4. 通用转发和SDN

4.1 匹配
数据平面
本地的、每个路由器的功能
决定从某个端口进入的分组从哪个端口输出
转发功能
控制平面
网络范围的逻辑
决定分组端到端穿行于各个路由器的路径

每个路由器（Per Route）的控制平面
每个路由器上都有实现路由单元算法元件（他们之间需要相互交互）-形成传统IP实现方式的控制平面
每台设备上既实现控制平面功能，又实现数据平面功能
控制功能分布式实现
路由表-粘连

数量众多、功能各异的中间盒
路由器的网络层功能：
IP转发：对于到来的分组按照路由表决定如何转发，数据平面
路由：　决定路径，计算路由表；处在控制平面
还有其他种类繁多的网络设备（中间盒）
交换机：防火墙；NAT;IDS;负载均衡设备
未来：　不断增加的需求和相应的网络设备
需要不同的设备去实现不同的网络功能
每台设备集成了控制平面和数据平面的功能
控制平面分布式地实现了各种控制平面功能
升级和部署网络设备非常困难

网络设备控制平面地实现方式特点：
互联网络设备：
传统方式都是通过分布式，每台设备地方法来实现数据平面和控制平面
垂直集成：每台路由器或其他网络设备，包括
1.硬件、在私有的操作系统
2.互联网协议标准（IP,RIP,IS-IS,OSPF,BGP）的私有实现
从上到下都由一个厂商提供（代价大，被设备上"绑架"）
每个设备都实现了数据平面和控制平面的事情
控制平面的功能是分布式实现的
设备基本上只能（分布式升级困难）按照固定方式工作，控制逻辑固化。不同的网络功能需要不同的"middleboxes"
防火墙
负载均衡设备
NAT boxes

(数据+控制平面的)集成>(控制逻辑)分布>固化
代价大；升级困难；管理困难等

4.2 传统方式实现网络的问题

问题：
垂直集成
昂贵、不便于创新的生态
分布式、固化设备功能===网络设备种类繁多
无法改变路由等工作逻辑，无法实现流量工程等高级特性
配置错误影响全网运行；升级和维护会设计到全网设备：管理困难
要增加新的网络功能，需要设计、实现以及部署新的特定设备，设备种类繁多
~2005：开始重新思考网络控制平面的处理方式

集中：远程的控制器集中实现控制逻辑
远程：数据平面和控制平面的分离

4.3 SDN:逻辑上集中的控制平面

一个不同的（通常是远程）控制器与CA交互，控制器决定分组转发的逻辑（可编程），CA所在设备执行逻辑

主要思路：
网络设备数据平面和控制平面分离
数据平面-分组交换机
将路由器、交换机和目前大多数网络设备的功能进一步抽象成：按照流表进行PDU的动作
流表： 由控制平面设置的控制逻辑
PDU： 帧分组
动作： 包括转发、丢弃、拷贝、泛洪、阻塞
统一化设备功能：SDN交换机（分组交换机），执行控制逻辑
控制平面-控制器+网络应用
分离集中
计算和下发控制逻辑：流表

优势：
水平集成控制平面的开放实现（而非私有实现），创造出好的产业生态，促进发育
分组交换机、控制器和各种控制逻辑网络应用的app可由不同厂商生产，专业化，引入竞争形成良好生态
集中式实现控制逻辑，网络管理容易
集中式控制器了解网络状况，编程简单，传统方式困难
避免路由器的误配置
基于流表的匹配+行动的工作方式允许"可编程的"分组交换机
实现流量工程的高级特性
在此框架下实现各种新型（未来）网络设备

垂直集成封闭，私有没创新缓慢，产业规模小
水平集成，开放接口，快速创新，产业巨大

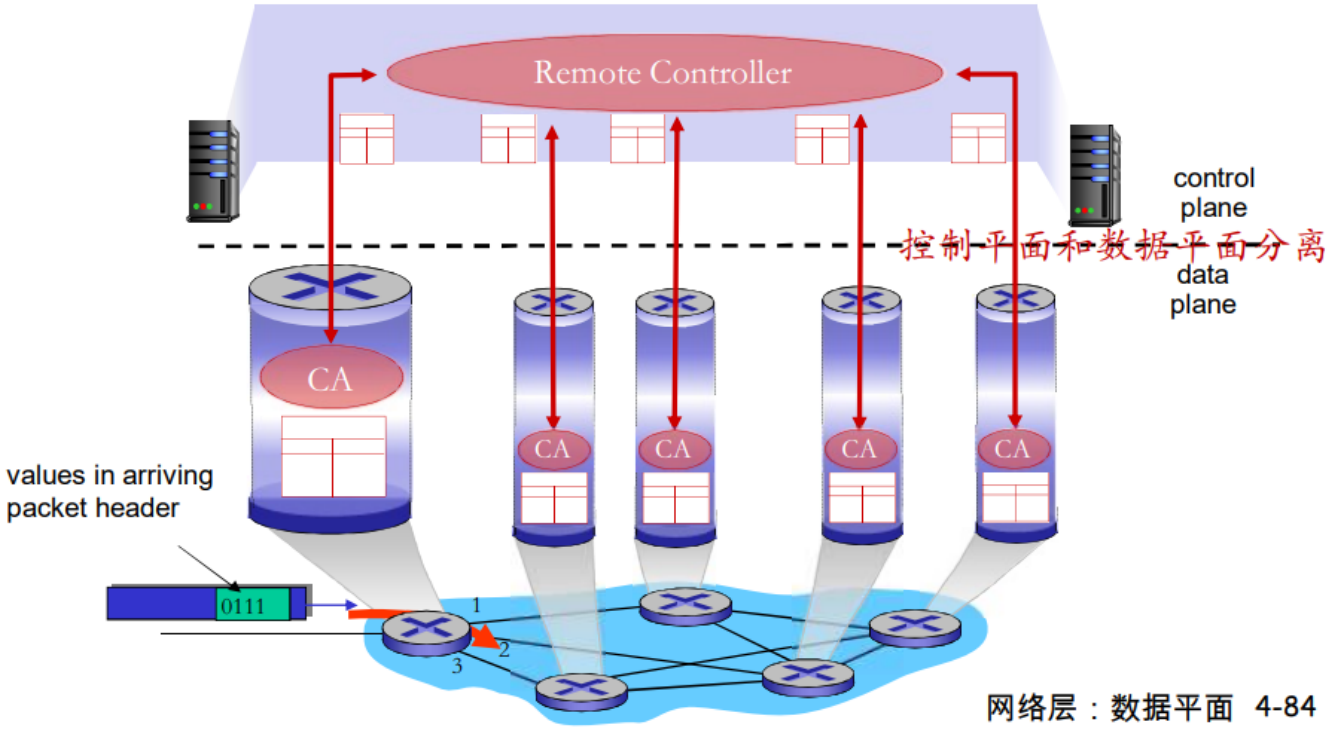
流量工程：
在SDN下可以实现：
流量分多路走
流量特定路径走
不同流量不同路径
见图

特点：
1.通用"flow-based"基于流的匹配+行动
2.控制平面和数据平面的分离
3.控制平面功能在数据交换设备之外实现
4.可编程控制应用

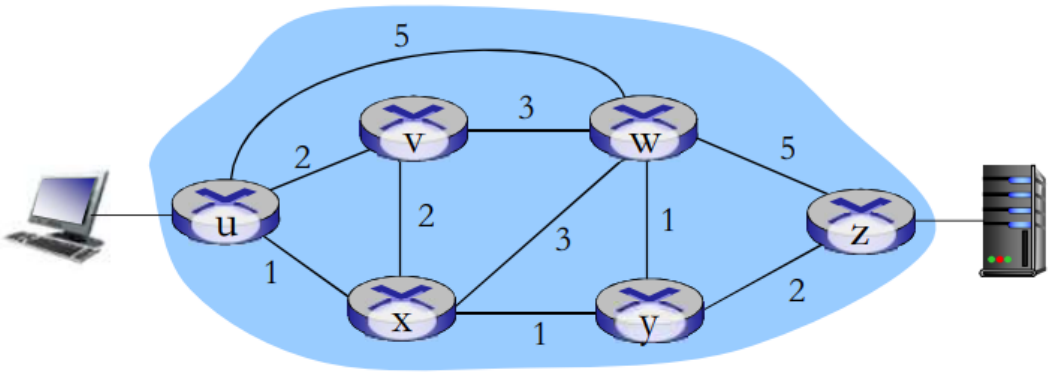
架构：
数据平面交换机
快速，简单，商业化交换设备，采用硬件实现通用转发功能
流表被控制器计算和安装
基于南向API（例如OpenFlow），SDN控制器访问基于流的交换机
定义了哪些可以被控制，哪些不能
也定义了和控制器的协议（e.g.,OpenFlow）
SDN控制器（网络OS）：
维护网络状态信息
通过上面的北向API和网络控制应用交互
通过相面的南向API和网络交换机交互
逻辑上集中，但是在实现上通常由于性能、可扩展性、容错性以及鲁棒性采用分布式方法
控制应用：
控制的大脑：采用下层提供的服务（SDN控制器提供的API）实现网络功能
路由器交换机

接入控制 防火墙
负载均衡
其他功能
非绑定：可以被第三方提供，与控制器厂商以通常上不同，与分组交换机厂商也可以不同

通用转发和SDN：
每个路由器包含一个流表（被逻辑上集中的控制器计算和分发）



流量工程：传统路由比较困难

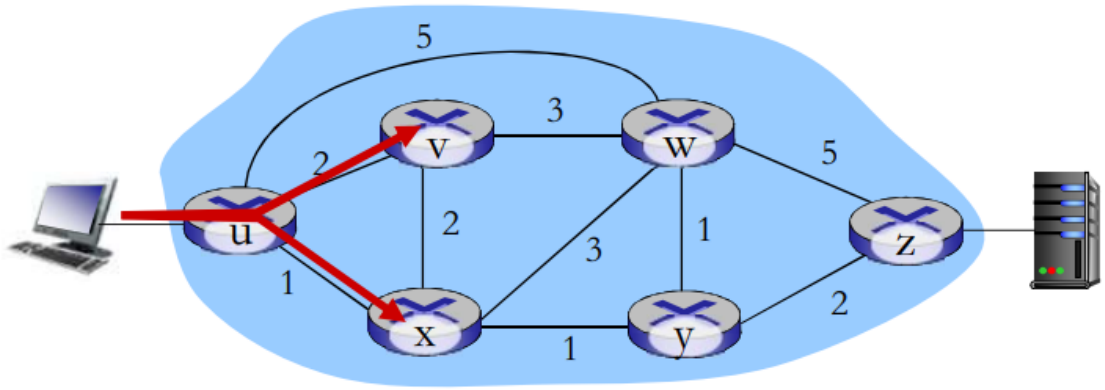


Q: 网管如果需要u到z的流量走uvwz,x到z的流量走xwyz，怎么办?

A: 需要定义链路的代价，流量路由算法以此运算（IP路由面向目标，无法操作）(或者需要新的路由算法)!

链路权重只是控制旋钮，错!

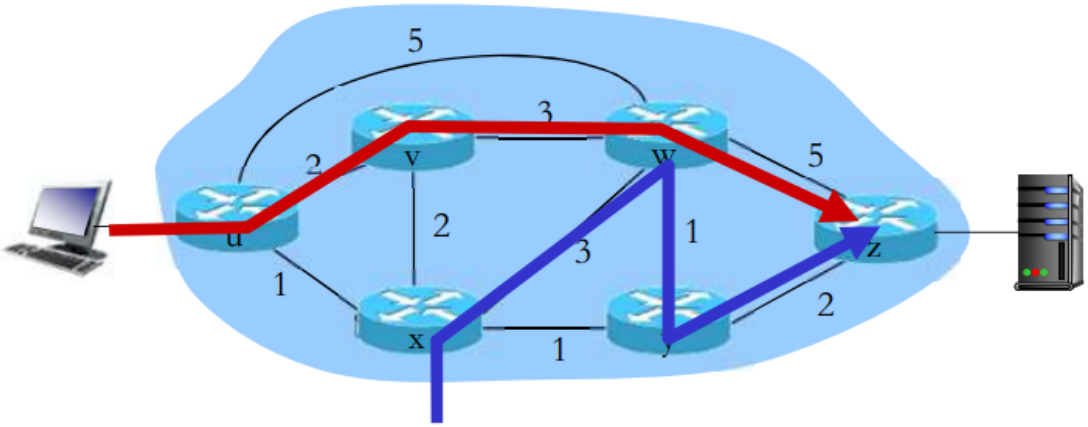
流量工程：困难



Q: 如果网管需要将u到z的流量分成2路：uvwz 和uxyz（负载均衡），怎么办？（IP路由面向目标）

A: 无法完成(在原有体系下只有使用新的路由选择算法，而在全网部署新的路由算法是个大的事情)

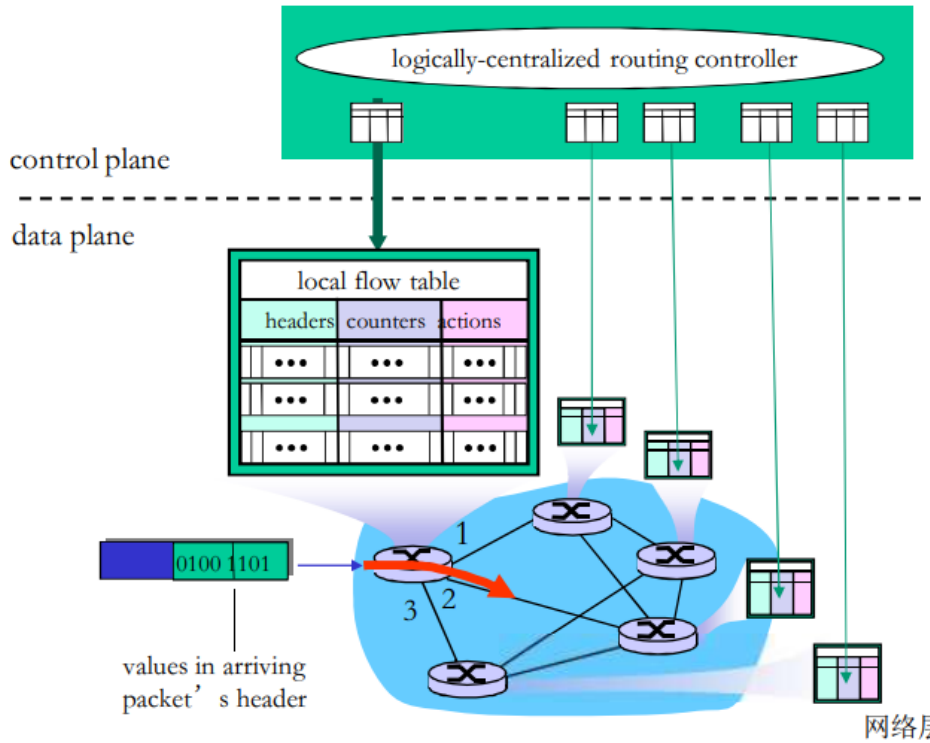
流量工程：困难



Q: 如果需要w对蓝色的和红色的流量采用不同的路由，怎么办？

A: 无法操作 (基于目标的转发，采用LS, DV 路由)

每个路由器包含一个流表（被逻辑上集中的控制器计算和分发）



4.4 OpenFlow 数据平面抽象

流：由分组（帧）头部字段所定义
通用转发：简单的分组处理机制
模式：将分组头部字段和流表进行匹配
行动：对于匹配上的分组，可以是丢弃、转发、修改
优先权Priority：几个模式匹配了，优先采用哪个，消除歧义
计数器Counter：#bytes 以及 #packets

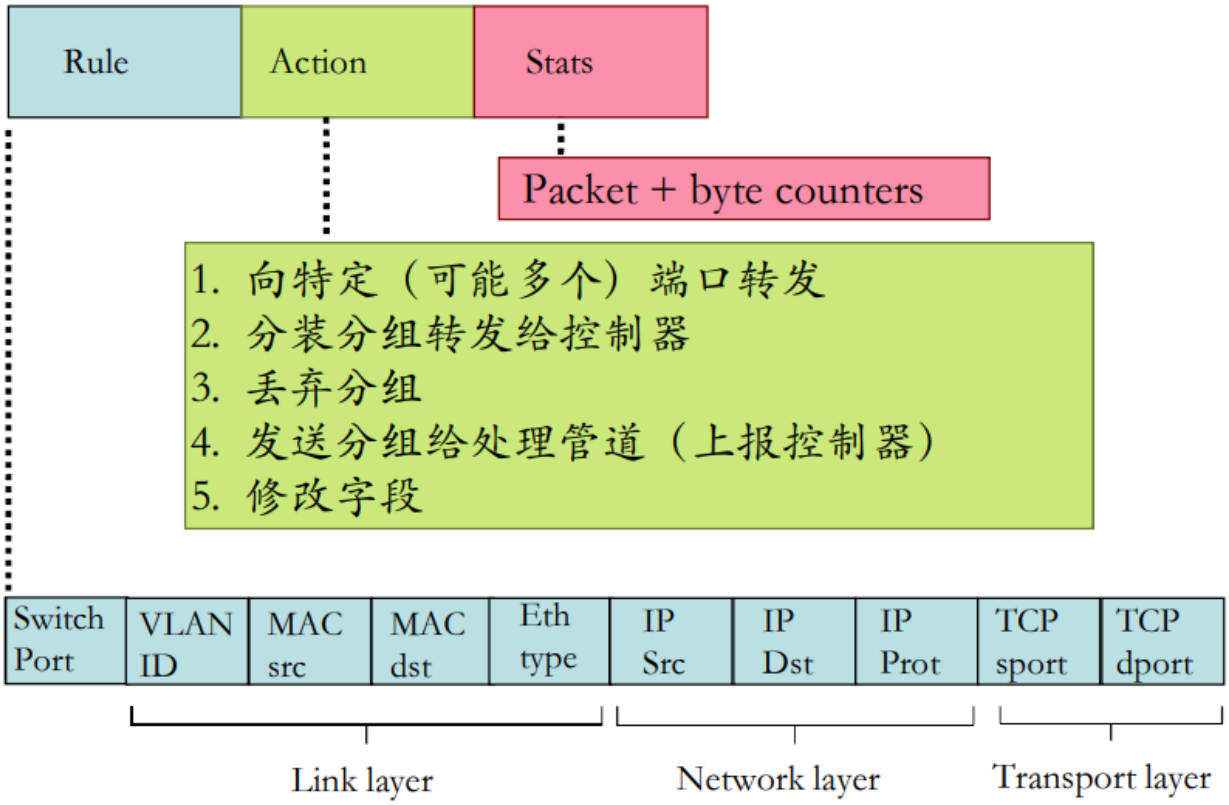
路由器中的流表定义了路由器的匹配+行动规则 （流表由控制器计算并下发）

match+action：统一化各种网络设备提供的功能

功能	match	action
路由器	最长前缀匹配	通过一条链路转发
交换机	目标MAC地址	转发或者泛洪
防火墙	IP地址，TCP/UDP ports	允许或者禁止
NAT	IP地址和端口号	重写地址和端口号

目前几乎所有的网络设备都可以在这个match+action模式框架进行描述，具体化为各种网络设备包括未来的网络设备

OpenFlow: 流表的表项结构



例子

基于目标的转发

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	51.6.0.8	*	*	*	port6

IP 数据报目标地址是51.6.0.8
应该被通过端口6转发

防火墙:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	*	*	*	*	22	drop

不转发（阻塞）所有具有目标TCP端口号 是22的分组

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	128.119.1.1	*	*	*	*	drop

所有由128.119.1.1发送的分组都应该被阻塞

基于层2目标的转发:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	22:A7:23: 11:E1:02	*	*	*	*	*	*	*	*	port3

所有层2源MAC地址是 22:A7:23:11:E1:02都应该被向端口3转发

OpenFlow 例子

例子: 来自H5和H6的数据报应该被发向H3或者H4通过 s1然后经由 s2