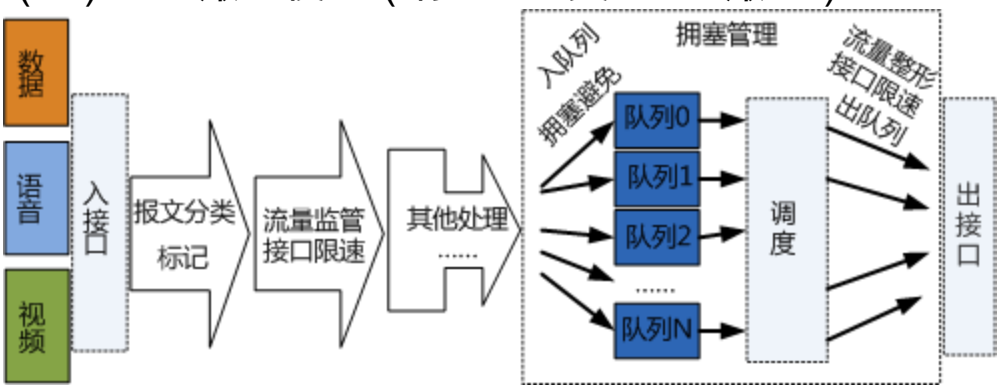
QOS 的服务模型有几种？
（1）尽力而为服务模型
（2）综合服务模型（资源预留，场景：公交车专用道，导致的问题，资源空闲时其他的业务流量也不能够使用）
（3）区分服务模型（标记--分类--差分服务）



分类/标记

分类/标记怎么做？有以下两种做法：
（1）简单流分类标记：根据各层报文头部中的优先级字段，来将外部优先级映射成为内部优先级。
（2）复杂流分类重标记；根据各层报文头部中的优先级字段，或者 SIP、DIP 等的五元组来对，流量进行分类，然后在打上相应的优先级。
（使用 MQC 实现，流分类、流行为、流策略）

扩展问题 1：DSCP（差分服务代码点）和 IPP（ip 优先级）的区别？
都为 ip 报文中的 TOS 的一部分：
IPP 是 TOS 前 3bit
DSCP 是 TOS 前 6bit---前 3bit 代表优先级，后 3bit 代表 D 延迟、T 吞吐、R 可靠性
AF（Assured-forwarding）（第 6bit 固定为 0）（后 3bit 代表丢弃

概率）
AF11 12 13 --重要数据
AF21 22 23 --视频点播流量
AF31 32 33 --直播流量
AF41 42 43 --语音的信令流量
EF --46-- 101 110（低延迟、高吞吐） --语音流量
BE --尽力而为

CS 类选择器（DSCP 用于兼容 IPP）
cs7 bfd 111
cs6 路由协议 110
cs5 语音流量 101
cs4 语音信令 100
cs3 直播流量 011
cs2 点播流量 010
cs1 001
cs0 000

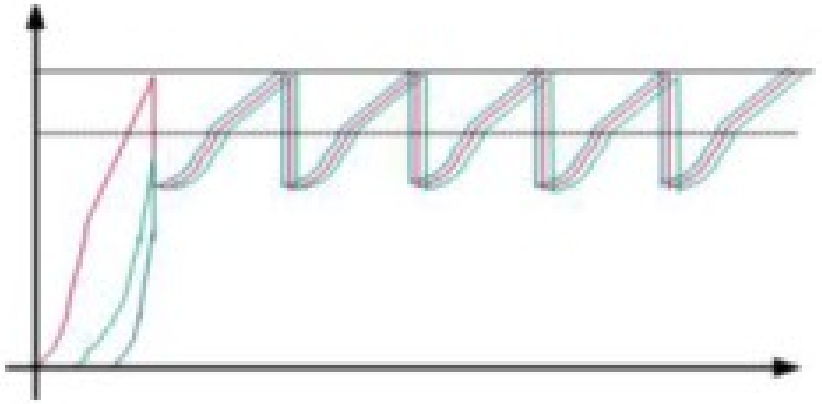
扩展问题 2：COS 和 TOS 的区别？
COS 指的是二层和 2.5 层中的优先级字段，分别为 vlan-tag 中的 PRI（802.1p）和 mpls 报头中 EXP。
TOS 指的是三层 ip 报头中的优先级字段，前 3bit 为 IPP，前 6bit 为 DSCP。

拥塞避免

（1）尾丢弃：

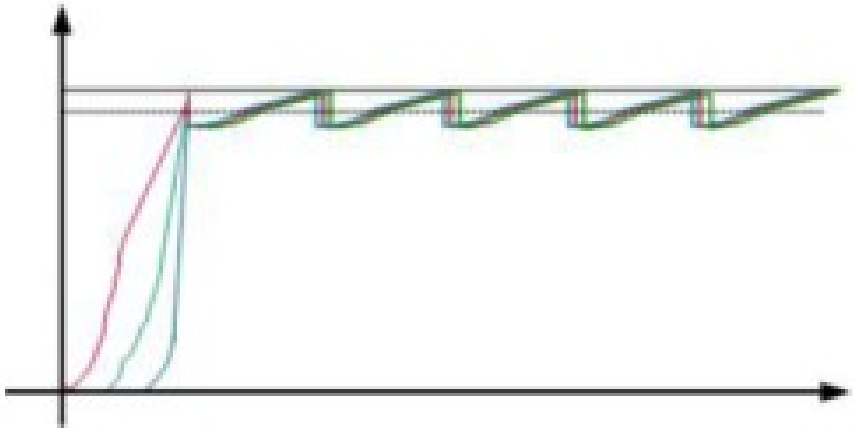
当队列的长度达到最大值后，所有新入队列的报文（缓存在队列尾部）都将被丢弃，这种丢弃策略会引发 TCP 全局同步现象，导致 TCP 连接始终无法建立。所谓 TCP 全局同步现象如图，三种颜色表示三条 TCP 连接，当同时丢弃多个 TCP 报文时，将造成多个 TCP

连接，同时触发滑窗减半机制。又会由于慢启动的机制，将流量慢慢的增大，之后又会在某个时间同时出现流量高峰，触发滑窗减半的机制。如此反复，使网络流量忽大忽小。



尾丢弃出现的问题：
1 TCP 同步（没有充分利用链路带宽）
2 TCP 饿死（ UDP 没有 TCP 那种滑动窗口）
3 无差别的丢弃

在 CBQ 中，EF 队列和 LLQ 队列不能使用丢弃策略，只能尾丢弃。为避免 TCP 全局同步现象，出现了 RED（Random Early Detection）技术。RED 通过随机地丢弃数据报文，让多个 TCP 连接不同时降低发送速度，从而避免了 TCP 的全局同步现象。使 TCP 速率及网络流量都趋于稳定。



（2）WRED：

RED 是没有差分服务的，即使优先级高的也可能被随机丢弃掉，所以基于 RED，实现了 WRED。流队列支持基于 DSCP 或 IP 优先级进行 WRED 丢弃，每一种优先级都可以独立设置报文丢包的上下门限及丢包率，报文到达下限时，开始丢包，随着门限的增高，丢包率不断增加，最高丢包率不超过设置的丢包率，直至到达高门限，报文全部丢弃，这样按照一定的丢弃概率主动丢弃队列中的报文，从而一定的程度上避免拥塞问题

WRED 针对队列，先有队列，才能配置相应的丢弃技术；
使用：1 可以在队列模板中使用　2 MQC-CBQ 中使用

扩展问题 1：尾丢弃和 WRED 的区别的什么？
尾丢弃：针对一个队列，当队列满的时候，对后续来的流量无差分的丢弃；
缺点：
1、无法提供差分服务
2、导致 TCP 全局同步
3、导致 TCP 饿死---- 记得分别画图解释

针对尾丢弃的这些缺点，就有了 WRED，WRED 可以针对不同的流量设置一个从什么时候开始丢弃（丢弃的低门限值）和最高门限，丢弃概率是多少，当队列中该流量达到最高门限值时，该报文将全部丢弃。
队列满的时候也是执行尾丢弃，从而实现差分的服务；同时举例说明如何解决尾丢弃的其他缺点；

扩展问题 2：wred 是怎么区分不同数据流的？
根据数据中的优先级字段来对数据流进行区分。

扩展问题 3：wred 中的 w 是什么意思？

W 的英文为 weight，是权值的意思，在 WRED 中，主要是用于实现针对不用的优先级的数据配置不同的丢弃上下阈值和丢弃百分比。路由器根据 IPP 或者 DSCP 判断权重值；交换机根据数据包的颜色判断权重值。

扩展问题 4：TCP 是怎么检测发生拥塞的？
没有收到相关的 TCP ACK 确认。

扩展问题 5：TCP 全局同步是什么原因导致的？
当多个 TCP 连接在发送数据时，TCP 数据发送过程中有慢启动机制，因此发送的 TCP 流量会逐渐增大，当网络发生拥塞时，TCP 又会有滑窗减半的机制，这样 TCP 的流量又会减半下来，如此重复，就形成了 TCP 全局同步的问题。

扩展问题 6：TCP 慢收敛机制？
其实就是 TCP 发送数据包的数量逐渐增多的过程。

队列技术

（1）FIFO：先进先出队列，是单队列技术，不会引入额外延迟，延迟只与队列长度有关，不提供任何差分服务。

（2）RR：轮询调度，采用轮询的方式，对多个队列进行调度 RR 以环形的方式轮询多个队列。如果轮询的队列不为空，则从该队列取走一个报文；如果该队列为空，则直接跳过该队列，调度器并不等待。单队列里还是先进先出。

（3）WRR：加权轮询调度，在队列之间进行轮流调度，根据每个队列的权重来调度各队列中的报文流。在进行 WRR 调度时，设备根据每个队列的权值进行轮循调度。调度一轮权值减一，权值减到

零的队列不参加调度，当所有队列的权限减到 0 时，开始下一轮的调度。从统计上看，各队列中的报文流被调度的次数与该队列的权值成正比，权值越大被调度的次数相对越多。由于 WRR 调度的以报文为单位，因此每个队
列没有固定的带宽，同等调度机会下大尺寸报文获得的实际带宽要大于小尺寸报文获得的带宽。

（4）DRR：差额轮询调度，类似于 CQ。解决了 WRR 只关心报文，同等调度机会下大尺寸报文获得的实际带宽要大于小尺寸报文获得的带宽的问题，通过调度过程中考虑了包长的因素，从而达到调度的速率公平性。DRR 调度中，Deficit 表示队列的带宽赤字，初始值为 0。每次调度前，系统按权重为各队列分配带宽，计算 Deficit 值，如果队列的 Deficit 值大于 0，则参与此轮调度，发送一个报文，并根据所发送报文的长度计算调度后 Deficit 值，作为下一轮调度的依据；如果队列的 Deficit 值小于 0，则不参与此轮调度，当前 Deficit 值作为下一轮调度的依据。

（5）PQ：PQ 调度算法维护一个优先级递减的队列系列并且只有当更高优先级的所有队列为空时才服务低优先级的队列，PQ 调度算法对低时延业务非常有用，然而 PQ 调度机制会使低优先级队列中的报文由于得不到服务而“饿死”。

（6）FQ：公平队列，目的是尽可能公平地分享网络资源，使所有流的延迟和抖动达到最优。不同的队列获得公平的调度机会，从总体上均衡各个流的延迟。短报文和长报文获得公平的调度：如果不同队列间同时存在多个长报文和短报文等待发送，让短报文优先获得调度，从而在总体上减少各个流的报文间的抖动。

（7）WFQ：与 FQ 相比，WFQ（Weighted Fair Queue）在计算报文调度次序时增加了优先权方面的考虑。从统计上，WFQ 使高

优先权的报文获得优先调度的机会多于低优先权的报文，短报文的调度机会多于长报文的调度机会。WFQ 调度在报文入队列之前，先对流量进行分类，有两种分类方式：

1 按流的“会话”信息分类：

根据报文的协议类型、源和目的 TCP 或 UDP 端口号、源和目的 IP 地址、ToS 域中的优先级位等自动进行流分类，并且尽可能多地提供队列，以将每个流均匀地放入不同队列中，从而在总体上均衡各个流的延迟。在出队的时候，WFQ 按流的优先级（precedence）来分配每个流应占有带宽。优先级的数值越小，所得的带宽越少。优先级的数值越大，所得的带宽越多。这种方式只有 CBQ 的 default-class 支持。

2 按优先级分类：

通过优先级映射把流量标记为本地优先级，每个本地优先级对应一个队列号。每个接口预分配 4 个或 8 个队列，报文根据队列号进入队列。默认情况，队列的 WFQ 权重相同，流量平均分配接口带宽。用户可以通过配置修改权重，高优先权和低优先权按权重比例分配带宽。

（8）CBQ（EF（包含 LLQ）、AF、BE）

1 EF 队列：满足低时延业务

EF 队列是具有高优先级的队列，一个或多个类的报文可以被设定进入 EF 队列，不同类别的报文可设定占用不同的带宽。在调度出队的时候，若 EF 队列中有报文，会优先得到调度，以保证其获得低时延。当接口发生拥塞时，EF 队列的报文会优先发送，但为了防止低优先级队列（AF、BE 队列）得不到调度，EF 队列以设置的带宽限速。当接口不拥塞时，EF 队列可以占用 AF、BE 的空闲带宽。这样，属于 EF 队列的报文既可以获得空闲的带宽，又不会占用超出规定的带宽，保护了其他报文的应得带宽。

设备除了提供普通的 EF 队列，还支持一种特殊的 EF 队列——LLQ 队列。两种队列都采用绝对优先调度，但是 LLQ 队列使用流量监

管实现，不论接口是否拥塞，流量都不会超过设置的带宽，LLQ 队列不缓存报文，可以将报文被发送的时延降低为最低限度。这为对时延敏感的应用（如 VoIP 业务）提供了良好的服务质量保证

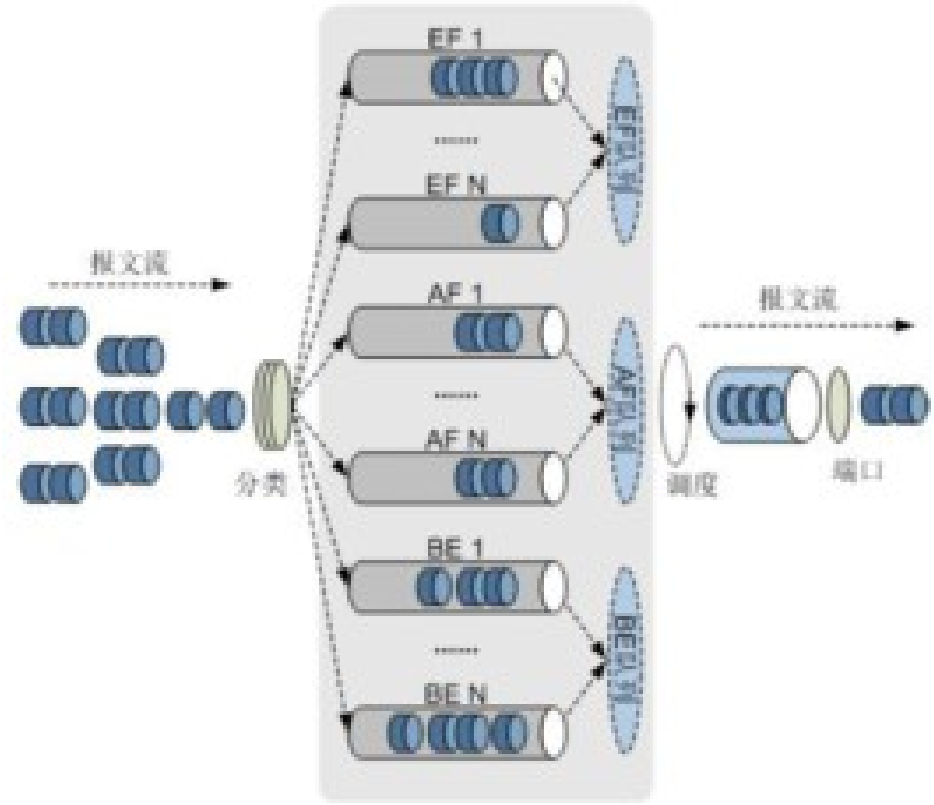
2 AF 队列：满足需要带宽保证的关键数据业务

每个 AF 队列分别对应一类报文，用户可以设定每类报文占用的带宽。在系统调度报文出队的时候，按用户为各类报文设定的带宽将报文出队发送，可以实现各个类的队列的公平调度。当接口有剩余带宽时，AF 队列按照权重分享剩余带宽。同时，在接口拥塞的时候，仍然能保证各类报文得到用户设定的最小带宽

对于 AF 队列，当队列的长度达到队列的最大长度时，缺省采用尾丢弃的策略，但用户还可以选择用 WRED 丢弃策略

3 BE 队列：满足不需要严格 QoS 保证的尽力发送业务

当报文不匹配用户设定的所有类别时，报文被送入系统定义的缺省类。虽然允许为缺省类配置 AF 队列，并配置带宽，但是更多的情况是为缺省类配置 BE 队列。BE 队列使用 WFQ 调度，使所有进入缺省类的报文进行基于流的队列调度。对于 BE 队列，当队列的长度达到队列的最大长度时，缺省采用尾丢弃的策略，但用户还可以选择用 WRED 丢弃策略。



扩展问题 1：拥塞管理和拥塞避免那个优先使用？
控制层面上：要先有队列调度技术，才能针对不同的队列使用 WRED，即使用拥塞避免技术；
数据层面上：从进入队列的数据的角度来讲，不被 WRED 丢弃的数据才有资格被队列调度技术调度；
即先进行拥塞避免技术，将能够转发的数据留着队列中，再由队列调度技术来进行调度；

扩展问题 2：MQC 全称是什么，怎么使用，有什么内容？
模块化 QOS 命令行；
在要求针对不同的业务流量进行整形或者配置 WRED 的时候使用；
MQC 的三要素包括：流分类、流行为、流策略，最后应用流策略；

扩展问题 3：FIFO 机制是不是不区分流量的优先级？
答：是的；

扩展问题 4：那为什么还需要 FIFO 这个机制呢？
答：1、在网络没有发生拥塞的时候，使用的就是 FIFO 的方式转发数据的；
2、在不使用任何 Qos 服务模型的时候，也就是使用尽力而为模型的时候，也就是使用 FIFO 的方式的；
3、网络发生拥塞时，单个队列中使用的就是 FIFO 的方式发送数据的；

令牌桶技术

扩展问题 1：双桶双速与双桶单速的差异有哪些？
（1）双桶双速是有两个速度的（cir 和 pir），系统使用 cir（承诺信息速率）朝着 cbs（承诺突发尺寸）注入令牌；使用 pir（峰值信息速率）朝 pbs（峰值突发尺寸），当有数据经过这两个桶时，先检查 pbs 再检查 cbs：
1 如果两个都够，从两个都桶取出相应的令牌，然后标记成 green.
2 如果 pbs 够，cbs 不过则从 pbs 取走相应的令牌并且标记成 yellow.
3 如果两个都不够就 marker 成 red.

（2）双桶单速只有一个速率（cir），cir 朝着 cbs 注入令牌，单 cbs 注满后，就会朝着 ebs（超额突发尺寸）注入令牌，当有报文来的时候，先检查 cbs，再检查 ebs：
1 如果 cbs 够，则报文被标记为绿色，且从 cbs 令牌桶取走相应的令牌；
2 如果报文大于 cbs，小于 ebs，则报文被标记为黄色，且 ebs 令牌桶取走相应的令牌；
3 如果报文大于 ebs，报文被标记为红色，不从 ebs 和 cbs 令牌桶取走相应的令牌

扩展问题 2：有单桶单速，双桶双速，双桶单速中，为什么要设置这三个机制，
一个单桶单速不就可以解决吗？为什么还要有另外两个？
因为在不同的场景，不同的客户需求下可以使用不同的技术：
1、单通单速：只有一个速率，不允许有突发的速率。
场景：适用于客户就只是买了一个 20M 之类的，固定速率网络使用；
2、单速双桶：只有一种速率但是允许超额的情况发生。
场景：适用于客户买了一个固定的网速，但是又要求可以有偶尔的超额的需求下使用；
3、双通双速：承诺一个最高速率和一个最低速率。
场景：适用于客户要求有两个上网速率，一个为保底速率和一个峰值速率的网络使用；

扩展问题 3：如果在双桶单速中 EBS 设置的过大，会出现什么问题？
答：E 通太大，会导致超额突发速率太高；（ISP 也不会希望，客户买的明明是 20M 的网络，偶尔却可以超额到 100 多兆的情况）

流量整形和流量监管

扩展问题 4：流量整形和监管有什么不同？
答：流量整形和流量监管主要都是用于 qos 中的限速区别：
（1）对超出速率的数据的处理：流量整形对超过限速的报文进行缓存，等到接口的带宽足够时，通过相应的队列技术进行调度并且转发出去；而流量监管对于超过限速的报文进行丢弃；
（2）接口上使用的方向，流量整形只能在出方向，监管可以在出方向和入方向上配置；

注意：
路由器使用 car 进行监管时， 只能使用双桶单速和双桶双速；

使用 gts 进行整形时， 只能使用单桶单速；
交换机使用流策略的 car 进行监管， 只能使用双桶单速和双桶双速；
使用 lr 进行监管时， 只能使用单桶单速；
使用 lr 进行整形（qos lr outbound）， 只能使用单桶单速；
使用 qos queue x shaping 进行整形， 只能使用双桶双速；

华为流量整形有哪几种方法？
（1）LR（line-rate 接口限速）——针对所有流量
1 只能基于接口做整形，也就是出接口的所有流量；
2 可以用在路由器接口使用，只是用于限速，配置接口速率百分比，必须结合队列使用才能生效；
3 也可以在在交换机接口使用，接口出方向做整形<qos lr outboun d>，在接口入方向做监管<qos lr inbound>；

（2）GTS（通用流量整形）——针对 IP 流量，只能用于出方向
1 实现方式有以下三种：
基于接口（针对接口下的所有三层流量进行整形）；
基于队列（使用队列模板对针对接口下的某一个队列中的三层流量进行整形）；
基于类（使用 MQC 的配置模式，针对不同的业务流量进行整形）；
2 只能用在路由器接口（交换机做不了 GTS，交换机的接口为二层接口，而二层接口无法配置 GTS，因为 GTS 是针对三层流量）
3 如果接口出现拥塞，可以使用队列技术调度缓存队列；

（3）FRTS（帧中继流量整形）——针对 FR 的接口使用
1 跟 GTS 的算法是一样的
2 用于帧中继接口
3 可以针对接口或者针对 pvc（如果两者同时启用，cir 小的配置生效）
前提：

a）简单流分类重标记针对数据包中的优先级进行一个分类后重标记，使得外优先级映射到本地优先级中，再根据不同的优先级自动的进入到相应的队列中；；
b）复杂流分类重标记：根据报文的优先级、五元组来对数据流量进行分类后重标记，再根据不同的优先级自动的进入到相应的队列中；

扩展问题 1：针对每个队列进行整形说的是接口队列的 8 个队列吗？
不是，是针对接口的部分队列中的流量进行整形，因为有的队列是不适用于整形的，比如说优先级为 EF 的语音流量，要求能够低延迟的转发，而整形虽然可以减少丢包但是会带来额外的延迟；

扩展问题 2：lr 使用在路由器和交换机上有什么区别？具体怎么使用？
1、在 router 上使用的时候只能在接口出方向上使用，用于流量整形。
在 switch 上使用的时候，出方向上是整形，入方向是监管。
2、路由器上使用的时候，是配置限速百分比，
switch 上的话是针对接口的所有流量进行限速，配置一个要限制的数值

扩展问题 3：linux 也可以使用一些软件实现 qos 那这个是不是基于软件呢？
Linux 是软件队列，但是他是基于电脑的 CPU 来执行，性能会比较差。

扩展问题 4：linux 也可以做路由器，那我们为什么还需要专业的路由器？
因为 Linux 如果要做路由器的话，需要使用相应的软件来实现，在 Linux 上使用软件来实现路由功能时，需要使用到 CPU 资源，这样

一方面消耗 CPU，一方面转发效率低。
而专业的路由器的话，是可以将控制层面的路由表项下放到数据层面上，也就是转发数据的时候由接口的硬件芯片来实现快速转发，数据转发的效率高；

扩展问题 5：缓存和队列有什么区别？
（1）数据进入队列的方式不同：
当网络发生拥塞的时候，数据先进入到队列中，然后在进过队列调度；
如果在调度过程中，令牌不够的时候，就会将超出接口速率的那一部分流量放入到缓存队列中；
（2）队列中数据的发送条件：
队列中的数据是通过队列调度技术调度出去；
而缓存队列中的数据是当令牌桶中令牌足够缓存队列中数据使用的时候，才调度出去；

扩展问题 6：延时是什么意思？延时就是抖动吗？延时和抖动有什么区别？
时延：是数据包由发送端到接收端的总时间；
抖动：数据包由发送端到接收端的时延差；

扩展问题 5：交换机和路由器的流量整形有什么不同？
（1）使用的技术不同：SW 用的是 LR，而 router 可以用 LR 也可以用 GTS；
（2）针对的流量不同：SW 的整形针对的所有的流量，而 router 的整形即可以针对所有流量也可以只针对 ip 流量；

扩展问题 6：流量整形能不能用于入方向，为什么？
不能，因为只有在接口出方向上才有缓存队列，在入方向上没有；

IPV6 中 QOS 和 IPV4 中 QOS 的区别？
ipv6 报头中多了 20bit 的流标签的字段，用于 Ipv6 的 Qos 区别不同的数据流；Flow Label：流标签，长度为 20bit。IPv6 中的新增字段，用于区分实时流量，不同的流标签+源地址可以唯一确定一条数据流，中间网络设备可以根据这些信息更加高效率的区分数据流。

三种服务模型，四大组件

QoS 服务模型：

尽力而为的服务模型（Best Effort Services Model）
综合服务模型（Integrated Services Model）
区分服务模型（Differentiated Services Model）

	优点	缺点
尽力而为服务模型	实现机制简单	对不同业务流不能进行区分对待
综合服务模型	可提供端到端QoS服务，并保证带宽、延迟	需要跟踪和记录每个数据流的状态，实现较复杂，且扩展性较差，带宽利用率较低
区分服务模型	不需跟踪每个数据流状态，资源占用少，扩展性较强；且能实现对不同业务流提供不同的服务质量	需要在端到端每个节点都进行手工部署，对人员能力要求较高

服务模型

服务模型，是指一组实现端到端 QoS 保证的方式，包括 Best Effor

t、IntServ 和 DiffServ 三种服务模型。

Best Effort 模型 尽力而为的服务模型

Best Effort 模型（即尽力而为模型）是一个单一的服务模型，也是最简单的服务模型。应用程序可以在任何时候，发出任意数量的报文，而且不需要事先获得批准，也不需要通知网络。Best Effort 模型中，网络尽最大的可能性来发送报文，但对时延、可靠性等性能不提供任何保证。

Best Effort 模型是 Internet 的缺省服务模型，它适用于绝大多数网络应用，如 FTP、E-Mail 等，它通过先入先出（FIFO）调度方式来实现。

IntServ 模型 综合服务模型

IntServ（Integrated Service）模型是一个综合服务模型，它的特点是在发送报文前要先向网络提出申请。这个请求是通过信令来完成的，一个实例是资源预留协议 RSVP（Resource Reservation Protocol）。应用程序首先通过 RSVP 信令通知网络它的 QoS 需求（如时延、带宽、丢包率等指标）。在收到资源预留请求后，传送路径上的网络节点实施许可控制（Admission control），验证用户的合法性并检查资源的可用性，决定是否为应用程序预留资源。一旦认可并为应用程序的报文分配了资源，则只要应用程序的报文控制在流量参数描述的范围内，网络节点将承诺满足应用程序的 QoS 需求。预留路径上的网络节点可以通过执行报文的分类、流量监管、低延迟的排队调度等行为，来满足对应用程序的承诺。IntServ 模型常与组播应用结合，适用于需要保证带宽、低延迟的实时多媒体应用，如电视会议、视频点播等。

当前，采用 RSVP 协议的 IntServ 模型定义了两种业务类型：

保证型服务（Guaranteed Service）

提供保证的带宽和时延限制来满足应用程序的要求。如 VoIP（Voi

ce over IP）应用可以预留 10M 带宽和要求不超过 1 秒的时延。

负载控制型服务（Controlled-Load Service）

保证即使在网络过载（overload）的情况下，仍能对报文提供类似 Best Effort 模型在未过载时的服务质量——即在网络拥塞的情况下，保证某些应用程序报文的低时延和低丢包率需求。

可以提供端到端的 QoS 投递服务是 IntServ 模型的最大优点。IntServ 模型的最大缺点是可扩展性不好。网络节点需要为每个资源预留维护一些必要的软状态（Soft State）信息；在与组播应用相结合时，还要定期地向网络发资源请求和路径刷新信息，以支持组播成员的动态加入和退出。

上述操作要耗费网络节点较多的处理时间和内存资源。在网络规模扩大时，维护的开销会大幅度增加，对网络节点特别是核心节点线速处理报文的性能造成严重影响。因此，IntServ 模型不适宜于在流量汇集的骨干网上大量应用。

DiffServ 模型 区分服务模型

为了在 Internet 上针对不同的业务提供有差别的服务质量，IETF 定义了 DiffServ（Differentiated Service）模型。

DiffServ 模型是一种多服务模型，它可以满足不同的 QoS 需求。与 IntServ 模型不同，应用程序在发出报文前，通过设置报文头部的优先级字段，向网络中各设备通告自己的 QoS 需求，而不需要通知途经的网络设备为其预留资源。

DiffServ 模型中，网络不需要为每个流维护状态，它根据每个报文携带的优先级来提供特定的服务。可以用不同的方法来指定报文的 QoS，如 IP 报文的优先级（IP Precedence），报文的源地址和目的地址等。网络通过这些信息来进行报文的分类、流量整形、流量监管和队列调度。

DiffServ 模型一般用来为一些重要的应用提供端到端的 QoS。通常在配置 DiffServ 模型后，边界设备通过报文的源地址和目的地址等

信息对报文进行分类，对不同的报文设置不同的优先级，并标记在报文头部。而其他设备只需要根据设置的优先级来进行报文的调度。

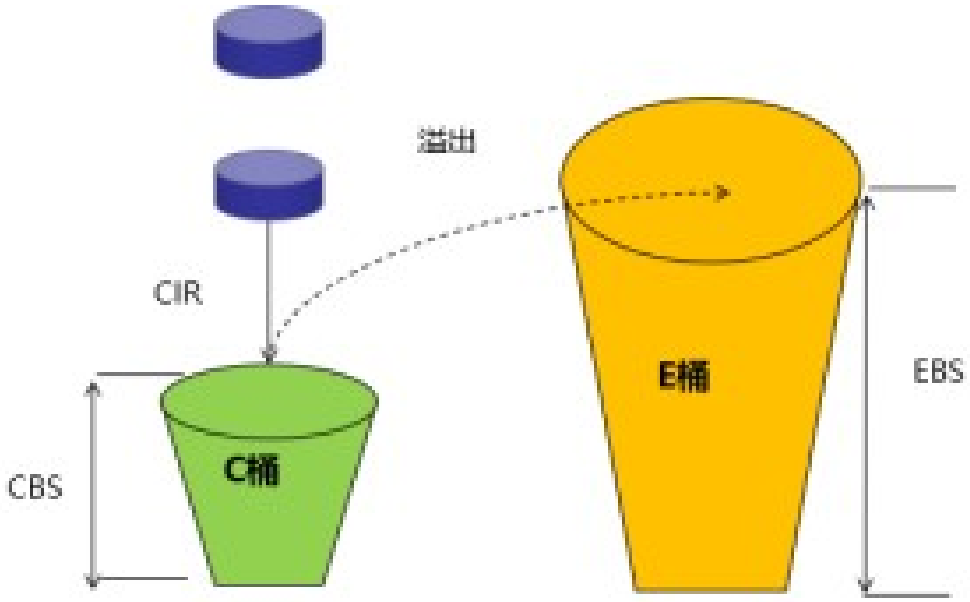
QoS 的度量指标：带宽、时延和抖动、以及丢包率

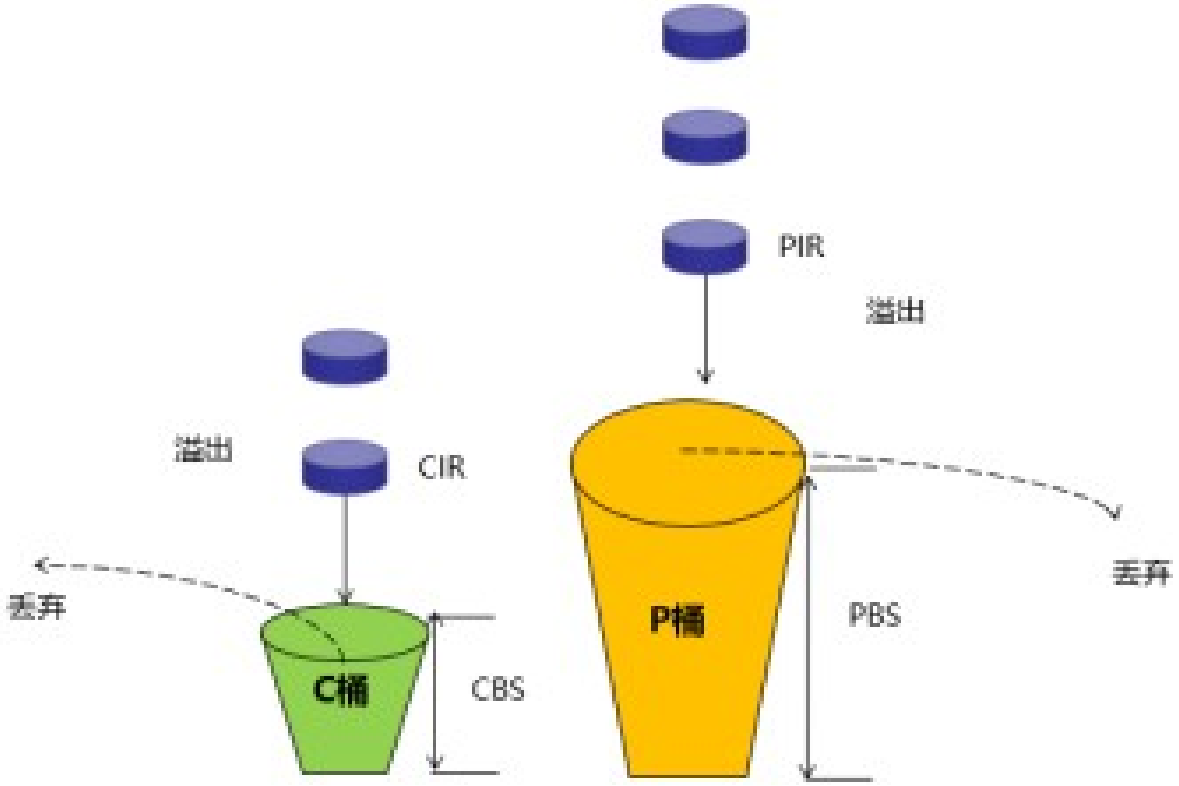
DiffServ 模型包含了四大组件，通过这四大组件实现端到端的 QoS：

1. 优先级映射（报文分类和标记）
要实现差分服务，首先需要将报文分为不同的类别。类别确定好了，设备才能针对性地提供服务。

2. 流量管理（流量监管、流量整形和接口限速）
令牌桶：单速单桶，单速双桶，双速双桶

限速：通过配置接口发送报文速率占接口带宽的百分比，实现对接口发送报文速率的限制
限制用的是单速单桶





流量监管：单速双桶或双速双桶，三色
是将流量限制在特定的带宽内。当业务流量超过额定带宽时，超过的流量将被丢弃。这样可以防止个别业务或用户无限制地占用带宽。
流量整形：单速单桶，双色
是一种主动调整流的输出速率的流控措施，使流量比较平稳地传送给下游设备，避免不必要的报文丢弃和拥塞。流量整形通常在接口出方向使用。
接口限速：单速单桶，双色
是对一个接口上发送或者接收全部报文的总速率进行限制。当不需要对报文类型进行进一步细化分类而要限制通过接口全部流量的速率时，接口限速功能可以简化配置。

流量监管与流量整形的区别
在进行报文流量控制时，流量监管是对超过流量限制的报文进行丢弃；而流量整形则将超过流量限制的报文缓存在队列中，等待链路空闲的时候再发送。

3. 拥塞管理
拥塞管理是在网络发生拥塞时，通过一定的调度算法安排报文的转发次序，保证网络可以尽快恢复正常。拥塞管理通常在接口出方向使用。

4. 拥塞避免
拥塞避免可以监视网络资源（如队列或内存缓冲区）的使用情况。在拥塞有加剧的趋势时，主动丢弃报文，避免网络拥塞继续加剧。拥塞管理通常在接口出方向使用。

综上所述，报文分类是基础，是有区别地实施服务的前提，流量监管、流量整形和接口限速主要用于预防拥塞，拥塞管理和拥塞避免是用来解决拥塞。