OF Message Types v1.0

Message Type	Category	Subcategory
HELLO	Symmetric	Immutable
ECHO_REQUEST	Symmetric	Immutable
ECHO_REPLY	Symmetric	Immutable
VENDOR	Symmetric	Immutable
FEATURES_REQUEST	Controller-Switch	Switch configuration
FEATURES_REPLY	Controller-Switch	Switch configuration
GET_CONFIG_REQUEST	Controller-Switch	Switch configuration
GET_CONFIG_REPLY	Controller-Switch	Switch configuration
SET_CONFIG	Controller-Switch	Switch configuration
PACKET_IN	Asynchronous	NA
FLOW_REMOVED	Asynchronous	NA
PORT_STATUS	Asynchronous	NA

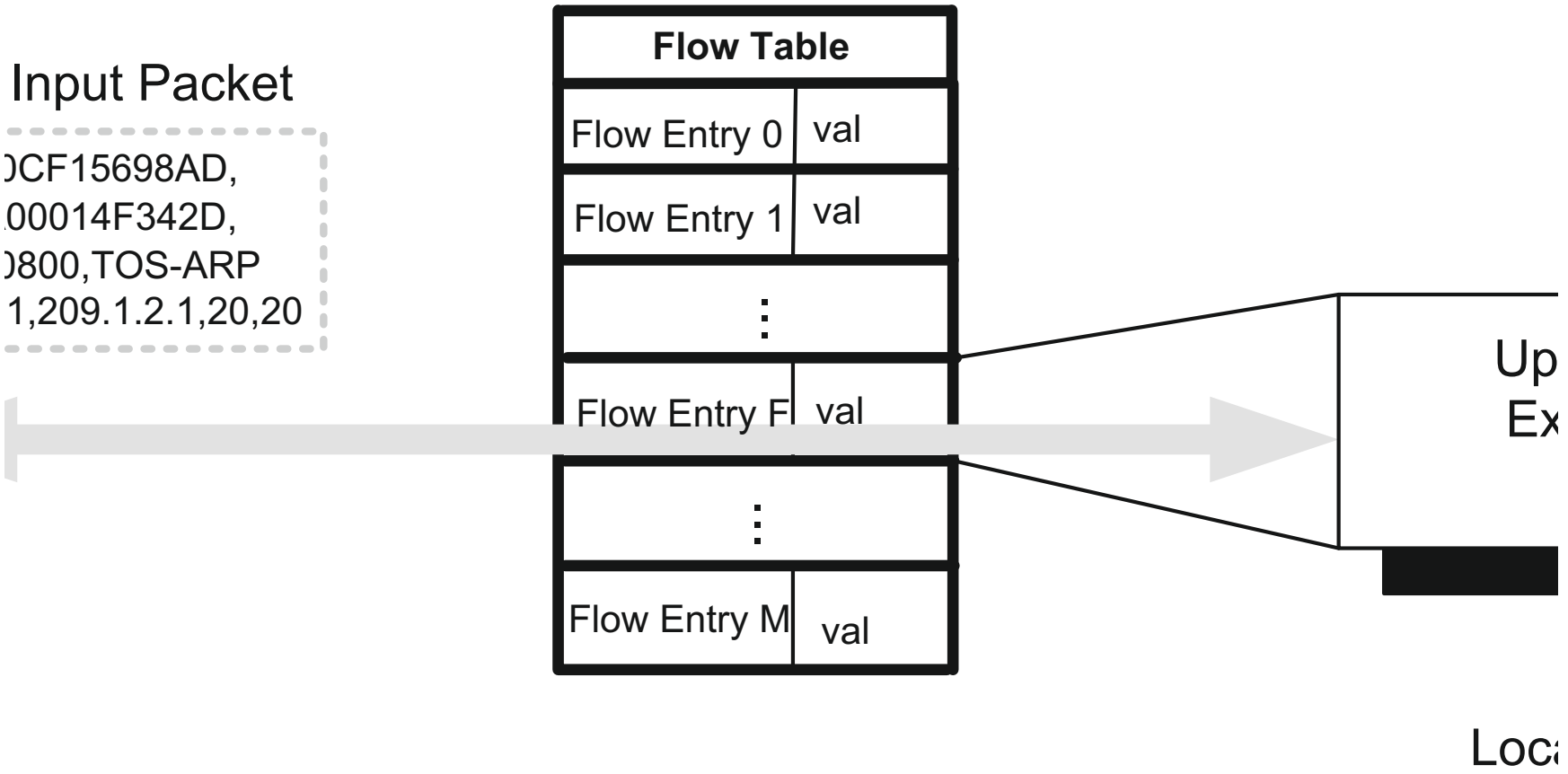
OF Message Types v1.0, contd

Message Type	Category	Subcategory
ERROR	Asynchronous	NA
PACKET_OUT	Controller-Switch	Cmd from controller
FLOW_MOD	Controller-Switch	Cmd from controller
PORT_MOD	Controller-Switch	Cmd from controller
STATS_REQUEST	Controller-Switch	Statistics
STATS_REPLY	Controller-Switch	Statistics
BARRIER_REQUEST	Controller-Switch	Barrier
BARRIER_REPLY	Controller-Switch	Barrier
QUEUE_GET_CONFIG_REQUEST	Controller-Switch	Queue Configuration
QUEUE_GET_CONFIG_REPLY	Controller-Switch	Queue Configuration

OF Message Types

- Immutable
  - Message type will not be changed in future versions
- Barrier Request Message
  - Switch **MUST** complete execution of commands received from the controller **BEFORE** request of Barrier Request
  - **AFTER** completion, switch sends BARRIER REPLY
- Three Phases of Switch Operation
  - Initialization
  - Operation
  - Monitoring
- Controller Failure
  - Switch enters *emergency mode* and resets TCP/TLS conn
  - All flows except those in “emergency flow cache” deleted

Packet matching operation



Sw. Forw. Packet to Controller

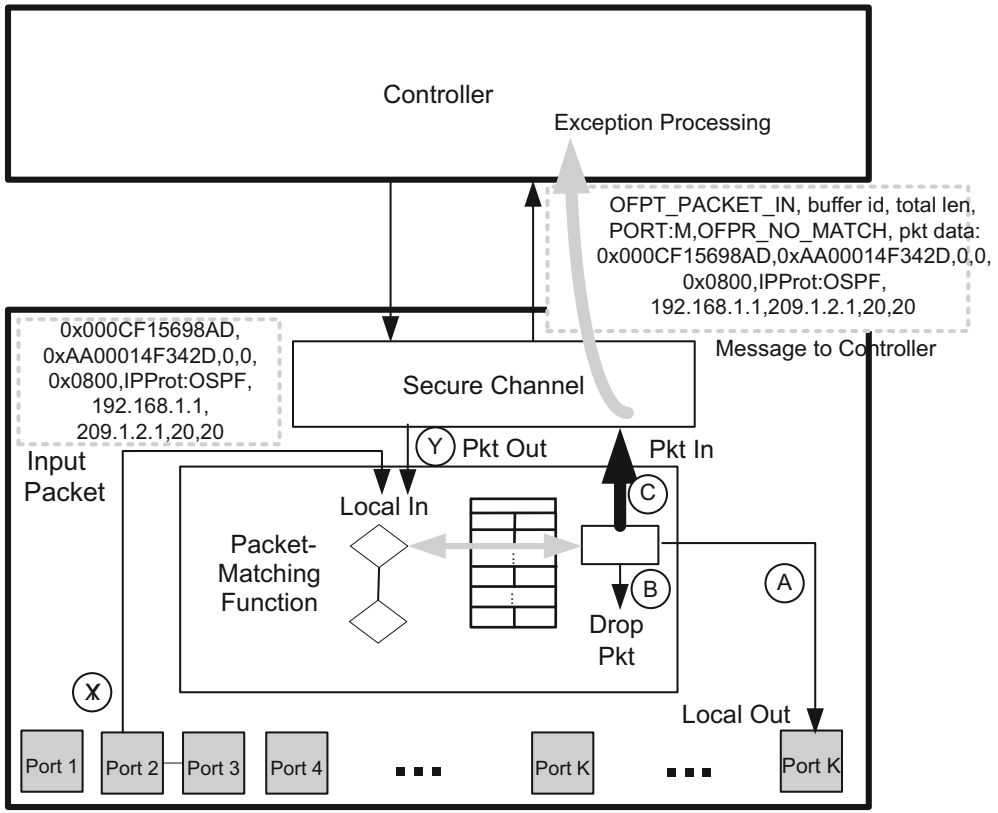


FIGURE 5.11

Switch forwarding incoming packet to controller.

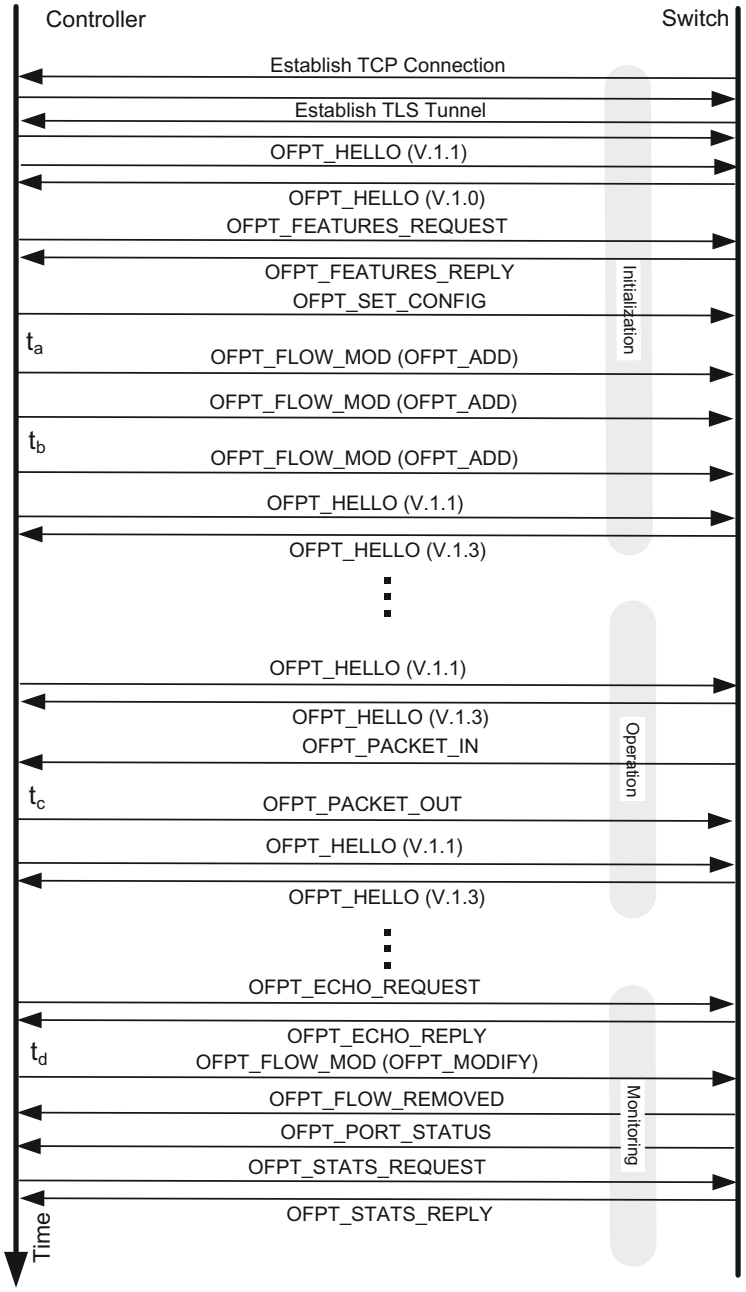


Fig 5.8 from “Software Defined Networks” book by Gorannson & Black, IIT M Library

OPENFLOW V1.1

OpenFlow v1.1

- Released on Feb. 28, 2011
- Addition of ***Multiple Flow Tables,*** as a pipeline
- Separated “Action” execution from Flow Table Entry
- Added notion of “Instruction” protocol object

- Different flow tables can have different matching rules
- FTE can be chained by an instruction in one FT to point to another FT (GOTO)
- For each packet, action set is initialized to empty
  - Each matching rule’s instruction in a given FT adds action items to packet’s action set
- Action set is executed after reaching end of pipeline
  - Alternate: Apply-Action instruction results in execution of some actions between flow tables

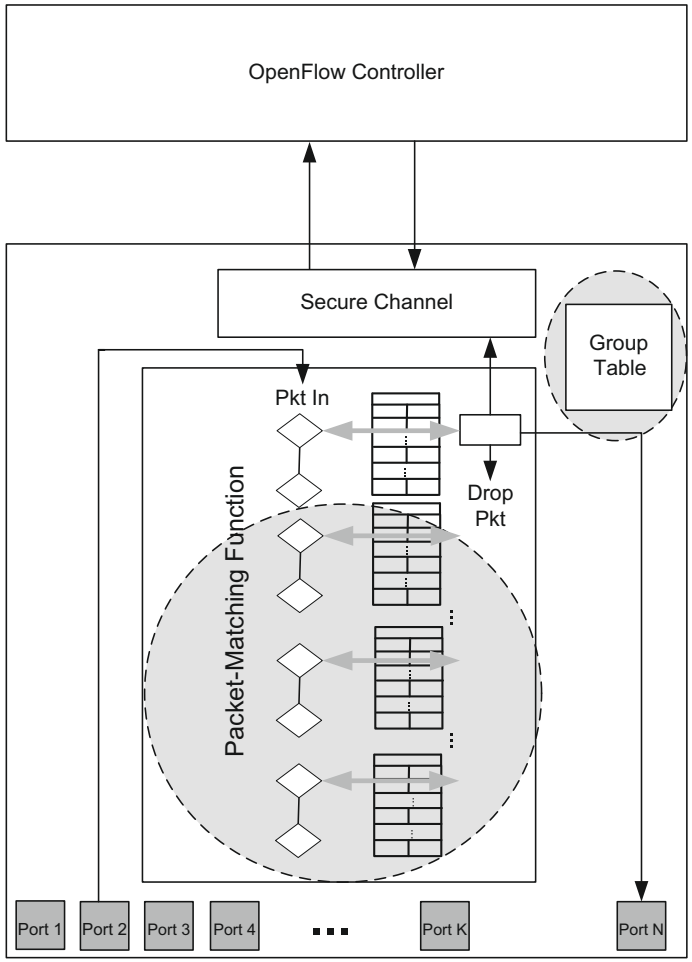


FIGURE 5.12

OpenFlow V.1.1 switch with expanded packet processing.

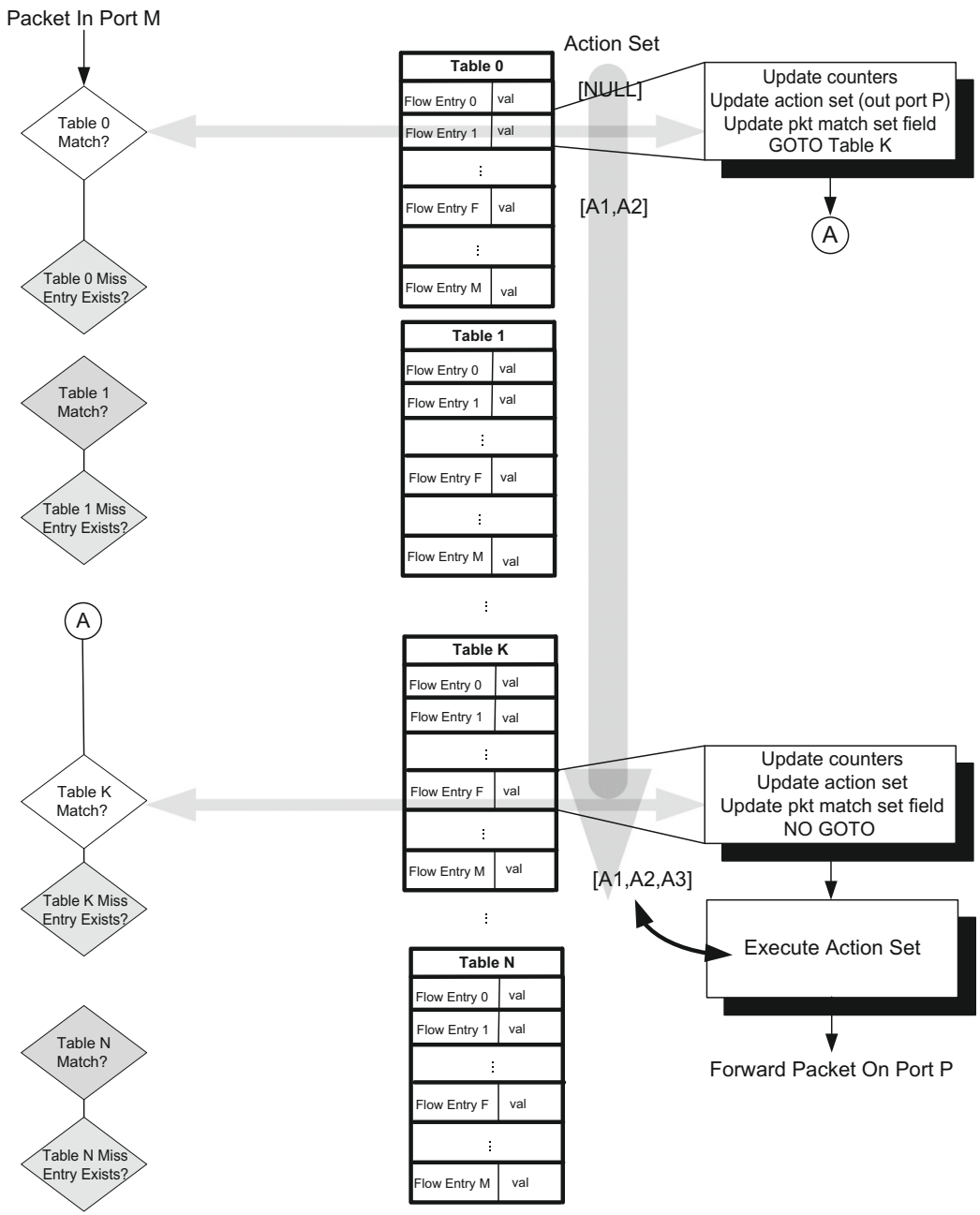


FIGURE 5.14

Packet matching function - basic packet forwarding, V.1.1.

OF 1.1: Action Order

- Copy TTL inward
- Pop (Outermost packet header is popped)
- Push (New header is inserted in front of current outermost header)
- Copy TTL outward
- Decrement TTL
- Set: Apply all set_field actions to the packet
- QoS: Apply all QoS actions to the packet, such as set_queue
- Group: If a group action is specified, apply the actions to the relevant action buckets
  - Can result in the packet being forwarded out the ports corresponding to those buckets
- Output: If no group action is specified, forward the packet out the specified port

OF 1.1: Groups

- Switch has a "**Group Table**"
  - GT has set of *group entries*
  - Entry has one of more *action buckets*
  - Buckets specify actions to be applied to packet before forwarding packet on specified port
- Multicast: defining group as a set of ports
- A group's buckets can fwd to other groups

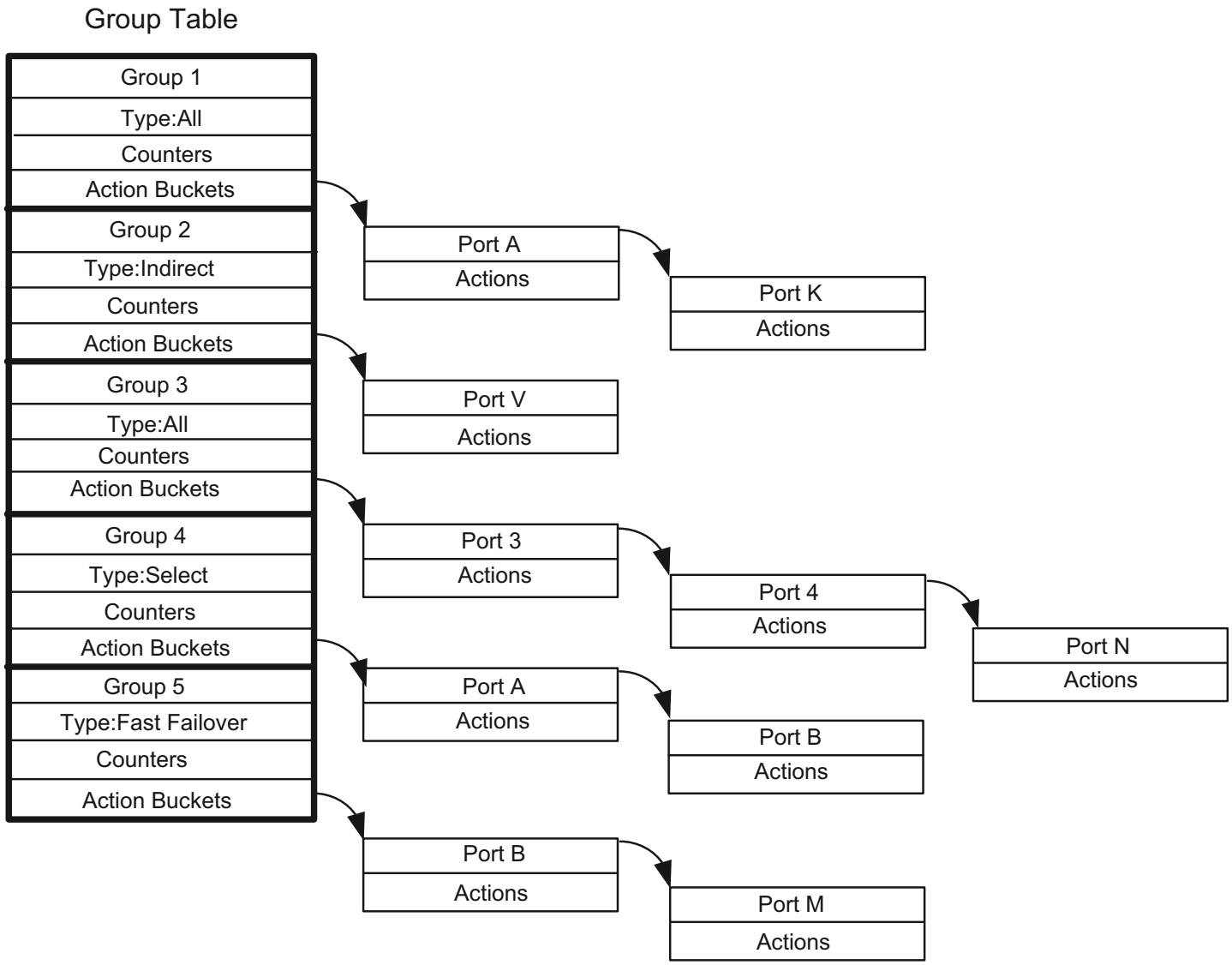


FIGURE 5.13

OpenFlow V.1.1 group table.

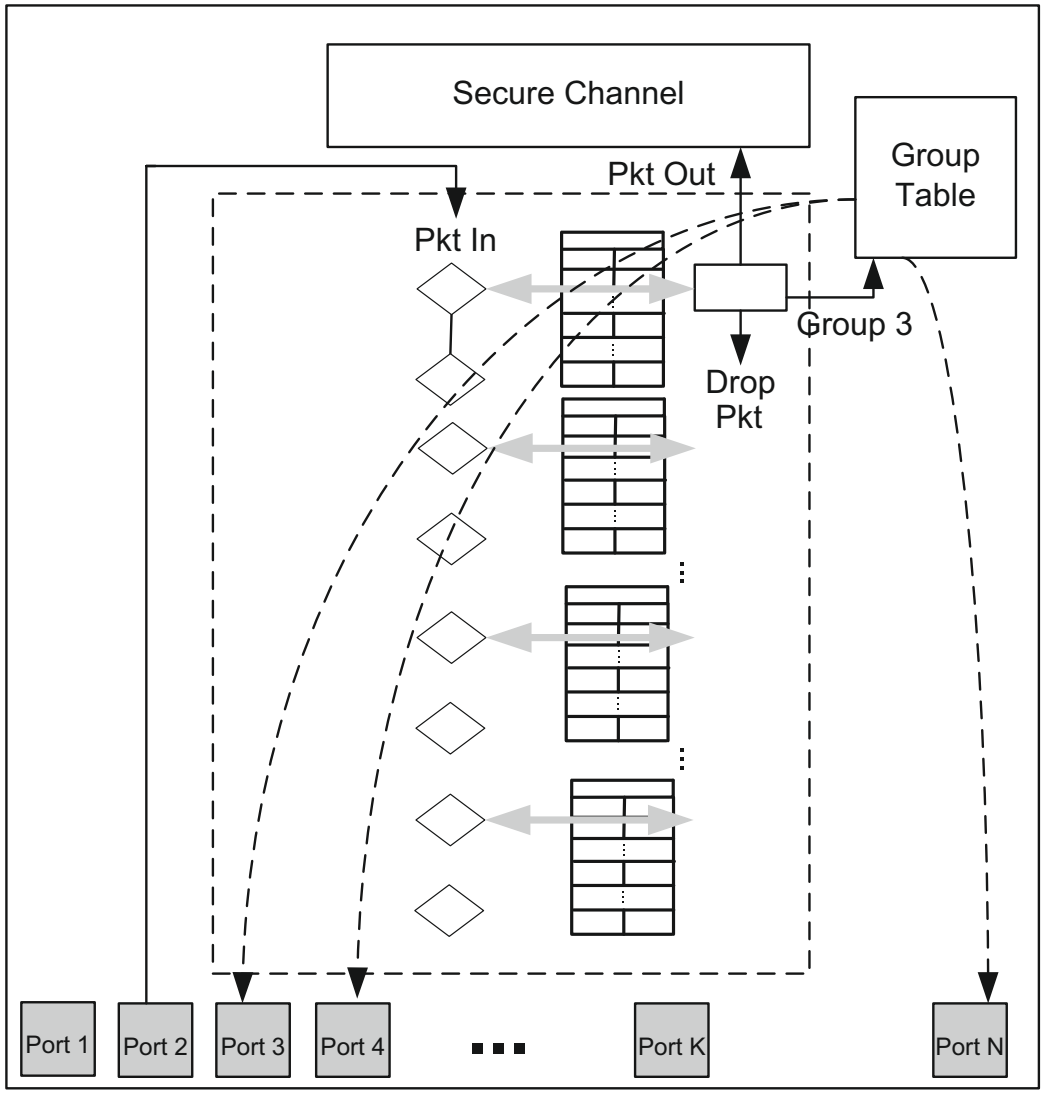


FIGURE 5.15

Multicast using group table in V.1.1.

Support for MPLS and VLAN Tags

- Added full VLAN support
- Supports pushing and popping of MPLS Labels and VLAN Tags
- PUSH: A new header is inserted in front of outermost header
  - Fields are copied from current outermost header, if present
  - Fields can be set with SET action
- POP: Remove outermost header
- Requires extensible match support

Controller Conn. Failure

- After loss of connection to controller, switch enters one of:
  - Fail secure mode
  - Fail standalone mode
  - Based on what switch supports/configured to support
- Fail secure mode: Operates normally, except that messages to controller are dropped
- Fail standalone: Switches stops OF behavior and changes to native underlying switch/router mode
- After reconnect to controller, normal OF operation starts

OPENFLOW V1.2

Table 5.3 Major New Features Added in OpenFlow 1.2

Feature Name	Description
Extensible match support	See Section 5.5.1
Extensible set_field packet-rewriting support	See Section 5.5.2
Extensible context expression in “packet-in”	See Section 5.5.3
Extensible error messages via experimenter error type	-
IPv6 support	See Section 5.5.2
Simplified behavior of flow-mod request	-
Removed packet parsing specification	See Section 5.5.1
Multiple controller enhancements	See Section 5.5.4

OF v1.2 Features

- OpenFlow Extensible Match (OXM) descriptor added
  - generic and extensible packet-matching capability
- OXM defines a set of Type-Length-Value (TLV) tuples to describe a matching field
  - Any header field of Ethernet, VLAN, MPLS, IPv4, and IPv6 switching and routing may be selected (with bitmask wildcard capability)
  - Earlier versions had more static descriptors
  - EXPERIMENTER match class defined: matches on fields in the packet payload
- Extensible SET_FIELD Packet Rewriting Support
  - set the value of any field in the packet header that may be used for matching

OXM

- Match packet header
  - Type field (16 bits)
  - Length field – Total length of Match packet (in bytes) – 16 bits
  - Pad bytes to align total match packet to 8 bytes
  - 0 or more OXM TLVs
- OXM TLV is 5 to 259 bytes long
- Header of OXM is 4 bytes long
  - oxm_class (16 bits): Class (e.g. OFPXMC_OPENFLOW_BASIC)
  - oxm_field (7 bits): identifies one of the match types within the match class
  - oxm_hasmask (1 bit): set if OXM include a bitmask in payload
  - oxm_length (8 bits): length of OXM payload
  - Example: oxm_type=OXM_OF_ETH_TYPE, oxm_hasmask=0, and oxm_value=0x0800

OF v1.2 Features, contd.

- Context information for Packet_IN message
  - Extended to include input virtual port, input physical port, and metadata built during packet matching
- Support for multiple Controllers
  - Addresses high-availability (HA) requirement
  - Switch can connect to multiple Controllers at the same time
  - Relative to a switch, a Controller can be: Equal, Master or Slave
- Equal and Master Modes
  - Allow controller full ability to program the switch
  - Master: Only One can be in this Mode; others are in Slave Mode
- Slave Mode: controller can request data from the switch, such as statistics, but no modifications

OPENFLOW V1.3

Table 5.4 Major New Features Added in OpenFlow 1.3

Feature Name	Description
Refactor capabilities negotiation	See Section 5.6.1
More flexible table-miss support	See Section 5.6.2
IPv6 extension header-handling support	-
Per-flow meters	See Section 5.6.3
Per-connection event filtering	See Section 5.6.4
Auxiliary connections	See Section 5.6.5
MPLS BoS matching	Bottom-of-stack bit (BoS) from the MPLS header may now be used as part of match criteria
Provider backbone bridging tagging	See Section 5.6.7
Rework tag order	Instruction execution order now determines tag order rather than it being statically specified
Tunnel-ID metadata	Allows for support of multiple tunnel encapsulations
Cookies in PACKET_IN	See Section 5.6.6
Duration for stats	The new *duration* field allows more accurate computation of packet and byte rate from the counters included in those statistics
On-demand flow counters	Disable/enable packet and byte counters on a per-flow basis

OF v1.3

- Released in Apr. 2012: important (later: v.1.3.1, 1.3.2)
- Many stable implement. based on v1.3
- Main changes
  - Refactor capabilities negotiation
  - More flexible table miss support
  - IPv6 Extension Header handling support
  - Per-flow meters
  - Per-connection event filtering
  - Auxiliary connections
  - MPLS Bottom-of-Stack (BoS) matching
  - Provider Backbone Bridging (PBB) tagging
  - Rework tag order
  - Tunnel-ID metadata
  - Cookies in packet-in
  - On-demand flow counters

OF v1.3 Features

- Table Miss Support
  - So far, only three options for Table Miss
    - Send to Controller
    - Send to Next Table
    - Drop the Packet
  - V1.3 adds notion of wildcard in a table that will match all missed packets
  - Suitable action can be defined

V1.3 Meter

- Metering is used to limit bandwidth usage of a flow
- Meter table stores metering information
  - Eac h meter has unique ID
  - Rate-limiting meters
  - Multiple meter bands attached to a meter ID
  - Each band has bandwidth rate and type
  - For a given packet, only one band is applied
    - Highest BW rate band that is lower than the current measured BW for a flow
  - meter-level counters and meter-band-level counters

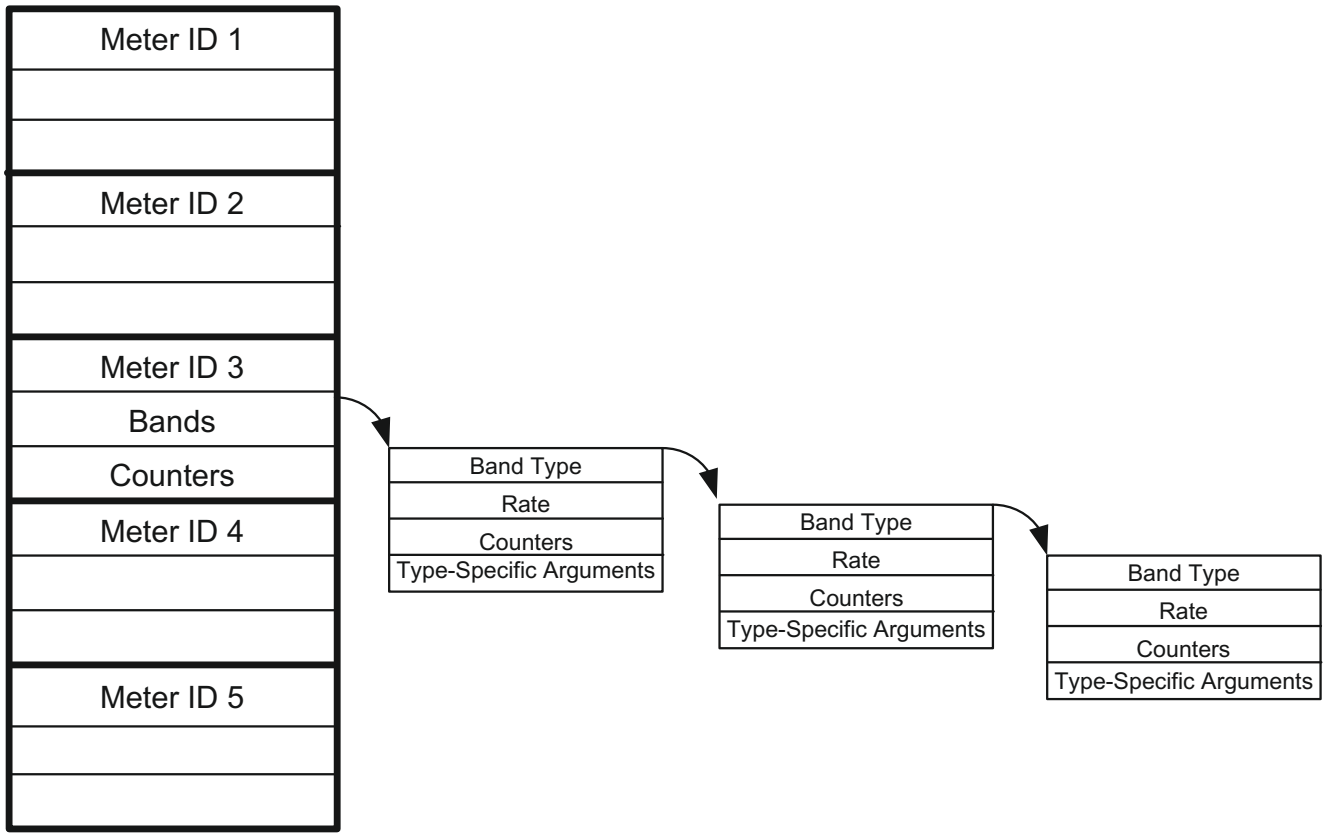


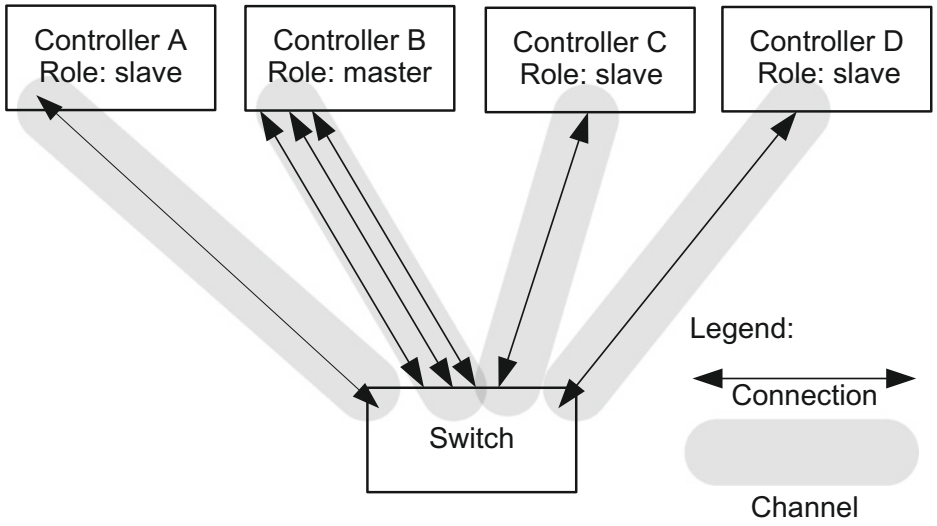
FIGURE 5.16

OpenFlow V.1.3 meter table.

Per Connection Event Filter

- When a switch talks to multiple controllers
  - Switch replies only to ctrller that initiated request
  - However, asynch. Messages from switch go to all controllers
  - V1.3 provides a SET_ASYNC message to inform switch about which packets it wants to receive
    - In addition, ctrller can filter packets not of interest

Multiple Connections



- Switch can open multiple connections to the Controller
- First one is TCP/TLS – others can be TCP/UDP etc.
- To avoid performance issues due to TCP window close
- To send high priority messages on main TCP and lower priority message on other channel

Cookies in PACKET_IN

- Switch matches a packet against FT and when it sends a PACKET_IN to Cntrler
  - Also, computes a cookie and sends along with packet; also stored in flow table entry
  - Controller caches the cookie and the flow table entry pointed by this cookie the first time for this packet
  - Now on, controller can avoid packet matching by using cookie

OPENFLOW V1.4

OF v1.4

- October 2013 (https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf)
- Changes
  - More extensible wire protocol
  - More descriptive reasons for packet-in
  - Optical port properties
  - Flow monitoring
  - **Eviction of flow rules**
  - **Flow table vacancy events**
  - **Bundle mechanism (Atomic set of commands to switch)**
  - Synchronized tables
  - Change default TCP port to 6653 (from 6633)

```
/* OXM Flow match field types for OpenFlow basic class. */
enum oxm_ofb_match_fields {
OFPXMT_OFB_IN_PORT
OFPXMT_OFB_IN_PHY_PORT
OFPXMT_OFB_METADATA
OFPXMT_OFB_ETH_DST
OFPXMT_OFB_ETH_SRC
OFPXMT_OFB_ETH_TYPE
OFPXMT_OFB_VLAN_VID
OFPXMT_OFB_VLAN_PCP
OFPXMT_OFB_IP_DSCP
OFPXMT_OFB_IP_ECN
OFPXMT_OFB_IP_PROTO
OFPXMT_OFB_IPV4_SRC
OFPXMT_OFB_IPV4_DST
OFPXMT_OFB_TCP_SRC
OFPXMT_OFB_TCP_DST
OFPXMT_OFB_UDP_SRC
OFPXMT_OFB_UDP_DST
OFPXMT_OFB_SCTP_SRC
OFPXMT_OFB_SCTP_DST
OFPXMT_OFB_ICMPV4_TYPE
OFPXMT_OFB_ICMPV4_CODE
OFPXMT_OFB_ARP_OP
OFPXMT_OFB_ARP_SPA
OFPXMT_OFB_ARP_TPA
OFPXMT_OFB_ARP_SHA
OFPXMT_OFB_ARP_THA
OFPXMT_OFB_IPV6_SRC
OFPXMT_OFB_IPV6_DST
OFPXMT_OFB_IPV6_FLABEL
OFPXMT_OFB_ICMPV6_TYPE
OFPXMT_OFB_ICMPV6_CODE
OFPXMT_OFB_IPV6_ND_TARGET
OFPXMT_OFB_IPV6_ND_SLL
OFPXMT_OFB_IPV6_ND_TLL
OFPXMT_OFB_MPLS_LABEL
OFPXMT_OFB_MPLS_TC
OFPXMT_OFP_MPLS_BOS
OFPXMT_OFB_PBB_ISID
OFPXMT_OFB_TUNNEL_ID
OFPXMT_OFB_IPV6_EXTHDR
OFPXMT_OFB_PBB_UCA
```

OF v1.5

- Work in Progress
- Not standardized as of Sep. 2015

Summary of OF Features

SDN Element	OpenFlow switch specification improvements	OF version
Separation of control and data plane		
Logically centralized controller		
Northbound APIs		
Programmability		
Flow Entries		

https://www.opennetworking.org/images/stories/downloads/sdn-resources/IEEE-papers/evolution-of-sdn-and-of.pdf

Other SDN Control Architectures

- Juniper's Contrail Controller (Linux)
  - XMPP as control plane
  - L2 and L3 virtual networks
  - Contributions to OpenDaylight
  - OpenContrail Network Virtualization project
- Cisco's Open Network Environment (ONE)
  - Centralized software controller
  - Programmable data plane
  - Ability to provide virtual overlays