Lecture 6 OpenFlow

OpenFlow = switch model + protocol

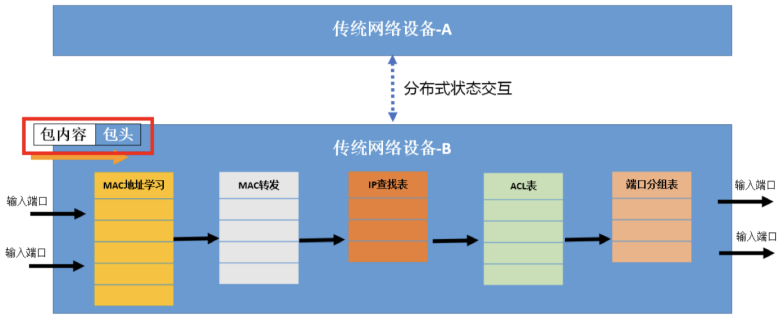
- OpenFlow defines *how switch forwards packets* and *how switches interact with the controller*
- SDN Controller ⇔ OpenFlow Protocol ⇔ Switch Model
- reference
  - OpenFlow1.0.0
  - OpenFlow1.3.4

OpenFlow Date Plane

Traditional Switch Model

1. 传统网络设备：

input port → MAC地址学习表 → MAC转发表 → IP查找表 → ACL表 → 端口分组表 → output port

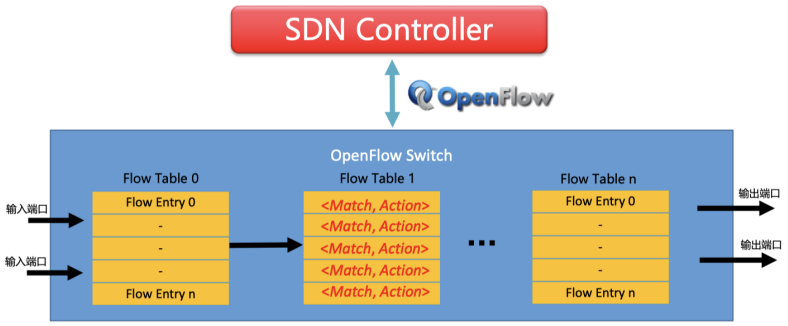


2. ACL表：

- ACL stands for Access Control List.
- It's a mechanism in computer security that defines who is allowed to access a particular resource or perform a certain action.
- ACLs are commonly used in computer networks, file systems, and various other applications where controlling access to resources is important for security purposes.

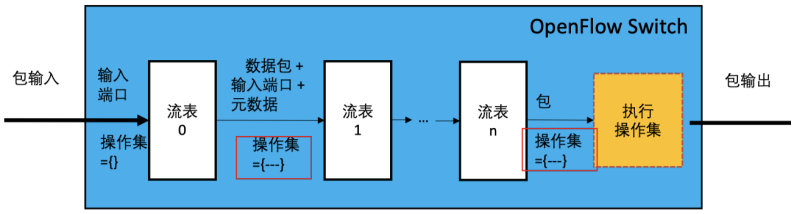
OpenFlow Switch Model

- input port
- → Flow Table 0 : Flow Entry 0, ……
- → Flow Table 1 : <match, action>, ……
- → Flow Table 2 : Flow Entry 0, ……
- → output port



Flow Table Pipeline

- 每经过一个流表，就在操作集中增添一个action，数据部分对应读取
- 最后有一个“执行操作集”，执行初始“包输入”的全部actions



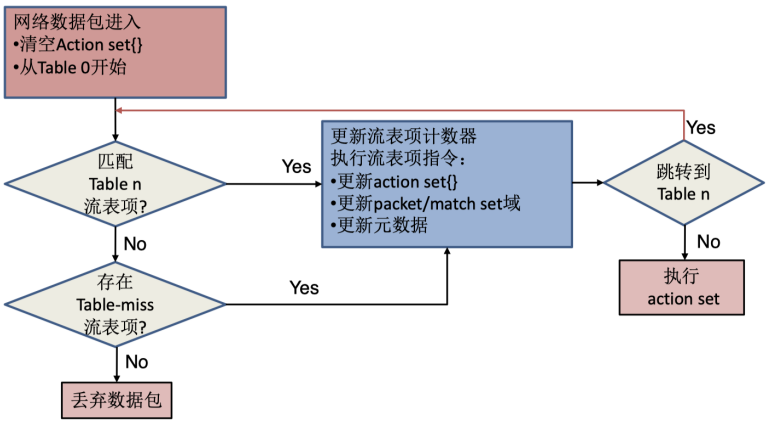
Flow Table Processing

网络数据包进入

- 清空Action Set{}
- 从Table 0开始

匹配 Table i 的流表项?

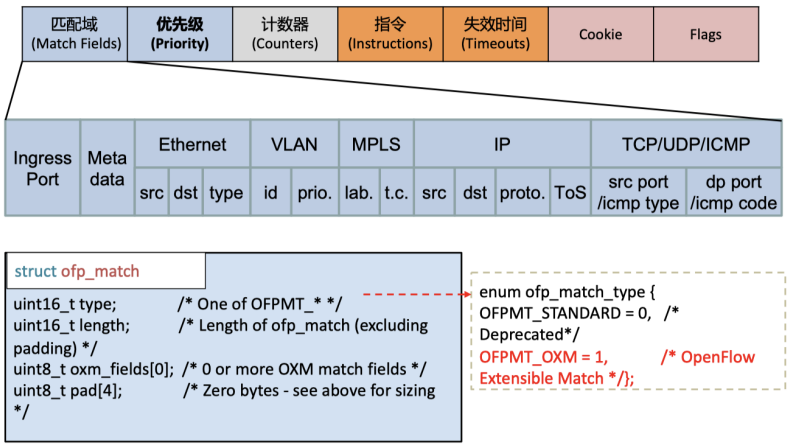
- Yes
  - 更新流表项counter
  - 执行流表项的指令
    - 更新 Action Set{}
    - 更新 Packet / Match Set Domain
    - 更新元数据
  - 跳转到 Table i
    - Yes：重新匹配 Table i 流表项，重新进入上述步骤
    - No：执行Action Set
- No
  - 存在Table-Miss流表项？（负责兜底）
    - Yes: 回到上一个yes
    - No: 丢弃数据包



OpenFlow Entry: Match Fields

Structure of "message" :

- Match Fields：匹配域
- Priority：优先级
- Counters：计数器
- Instructions：指令
- Timeouts：失效时间
- Cookie
- Flags



Match Fields

- Ingress Port (layer2): not in packet "message"
- Meta Data：
  - 是描述数据的数据，是关于数据的属性和特征的信息，而不是数据本身
  - 元数据可以提供有关数据的各种信息，例如其类型、格式、大小、创建时间、修改时间、所有者.....
- Ethernet
  - src
  - dst

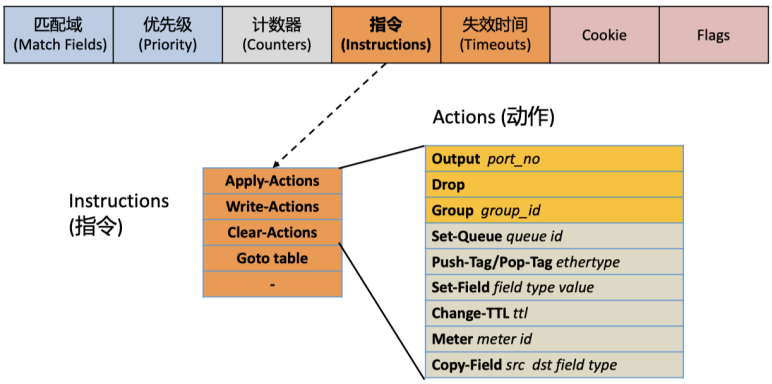
    - type
- (VLAN)
- (MPLS)
    - 多协议标签交换 (Multiprotocol Label Switching). MPLS是一种 *基于标签的转发* 技术，用于提高数据包在网络中的转发效率和灵活性
    - MPLS通过 *为数据包添加一个标签（Label）来识别和转发数据*，而不是仅仅依赖于目的地址。这个标签在网络中的路由器上被用来确定数据包的转发路径，从而加快了数据包的传输速度和网络的整体性能。与传统的IP路由相比，MPLS提供了更快速、更可靠的数据传输。
    - MPLS的 *主要优势之一是它可以创建虚拟专用网络*（VPN），使得 *不同的用户或组织可以共享相同的物理网络基础设施*，同时保持彼此的数据传输隔离和安全性。
- IP
    - src
    - dst
    - proto.
    - ToS：Type of Service
- TCP / UDP / ICMP
    - src port / icmp type
    - dp port / icmp code

OpenFlow Entry: Instructions

> - Instructions → Action
> - Instruction 是对Action的操作

Instructions Contains

- Apply-Actions（执行）
- Write-Actions（写）
- Clear-Actions（清除）
- Goto Table（跳转）



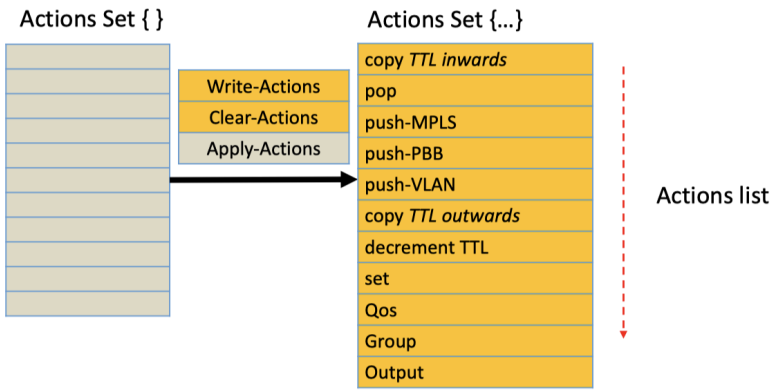
Process in Details

must support	description
Goto-Table ⇒ *next table id*	指示processing pipeline中的下一个table，其中table-id 必须大于当前的table-id，即向后跳转，最后一个table不能包含该Instruction
Write-Actions ⇒ *actions*	把指定的actions添加到当前的action set，如果当前set中 存在系统类型的action，就重写并添加
Clear-Action	立即清除action set中的所有actions

must support	description
Output ⇒ *port_no*	这种action转发packet到指定openflow端口，OFS必须支持转发到物理端口，switch定义的logical端口和required 保留端口
Drop	没有明确的action时，Drop
Group *group_id*	通过指定的group处理packet，精确地解释跟group type 有关

Action Set

Actions 顺序执行：

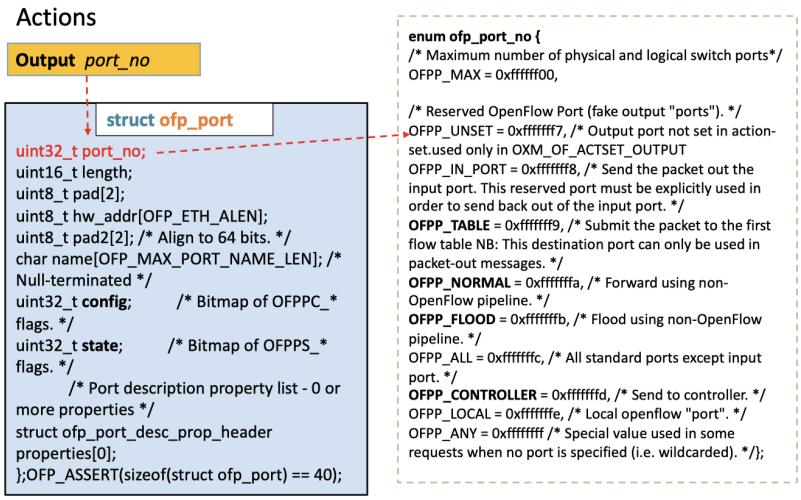


Why Instructions and Actions?

- OpenFlow 1.0 has only a single flow table
  - All wildcard entries have a priority associated with them
- Exposing multiple tables has many advantages
  - Better Flexibility: a lot of software has multiple tables internally (L2/L3)
  - Smaller Table: forcing orthogonal processing (MAC, routing, ACL) into a single table creates huge ruleset due to cross product of rules
- OpenFlow 1.1 introduces a more flexible pipeline with multiple tables
  - packets are processed through the *pipeline*
  - as a packet goes through the pipeline, it's associated with an *action set*
  - Actions which were directly attached to flows in previous versions are now encapsulated(解封装) in instructions

- Instructions may apply those actions between tables or accumulate them in the packet action set

Port Model



port有些并不是真实的物理端口，而是逻辑端口（controller port / flooding port）

must support	description
All	描述通用转发模型中能 *转发某个指定数据包的所有端口* ，只能用作输出端口。在这种情况下，这个数据包会 *被复制然后发送给所有的标准端口* ，当然不包括数据包的输入端口和配置为OFPPC_NO_FWD的端口
CONTROLLER	描述控制器的控制通道， 可用作输入/输出端口：当用在输出端口，将数据包 *封装在 packet-in消息* 中，按照协议规定的方式发送出去；当用在输入端口，标识一个 *来自控制器的数据包*
TABLE	描述通用转发模型处理流水线的起始位置，只有在packet-out消息操作列表的output操作中才有效，将数据包提交给流水线的第一个flow table来处理
IN_PORT	描述数据包的输入端口，只能用作输出端口，发送数据包到自己的输入端口
ANY	当OpenFlow命令没有指定端口时使用的类型，不能用作输入端口和输出端口

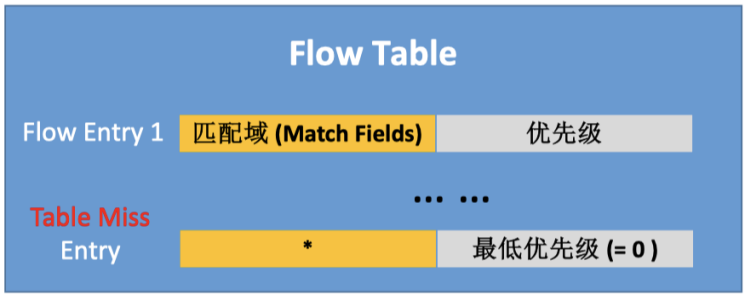
optional support	description
LOCAL	描述通用转发模型 *本地的网络栈和管理栈* ， 可用作输入/输出端口。 使得远端设备通过 OpenFlow网络本身与转发模型交互，使用其网络服务，而不是通过一个独立的控制网络。 采用一些默认流选项可以实现网内控制器链接，不需要独立的控制器通道。
NORMAL	描述 *传统的非OpenFlow转发处理流水线* ，只能用作输出端口，使用传统流水线处理数据包。当转发模型不支持从OpenFlow流水线到传统处理流水线的转发时，必须指定不支持这种操作
FLOOD	描述传统处理 *流水线中的泛洪操作* ，只能用作输出端口，通常发送数据包给所有的标准端口， 不包括输入端口和OFPPS_BLOCKED状态的端口。转发模型需要用数据包VLAN ID选择向那些端口执行flood操作

OpenFlow Entry: Priority

Priority: 0~65535, **larger values** indicate **higher priority**

可以解决多个domain优先级匹配的问题

- Every flow table must support a *table-miss* flow entry to process table misses
  - **wildcards** all match fields (all fields omitted) and has the **lowest priority**
- The table-miss flow entry behaves in most ways like any other flow entry
  - it *doesn't exist by default* in a flow table
  - the controller may add it or remove it by protocol at any time
  - if the table-miss flow entry doesn't exist, by default packets unmatched by flow entries are dropped (discarded)
    - review: OVS 指令



OpenFlow Entry: Timeout

- hard timeout (default set to 0)
  - 人为硬性规定的删除时间
- soft timeout (default set to 1min)
  - 如果一个表空闲的时间达到阈值，则删除

the flow entry will timeout after *idle_timeout* seconds with no traffic, or *hard_timeout* seconds, whichever comes first

OpenFlow Entry: Counter

Per Flow Entry (针对这个流表而言)

- Received Packets
- Received Bytes
- Duration (s, ns)

OpenFlow Switches in Real World

Hardware-based Switch

- Commercial hardware switches with OpenFlow capability

    - Network abstraction is realized by firmware upgrading (固件升级)
- Show **high processing speed**
- Have *space limitation* on saving the flow table entries
- Not easy to upgrade
    - Most switches only support OpenFlow up to version 1.0

Software-based Switch

- OpenFlow enabled software switch (runs on x86 commodity computer)
- Performance is **relatively low**
- Store large amount of flow entries *without bound*
- Under active development, support most recent OpenFlow spec

Software-based OpenFlow Switches

OpenFlow Software Switch

- An OpenFlow compatible user-space software switch implementation
- original code is based on the Stanford 1.0 reference switch
- The implementation is feature-complete, and passes the available oftest 1.1 test case
- CPqD version supports OpenFlow up to v1.3

Open vSwitch (OVS)

Preface

- A virtual switch
- User-Space: configuration, control
- Kernel-Space: datapath (including in main Linux Kernel from v3.3)

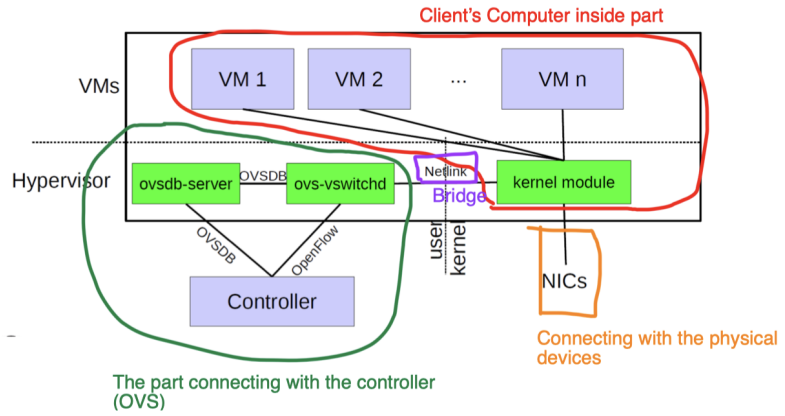
Widely Used

- Most popular OpenStack networking backend
- Default network stack in XenServer
- Thousands of subscribers to OVS mailing lists

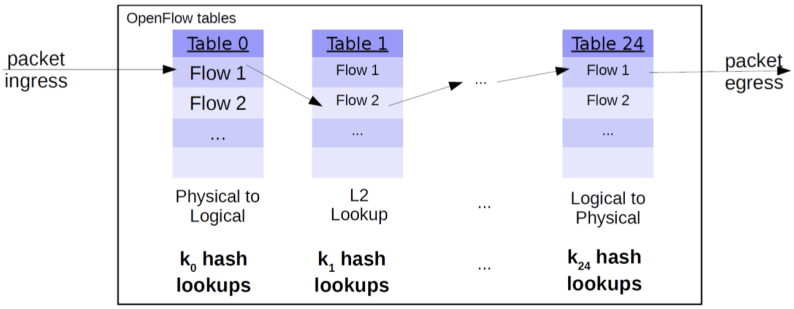
Architecture

Both kernel and user space

- Client's Computer inside part
- The part connecting with the controller (OVS)
- bridge
- Connecting with the physical devices



Pipeline



- However, there are some problems:
- The fact is that: Classification is expensive on general-purpose CPUs
- So, what if there are 100+ hash lookups per packet for tuple space search?
- We propose the "OVS Cache" Architecture !

We ignore the Architecture of OvS Cache here !

- Microflow Cache
  - Speed up the Microflow Cache
  - Naive Approach to Populating Cache
  - Lazy Approach to Populating Cache
- “Megaflow” Cache
- Dual Caches

OpenFlow Session Setup

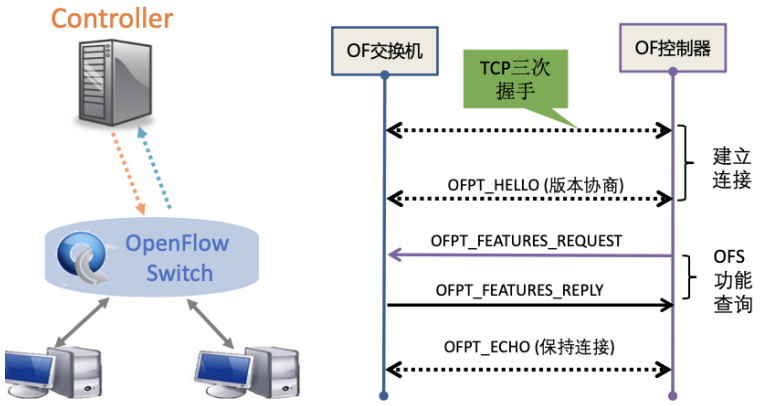
OpenFlow Messages

消息类型	消息例子	描述
Controller to Switch	- FlowMod / - Packet out / - Switch Configuration / - Switch Features	添加、修改、删除流表项 / 将数据包发送给指定的交换机端口 / 配置交换机 / 查询交换机的功能和统计

消息类型	消息例子	描述
Asynchronous (异步)	- Packet in / Flow Removed / Port Status	没有匹配交换机的任意流表项，通知控制器 / 流表项删除，通知控制器 / 端口状态改变，通知控制器
Symmetric（对称）	- Hello / - Echo / - Experimenter	控制器和交换机建立连接时使用 / 用来确定交换机与控制器的连接是否活跃 / 用来消息拓展

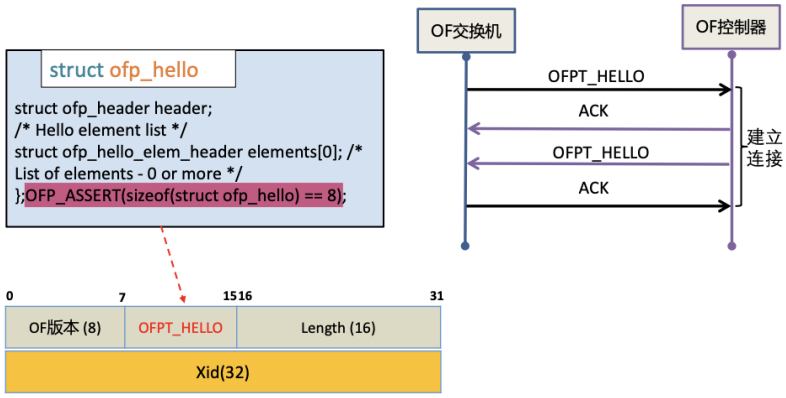
OF Switch & OF Controller

- 建立连接
  - TCP 三次握手
  - OFPT_HELLO（版本协商）
- OFS功能查询
  - OFPT_FEATURES_Request
  - OFPT_FEATURES_Reply



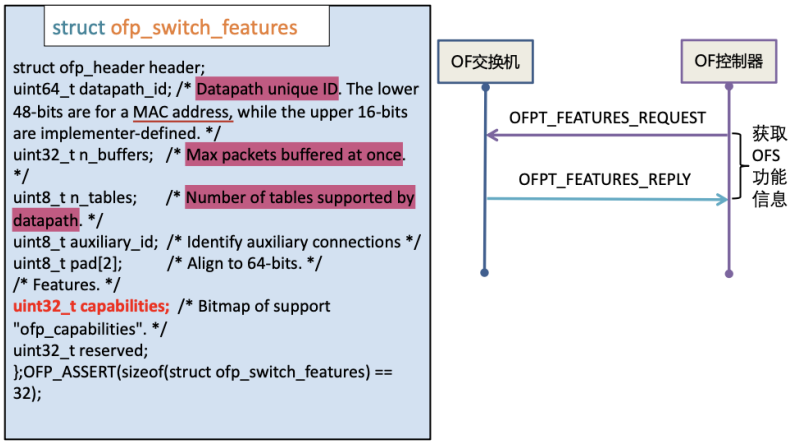
Hello Message

- OFPT_HELLO
- ACK
- OFPT_HELLO
- ACK



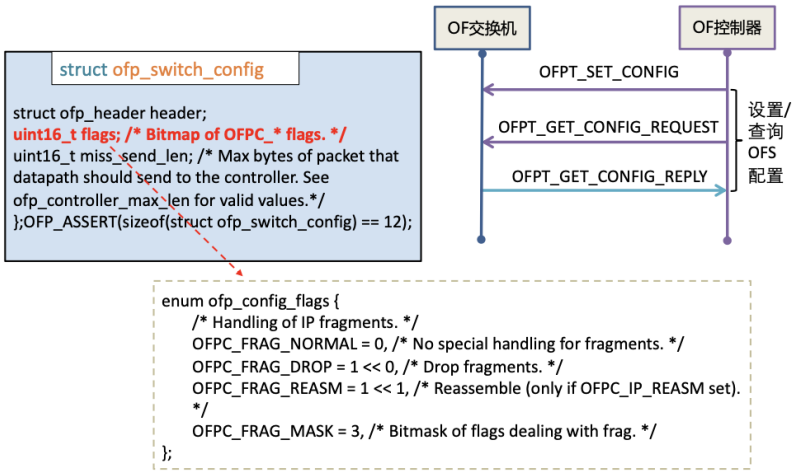
Switch Features Messages

- OFPT_FEATURES_REQUEST
- OFPT_FEATURES_REPLY



Switch Config Message

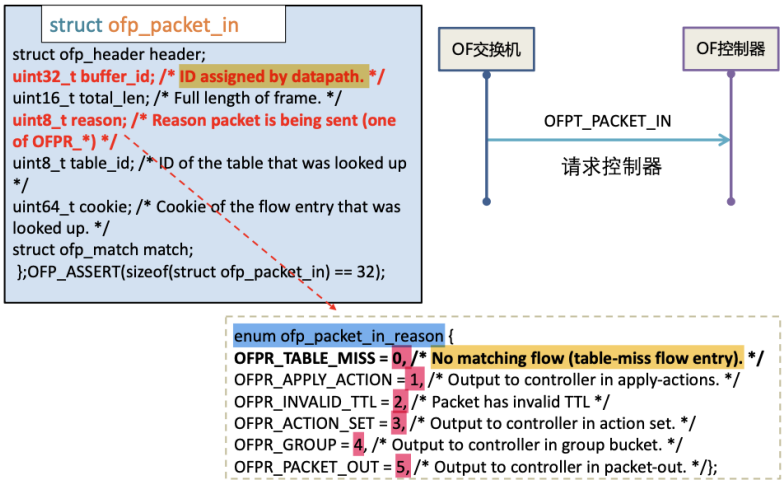
- OFPT_SET_CONFIG
- OFPT_GET_CONFIG_REQUEST
- OFPT_GET_CONFIG_REPLY



Packet-in Messages

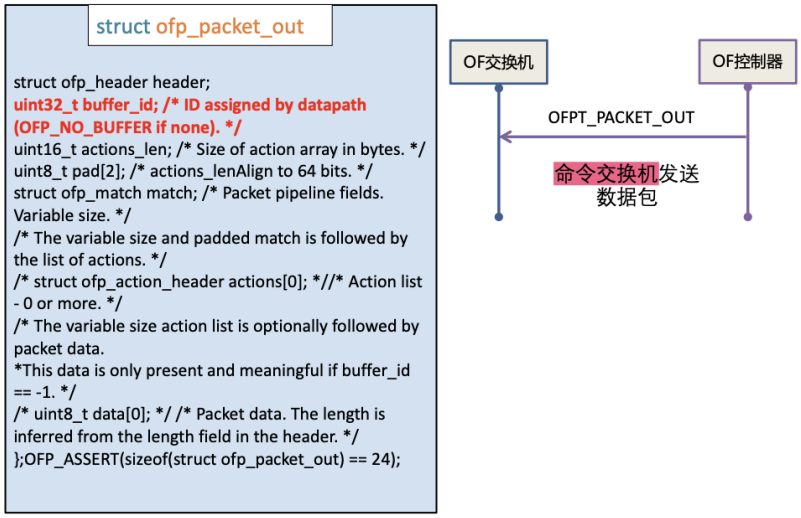
情景：A packet is coming in, and the Switch asks controller how to react !

- OFPT_PACKET_IN 请求控制器
  - buffer_id：当有一个不知道发到哪里的包时，先缓存起来



Packet-out Messages

- OFPT_PACKET_OUT 命令交换机发送数据包
  - buffer_id：buffered packet to apply to
  - action[]：装有action，以及对应端口

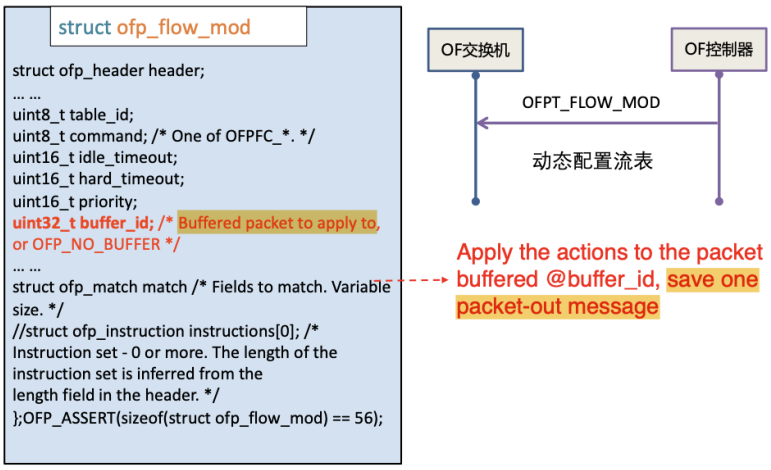


PS: 这里的 in / out 都是相对于Switch而言的!

FlowMod Messages

- OFPT_FLOW_MOD 动态配置流表

FlowMod Messages



OpenFlow eXtensible Match (OXM)

- OXM = compact TLV (type-length-value) format