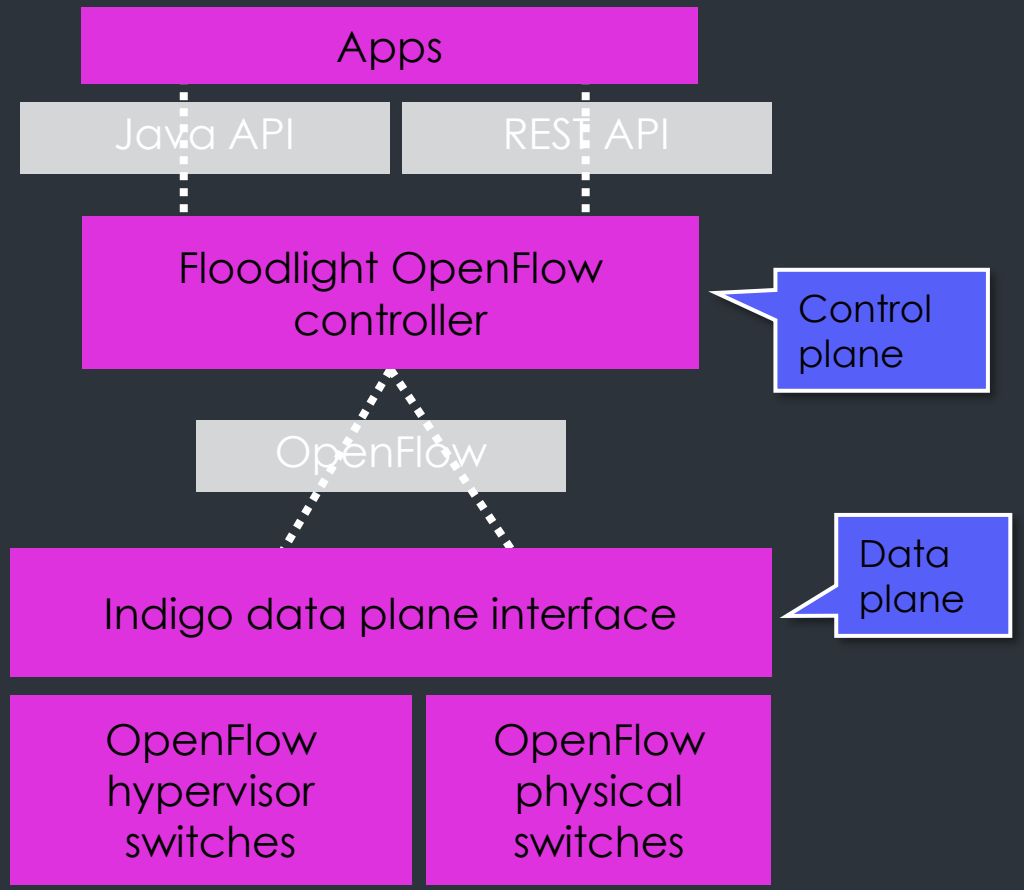
Java based Open Source SDN / OpenFlow controller

Big Switch Networks sponsored community project

Indigo virtual switch (IVS) runs on Linux

Apache license

www.projectfloodlight.org



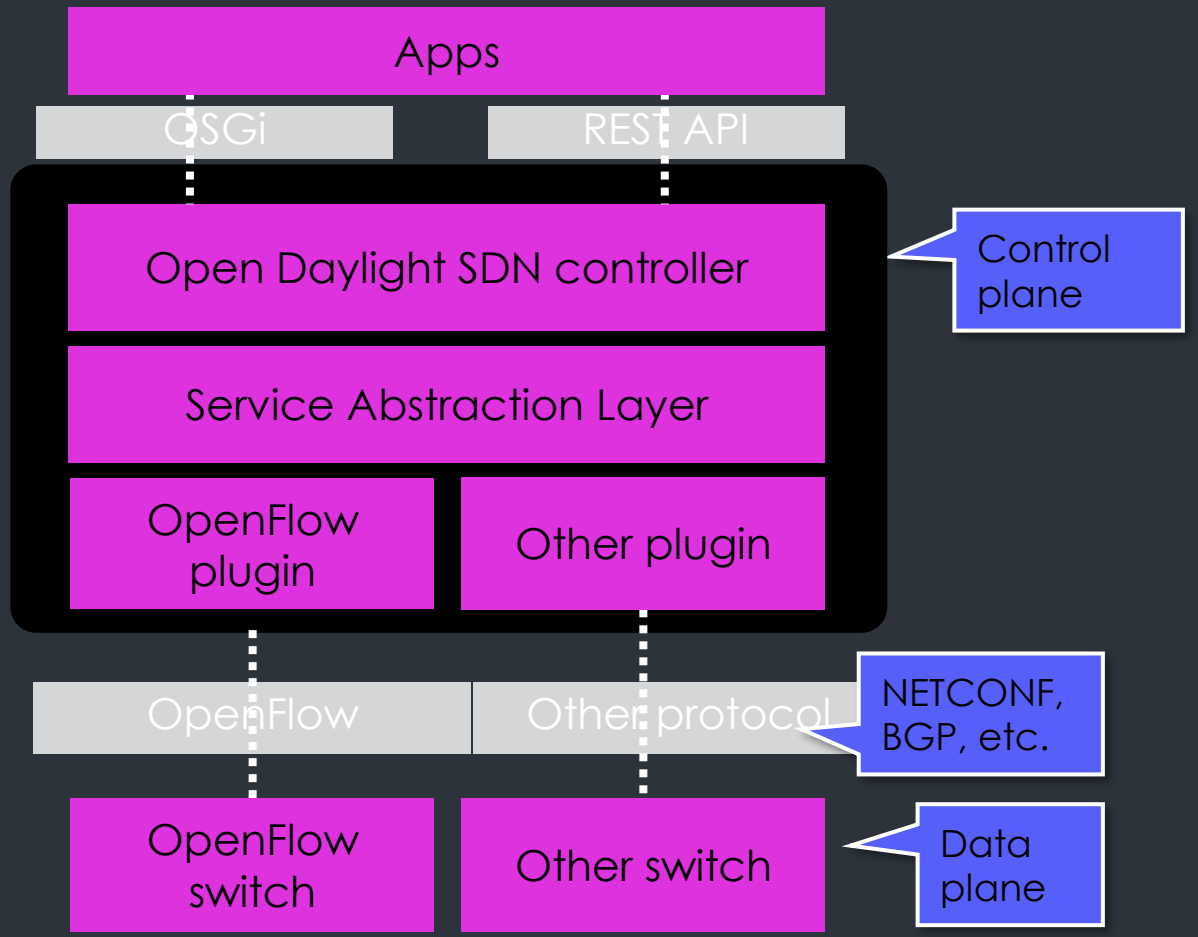
OpenDaylight

Developing open source controller for SDN

Implemented in Java

Gaining industry support: Brocade, Cisco, Citrix, Ericsson, HP, IBM, Juniper, Microsoft, RedHat, …

www.opendaylight.org



OpenFlow operation

OpenFlow operation

1. Establishment of secure channel and topology discovery
2. Configuration of flow table entries
3. Packet flow through switch, table processing

Secure channel

SSL Connection

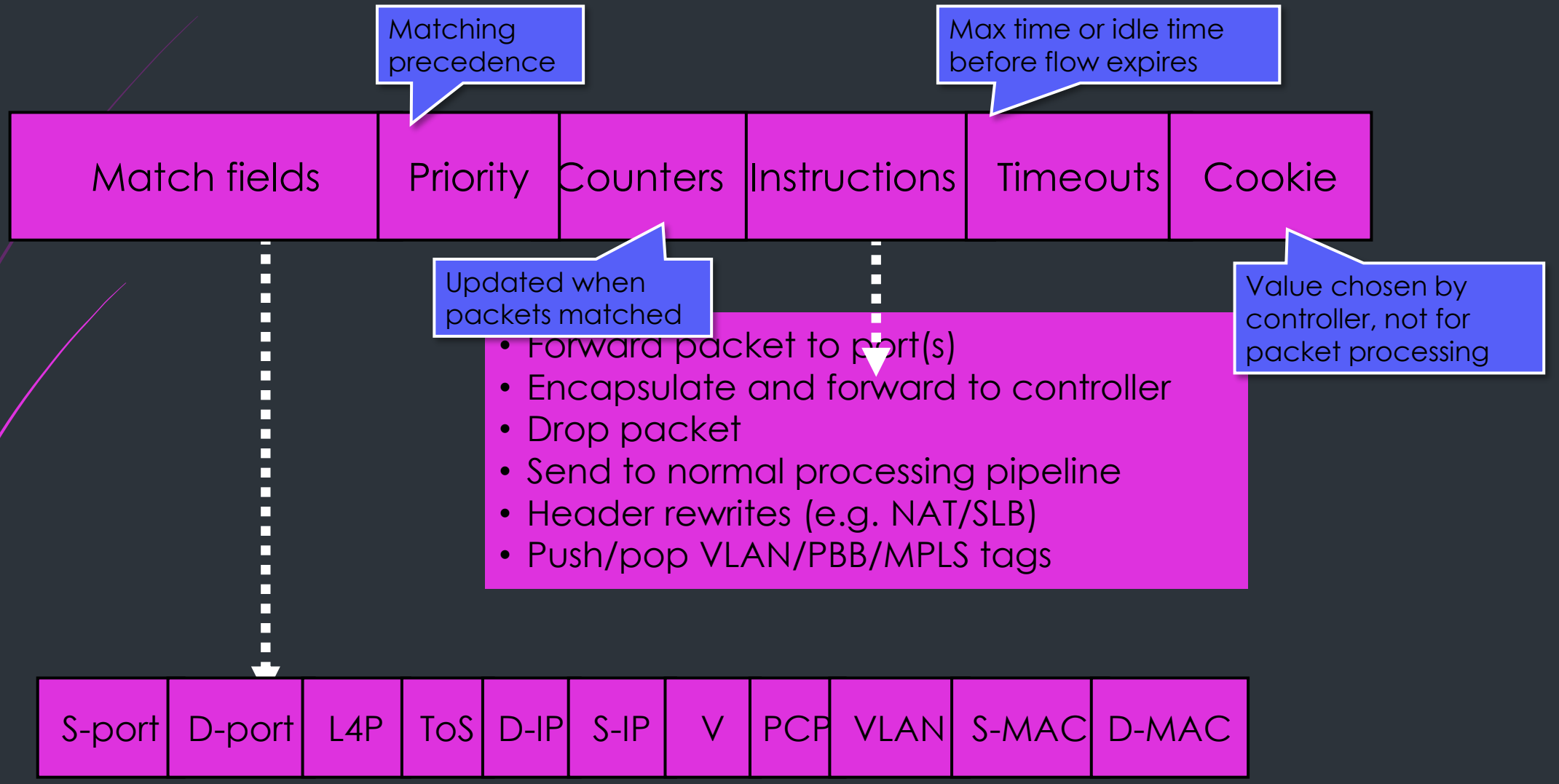
Site-specific key

Controller discovery protocol

Encapsulate packets for controller

Send link/port state to controller

Flow entry main components



OpenFlow table entry

Each entry contains
- Ingress port
- Source and destination MAC address
- Ethertype
- VLAN tag & priority bits
- MPLS label & traffic class
- IP source and destination address (and masks)
- Layer 4 protocol
- IP ToS/DSCP bits
- TCP/UDP port numbers

Adding and removing flow entries

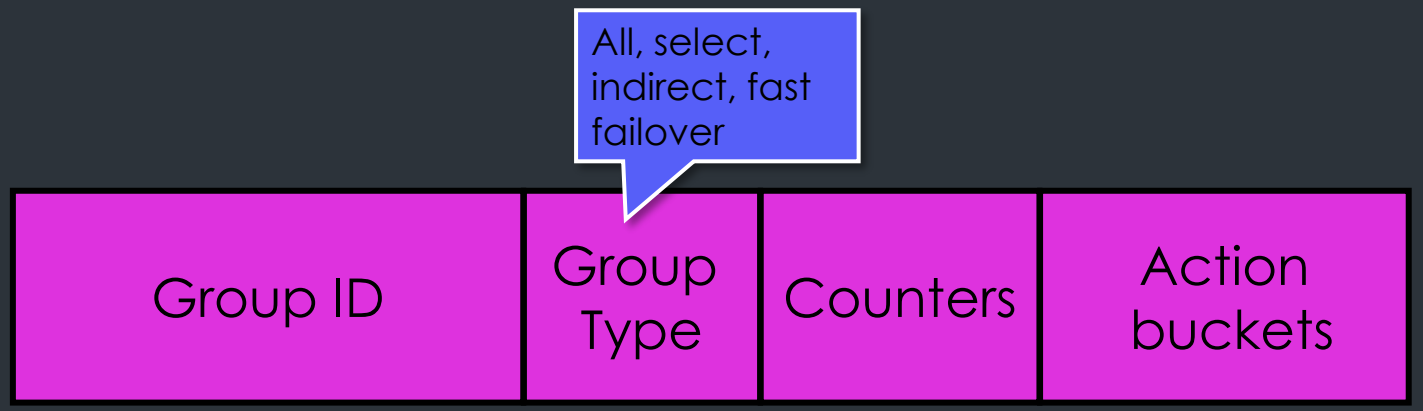
Flow addition
- If incoming packet does not match a flow, a flow must be created, or the packet is dropped
- Controller is notified
- Controller can create the flow if necessary

Flow removal
- Timer expiry
  - idle_timeout = entry removed x seconds after last packet
  - hard_timeout = entry removed after x seconds
- Controller actively removes
  - OFPFC_DELETE or OFPFC_DELETE_STRICT

Controller can modify flows too

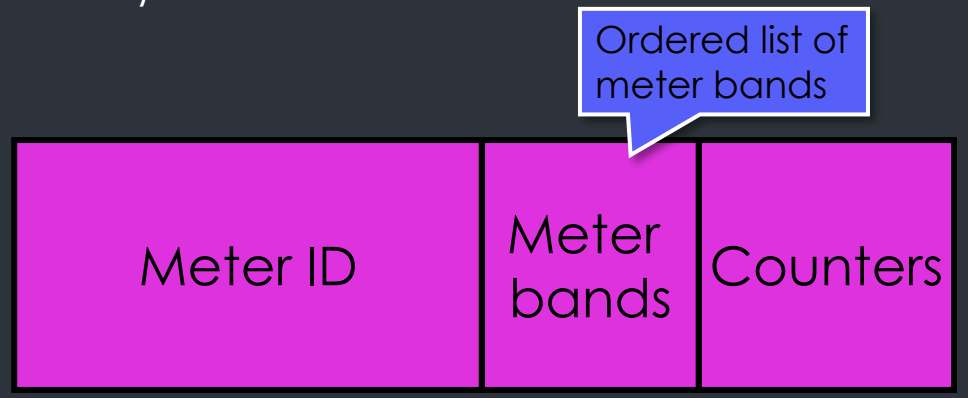
Group table

- Flow entry can also point to group table
- For additional methods of forwarding
  - All - for multicast/broadcast forwarding, clones packet for each "bucket"
  - Select - execute one bucket in group selected by implementation specific algorithms
  - Indirect - execute single bucket, for multiple flow entries or groups to point to common group ID, for fast convergence
  - Fast failover - execute first live bucket



Meter table

- Meter entries with per-flow meters
  - A meter measures rate of packets and enables controlling those packets
- For simple QoS operations, e.g. rate limiting
  - Can be combined with per-port queues to implement complex QoS frameworks (Diffserv)



Switch table processing pipeline

- Packets can be matched against multiple tables
- Each flow table can contain multiple flow entries
- There is one table in minimum

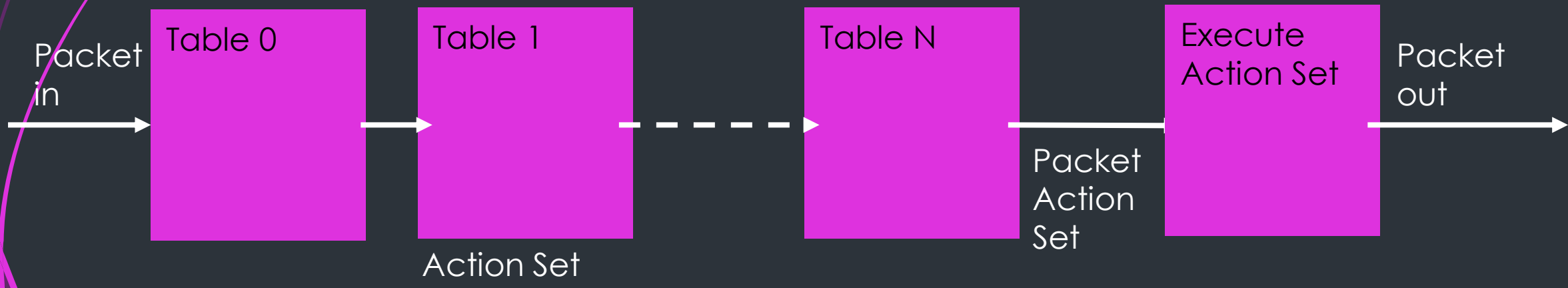
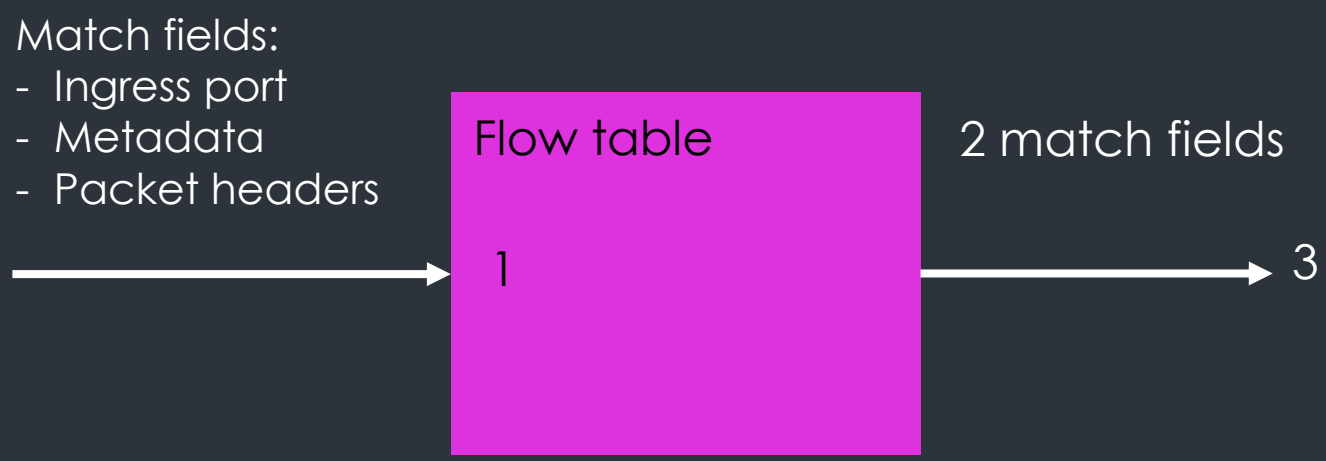
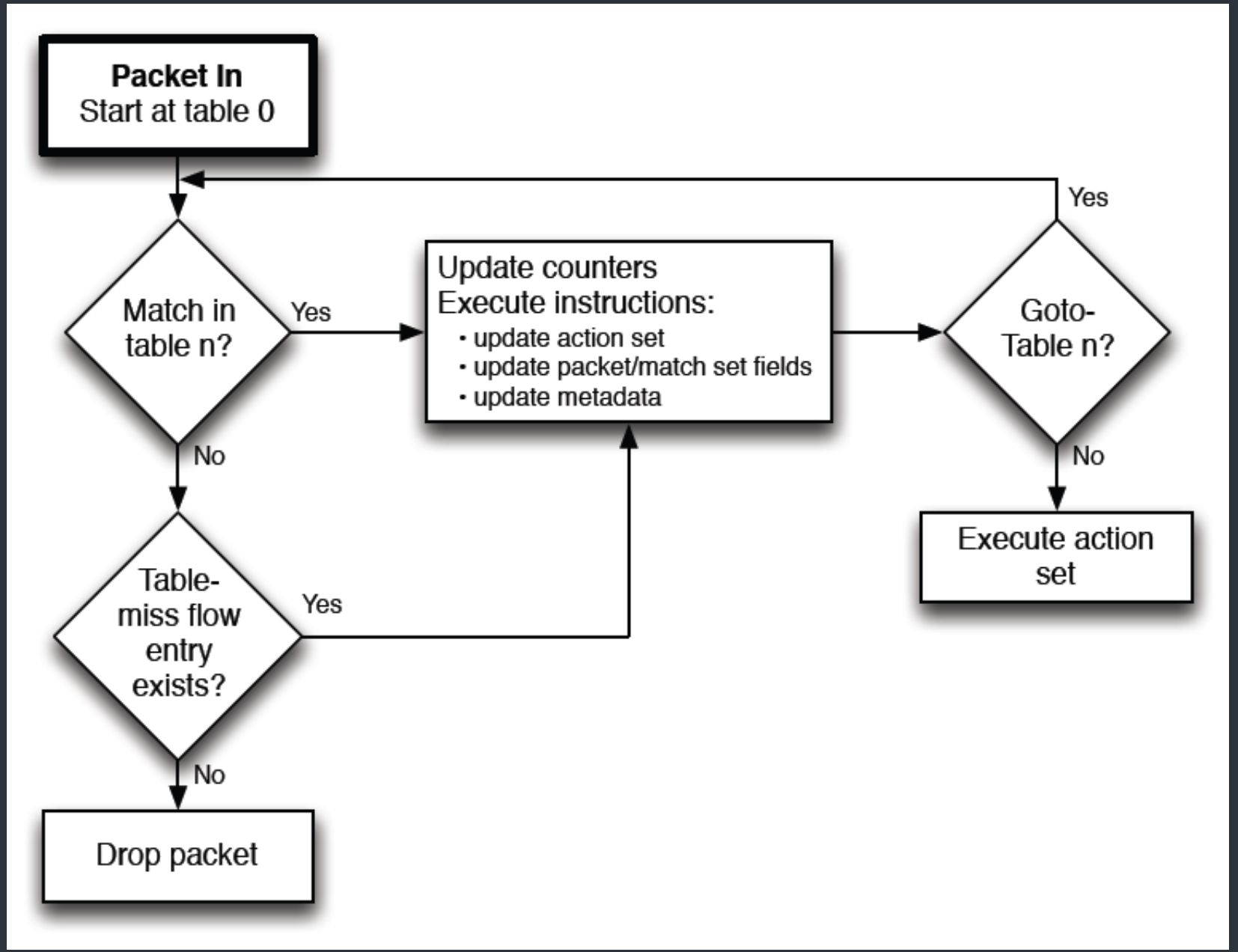


Table processing in switch

1. Find highest-priority matching flow entry
2. Apply instructions
   - Modify packet and update match fields
   - Update action set
   - Update metadata
3. Send match data and action set to next table



Packet flow through switch



Action processing order

1. copy TTL inwards: apply copy TTL inward actions to the packet
2. pop: apply all tag pop actions to the packet
3. push-MPLS: apply MPLS tag push action to the packet
4. push-PBB: apply PBB tag push action to the packet
5. push-VLAN: apply VLAN tag push action to the packet
6. copy TTL outwards: apply copy TTL outwards action to the packet
7. decrement TTL: apply decrement TTL action to the packet
8. set: apply all set-field actions to the packet
9. qos: apply all QoS actions, such as set queue to the packet
10. group: if a group action is specified, apply the actions of the relevant group bucket(s) in the order specified by this list
11. output: if no group action is specified, forward the packet on the port specified by the output action

Table examples

- Routing

Switch port	src MAC	dst MAC	Ethertype	VLAN	src IP	dst IP	IP proto	src port	dst port	ACTION
*	*	*	*	*	*	83.145.204.153	*	*	*	Port 4

- VLAN Switching

Switch port	src MAC	dst MAC	Ethertype	VLAN	src IP	dst IP	IP proto	src port	dst port	ACTION
*	*	5c:ff:35..	*	79	*	*	*	*	*	Ports2,8,11

- Firewall

Switch port	src MAC	dst MAC	Ethertype	VLAN	src IP	dst IP	IP proto	src port	dst port	ACTION
*	*	*	*	*	*	*	6	*	23	DROP