CS351 - Cloud Computing
Lecture #7

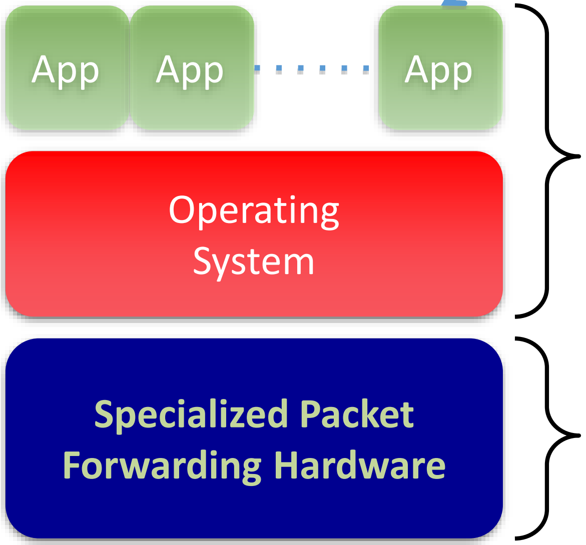
Introduction to Software Defined Network (SDN)

Outline

- Introduction.
- What is Software-Defined Network?
- OpenFlow.
- Research Problems in SDN.

The Networking Industry (2007)

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



Million of lines of source code

Manage by 5400 RFCs

500M gates
10Gbytes RAM

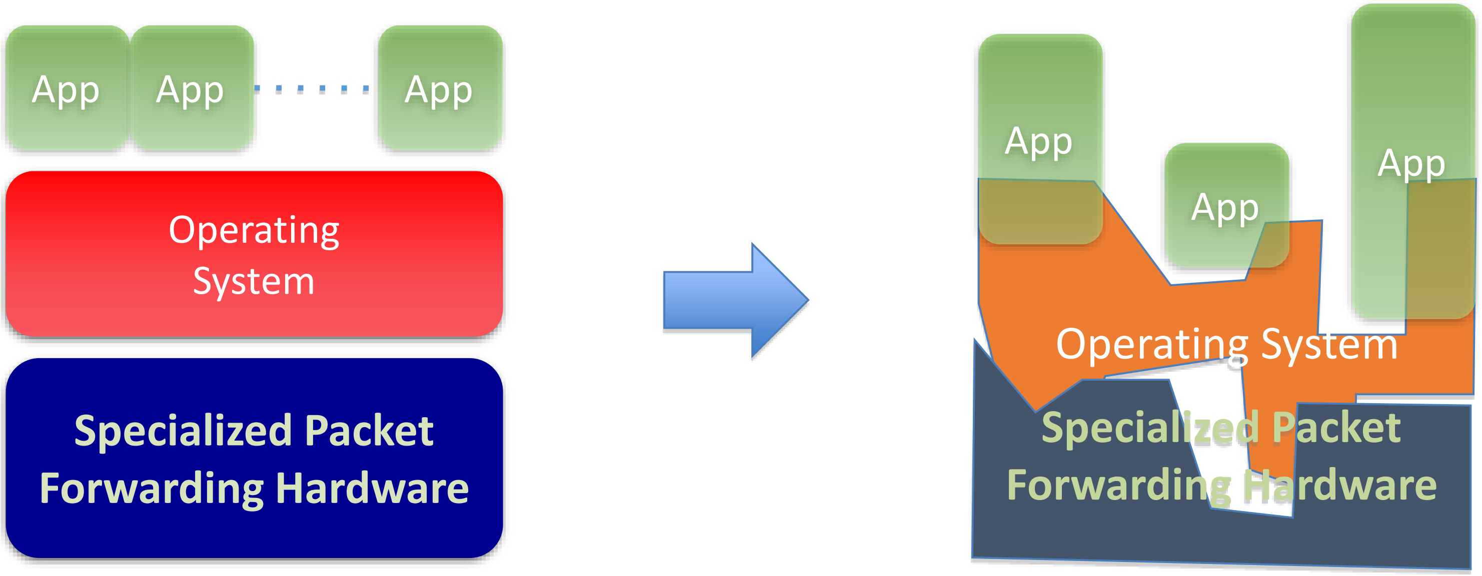
Power Hungry

Many complex functions baked into the infrastructure

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

An industry with a “mainframe-mentality”

Reality…!!!!



Closed equipment

- Software bundled with hardware.
- Vendor-specific interfaces.

Over specified : Slow protocol standardization.

Few people can innovate

- Equipment vendors write the code.
- Long delays to introduce new features.

Operating a network is expensive

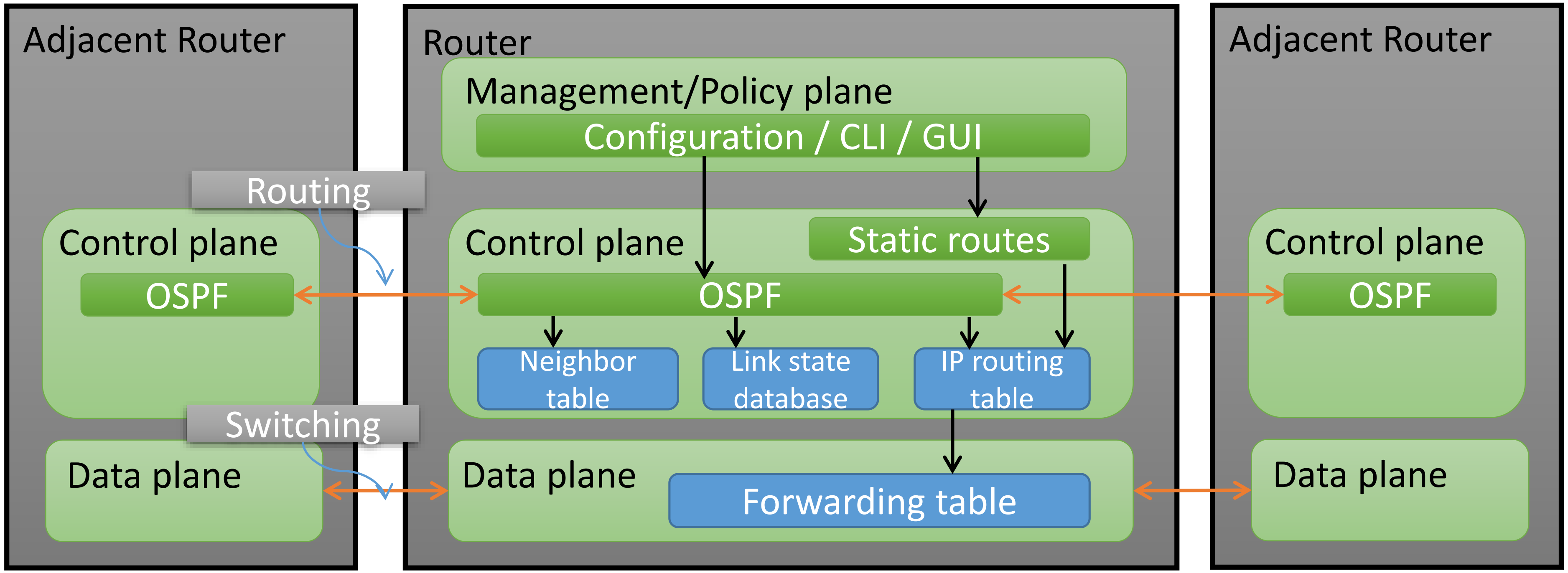
- More than half the cost of a network.
- Yet, operator error causes most outages.

Buggy software in the equipment

- Routers with 20+ million lines of code
- Cascading failures, vulnerabilities, etc.

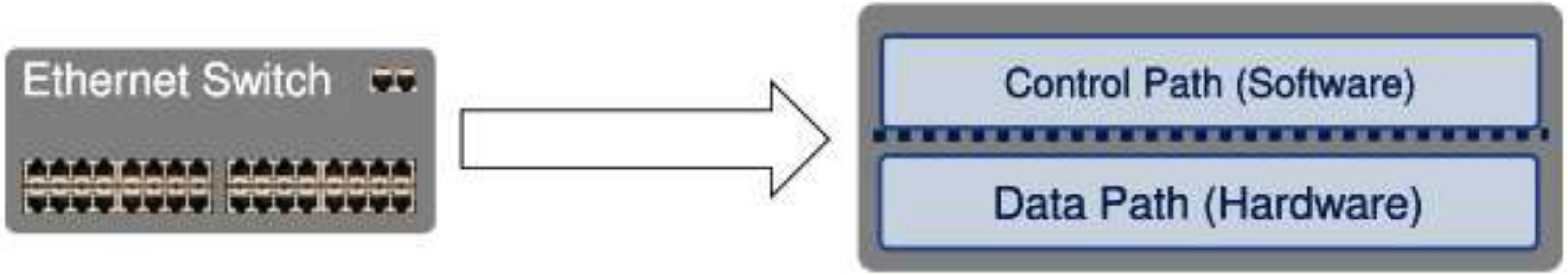
Traditional Network Router

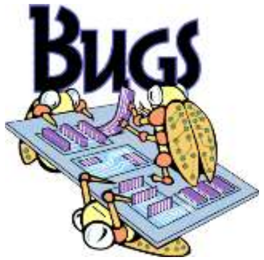
- Router can be partitioned into control and data plane
  - Management plane/ configuration
  - Control plane / Decision: OSPF (Open Shortest Path First)
  - Data plane / Forwarding



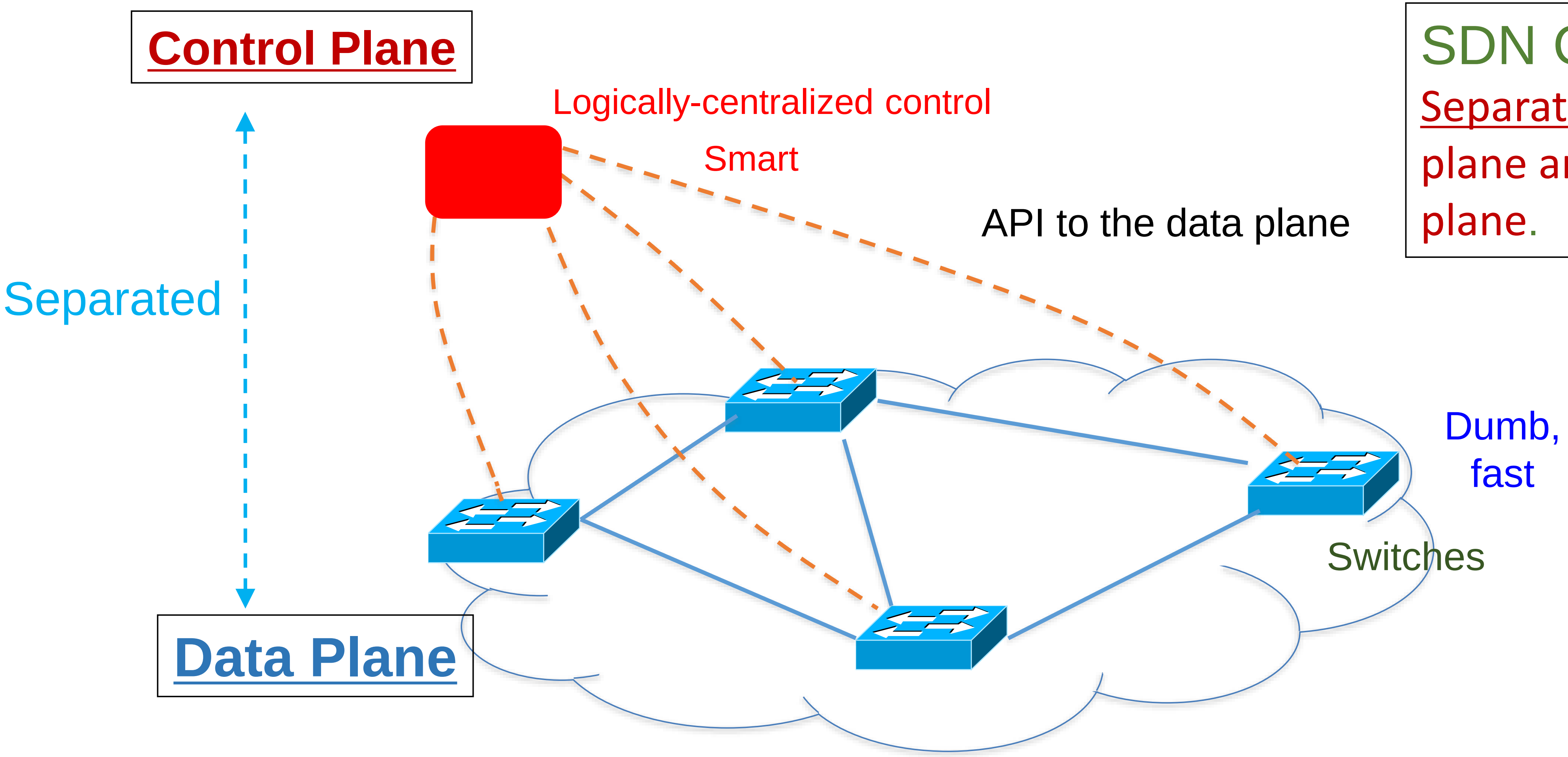
Traditional network Router In Summary

- Typical Networking Software
  - Management plane
  - Control Plane – The brain/decision maker
  - Data Plane – Packet forwarder



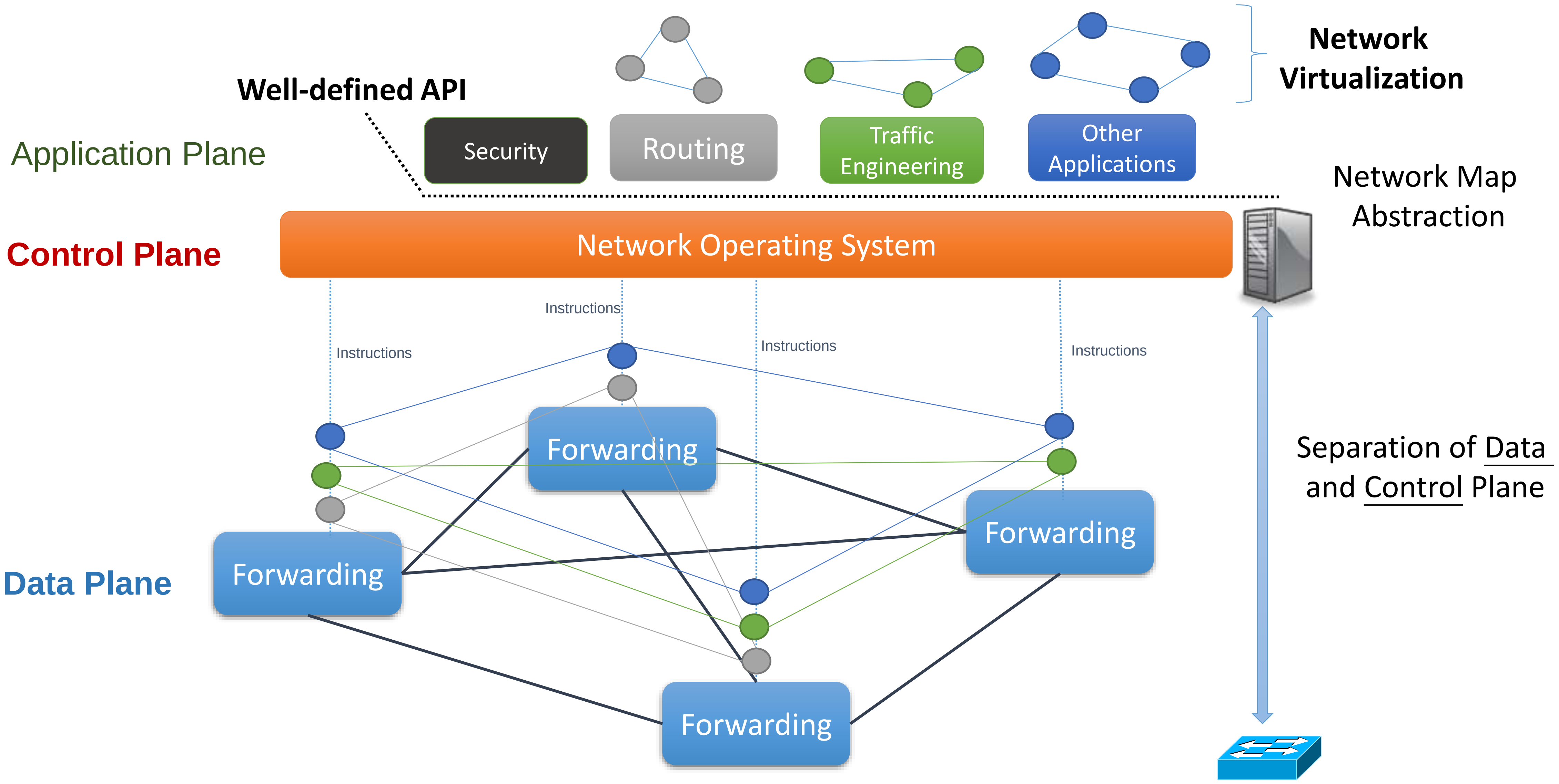


Imagine **IF** The Network is……..!!!



SDN Concept:
Separate *Control* plane and *Data* plane.

Software-Defined Network with key Abstractions



SDN Basic Concept

- Separate Control plane and Data plane entities.
  - Network intelligence and state are logically centralized.
  - The underlying network infrastructure is abstracted from the applications.
- Execute or run Control plane software on general purpose hardware.
  - Decouple from specific networking hardware.
  - Use commodity servers and switches.
- Have programmable data planes.
  - Maintain, control and program data plane state from a central entity.
- An architecture to control not just a networking device but an entire network.

SDN in Real World – Google's Story

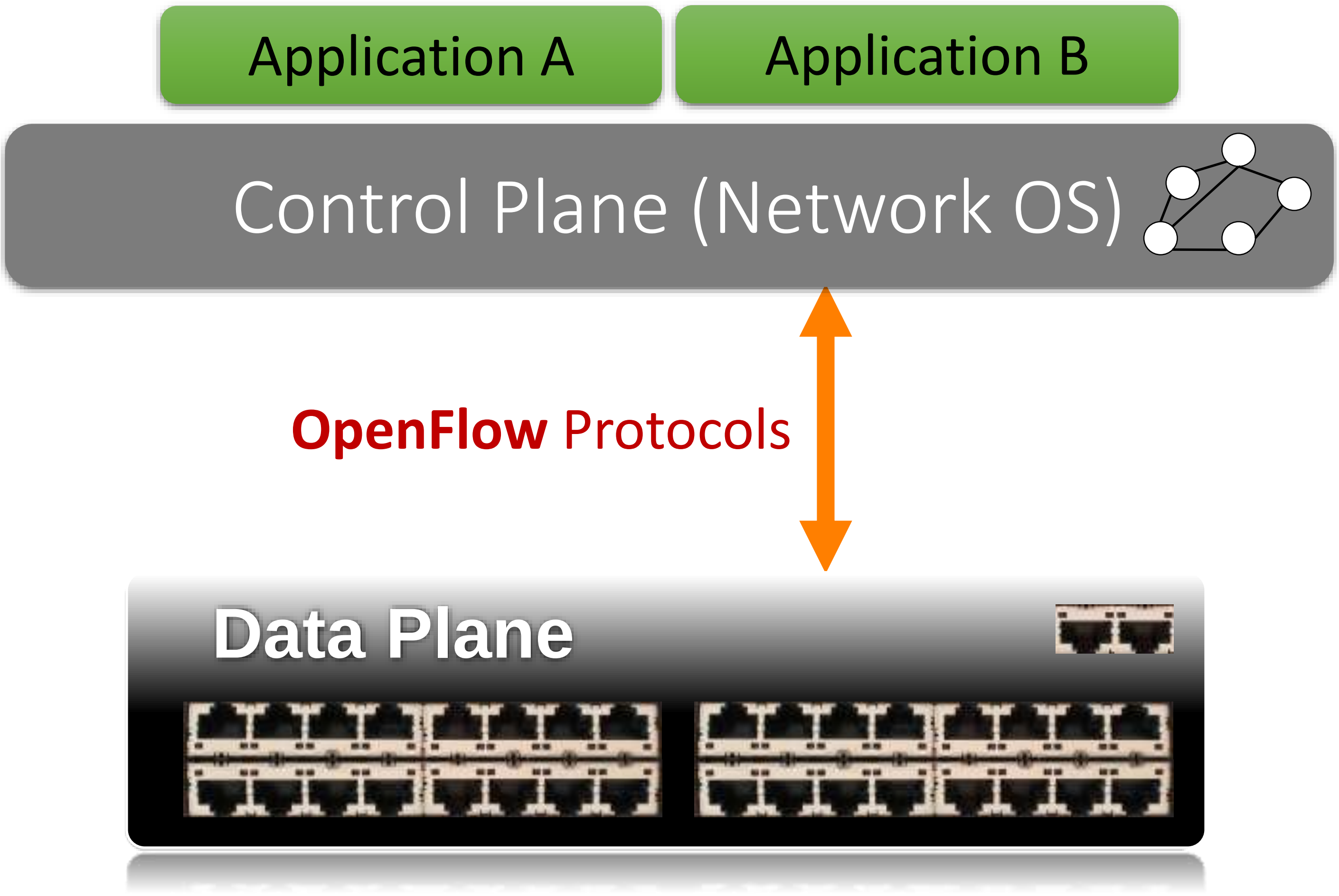
- The industries were skeptical whether SDN was possible.
- Google had big problems:
  - **High financial cost** managing their datacenters: Hardware and software upgrade, over provisioning (fault tolerant), manage large backup traffic, time to manage individual switch, and a lot of men power to manage the infrastructure.
  - **Delay** caused by rebuilding connections after link failure.
    - Slow to rebuild the routing tables after link failure.
    - Difficult to predict what the new network may perform.
- Google went ahead and implemented SDN.
  - Built their hardware and wrote their own software for their internal datacenters.
  - Surprised the industries when Google announced SDN was possible in production.
- How did they do it?
  - Read "*B4: Experience with a Globally-Deployed Software Defined WAN*", ACM Sigcomm 2013.

The Origin of SDN

Martin Casado

- 2006: Martin Casado, a PhD student at Stanford and team propose a clean-slate security architecture (SANE) which defines a centralized control of security (in stead of at the edge as normally done). Ethane generalizes it to all access policies.
- The idea of *Software Defined Network* is originated from OpenFlow project (*ACM SIGCOMM 2008*).
- 2009: Stanford publishes OpenFlow V1.0.0 specs.
- June 2009: Martin Casado co-founds Nicira.
- March 2011: Open Networking Foundation is formed.
- Oct 2011: First Open Networking Summit. Many Industries (Juniper, Cisco announced to incorporate.
- July 2012: VMware buys Nicira for **$1.26B**.
- Lesson Learned: Imagination is the key to *unlock* the power of possibilities.

What is OpenFlow?



What is OpenFlow?

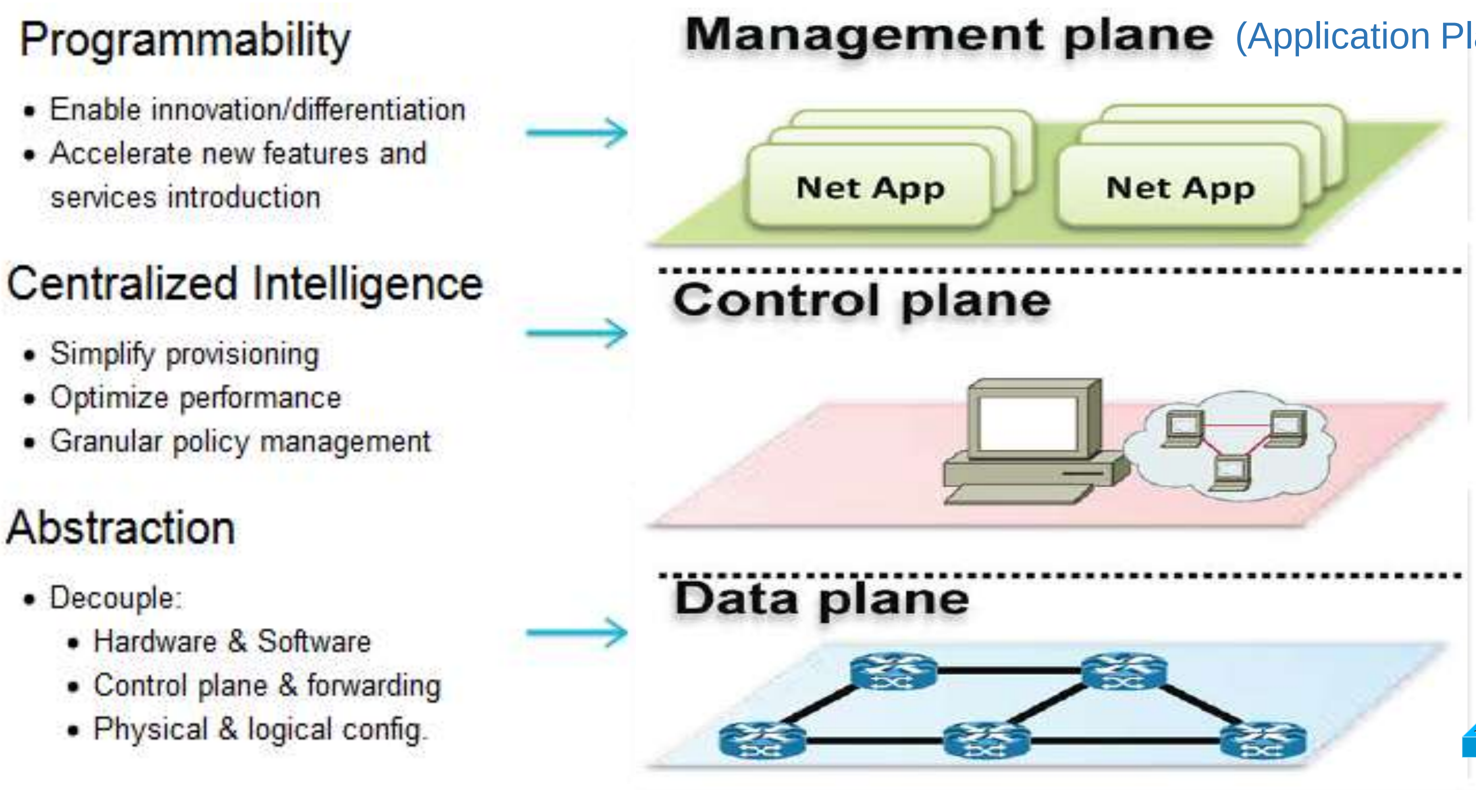
- Allow separation of control and data planes.
- Centralization of control.
- Flow based control.
- Takes advantage routing tables in Ethernet switches and routers.

- SDN is not OpenFlow.
  - **SDN** is a concept of the physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices.
  - ***OpenFlow*** is communication interface between the control and data plane of an *SDN architecture*.
    - Allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual.
    - Think of as a protocol used in switching devices and controllers interface.

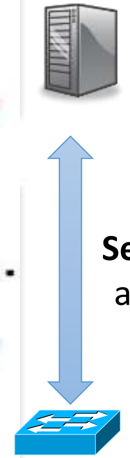
How is OpenFlow related to SDN in The Nut Shell?

OpenFlow allows you to do:

SDN Concept

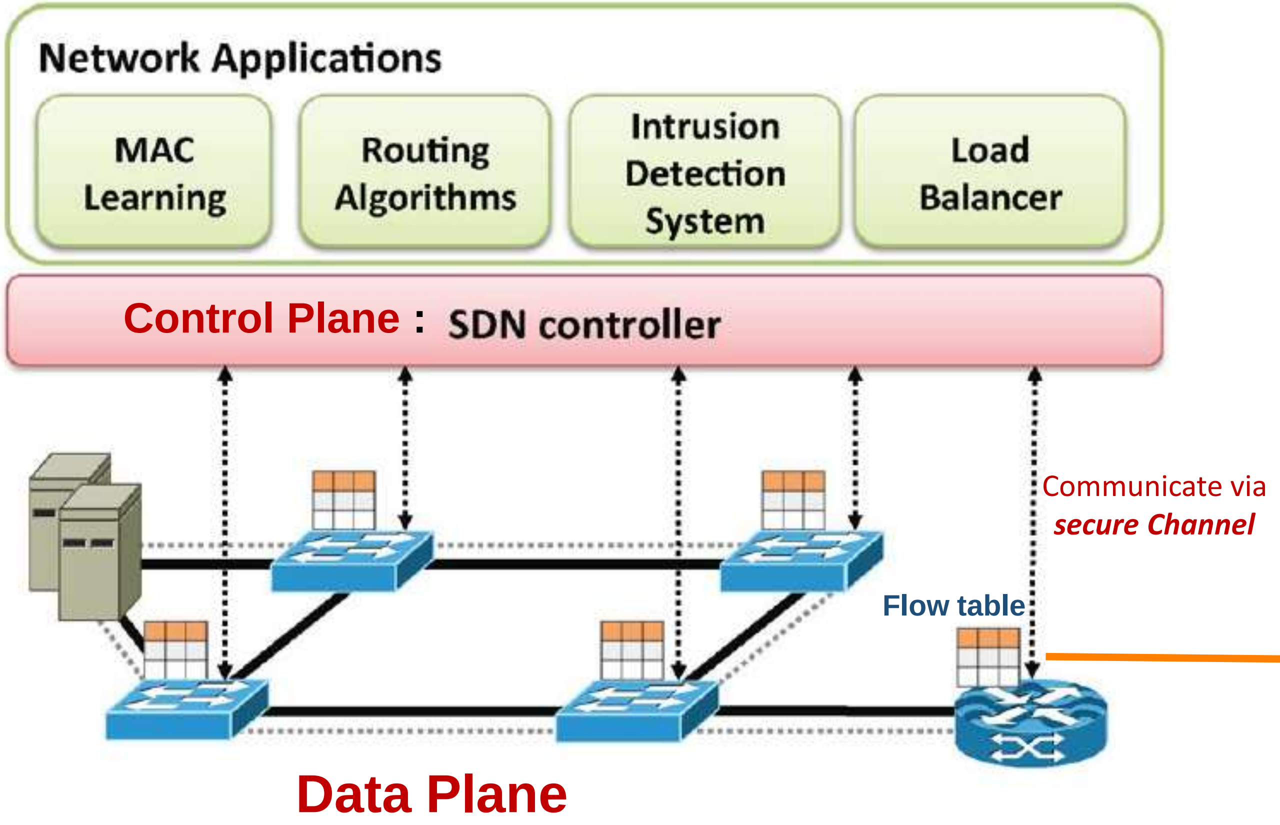


(Application Plane)



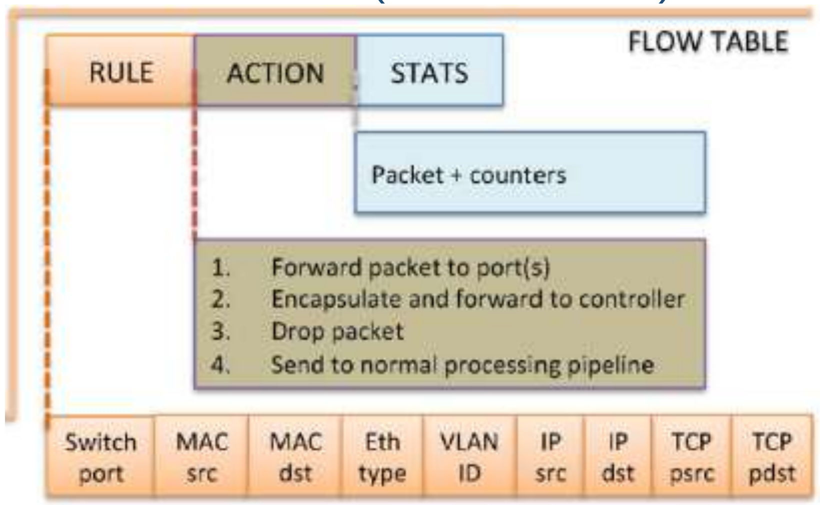
Separation of Data and Control Plane

Basic OpenFlow: How Does it Work?



- Controller manages the traffic (network flows) by manipulating the flow table at switches.
  - Instructions are stored in flow tables.
- When packet arrives at switch, match the header fields with flow entries in a flow table.
- If any entry matches, performs indicated actions and update the counters.
- If Does not match, Switch asks controller by sending a message with the packet header.

Flow Table (has 3 sections)



Match the packet header

The Actual Flow Table Looks Like

Port	Src MAC	Dst MAC	VLAN ID	Priority	EtherType	Src IP	Dst IP	Protocol IP Proto	QoS IP ToS	Src L4 Port ICMP Type	Dst L4 Port ICMP Code	Action	Counter
*	*	0A:C8:*	*	*	*	*	*	*	*	*	*	Port 1	102
*	*	*	*	*	*	*	192.168.*.*	*	*	*	*	Port 2	202
*	*	*	*	*	*	*	*	*	*	21	21	Drop	420
*	*	*	*	*	*	*	*	0x806	*	*	*	Local	444
*	*	*	*	*	*	*	*	0x1*	*	*	*	Controller	1

OpenFlow Table: Basic Actions

- All: To all interfaces except incoming interface.
- Controller: Encapsulate and send to controller.
- Local: send to its local networking stack.
- Table: Perform actions in the next flow table (table chaining or multiple table instructions).
- In_port: Send back to input port.
- Normal: Forward using traditional Ethernet.
- Flood: Send along minimum spanning tree except the incoming interface.

OpenFlow Table: Basic Stats

Per Table	Per Flow	Per Port	Per Queue
Active Entries	Received Packets	Received Packets	Transmit Packets
Packet Lookups	Received Bytes	Transmitted Packets	Transmit Bytes
Packet Matches	Duration (Secs)	Received Bytes	Transmit overrun errors
	Duration (nanosecs)	Transmitted Bytes	
		Receive Drops	
		Transmit Drops	
		Receive Errors	
		Transmit Errors	
		Receive Frame Alignment Errors	
		Receive Overrun erorrs	
		Receive CRC Errors	
		Collisions	

- Provide counter for incoming flows or packets.
- Information on counter can be retrieved to control plane.
- Can be used to monitor network traffic.

Additional Feature to **Rules** and **Stats**

OpenFlow Version	Match fields	Statistics	# Matches Req	# Matches Opt	# Instructions Req	# Instructions Opt	# Actions Req	# Actions Opt	# Ports Req	# Ports Opt
v 1.0	Ingress Port Ethernet: src, dst, type, VLAN IPv4: src, dst, proto, ToS TCP/UDP: src port, dst port	Per table statistics Per flow statistics Per port statistics Per queue statistics	18	2	1	0	2	11	6	2
v 1.1	Metadata, SCTP, VLAN tagging MPLS: label, traffic class	Group statistics Action bucket statistics	23	2	0	0	3	28	5	3
v 1.2	OpenFlow Extensible Match (OXM) IPv6: src, dst, flow label, ICMPv6		14	18	2	3	2	49	5	3
v 1.3	PBB, IPv6 Extension Headers	Per-flow meter Per-flow meter band Threshold	14	26	2	4	2	56	5	3
v 1.4	—	— Optical port properties	**14**	**27**	**2**	**4**	**2**	**57**	**5**	**3**