RDMA over Commodity Ethernet at Scale(Abstruct)

Abstruct：

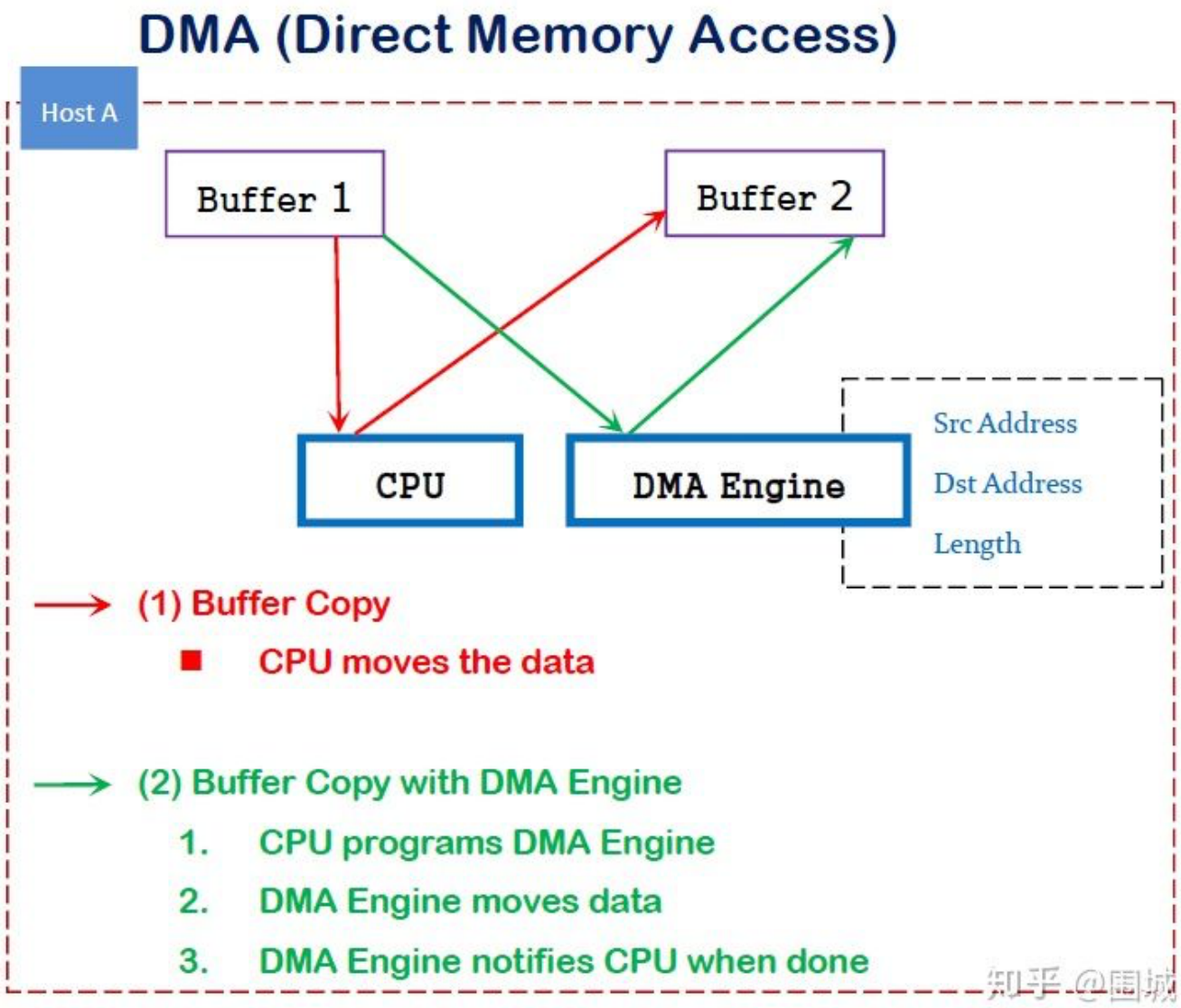
> Over the past one and half years, we have been using RDMA over commodity Ethernet (RoCEv2) to support some of Microsoft's highly-reliable, latency-sensitive services. This paper describes the challenges we encountered during the process and the solutions we devised to address them. In order to scale RoCEv2 beyond VLAN, we have designed a DSCP-based priority flow-control (PFC) mechanism to ensure large-scale deployment. We have addressed the safety challenges brought by PFCinduced deadlock (yes, it happened!), RDMA transport livelock, and the NIC PFC pause frame storm problem. We have also built the monitoring and management systems to make sure RDMA works as expected. Our experiences show that the safety and scalability issues of running RoCEv2 at scale can all be addressed, and RDMA can replace TCP for intra data center communications and achieve low latency, low CPU overhead, and high throughput.

> 在过去的一年半中，我们一直在使用基于商用以太网的 RDMA (RoCEv2) 来支持 Microsoft 的一些高度可靠、延迟敏感的服务。本文描述了我们在此过程中遇到的挑战以及我们为解决这些问题而设计的解决方案。为了将RoCEv2扩展到VLAN之外，我们设计了基于DSCP的优先级流量控制（PFC）机制以确保大规模部署。我们解决了 PFC 引起的死锁（是的，它发生了！）、RDMA 传输活锁和 NIC PFC 暂停帧风暴问题带来的安全挑战。我们还构建了监控和管理系统，以确保 RDMA 按预期工作。我们的经验表明，大规模运行RoCEv2的安全性和可扩展性问题都可以得到解决，并且RDMA可以替代TCP进行数据中心内部通信，并实现低延迟、低CPU开销和高吞吐量。

Abstruct reading：

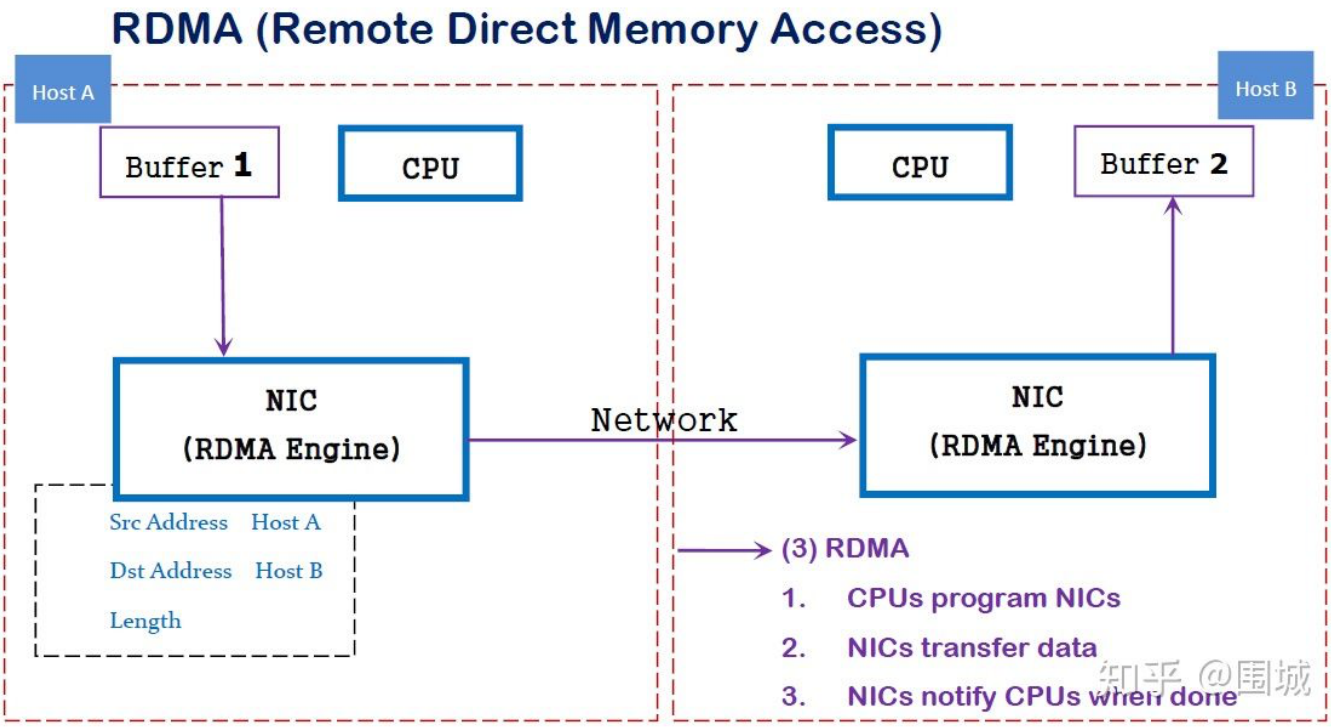
RDMA：

1）DMA概念：
DMA(直接内存访问)是一种能力，允许在计算机主板上的设备直接把数据发送到内存中去，数据搬运不需要CPU的参与。
传统内存访问需要通过CPU进行数据copy来移动数据，通过CPU将内存中的Buffer1移动到Buffer2中。DMA模式：可以同DMA Engine之间通过硬件将数据从Buffer1移动到Buffer2,而不需要操作系统CPU的参与，大大降低了CPU Copy的开销。



2）RDMA概念：

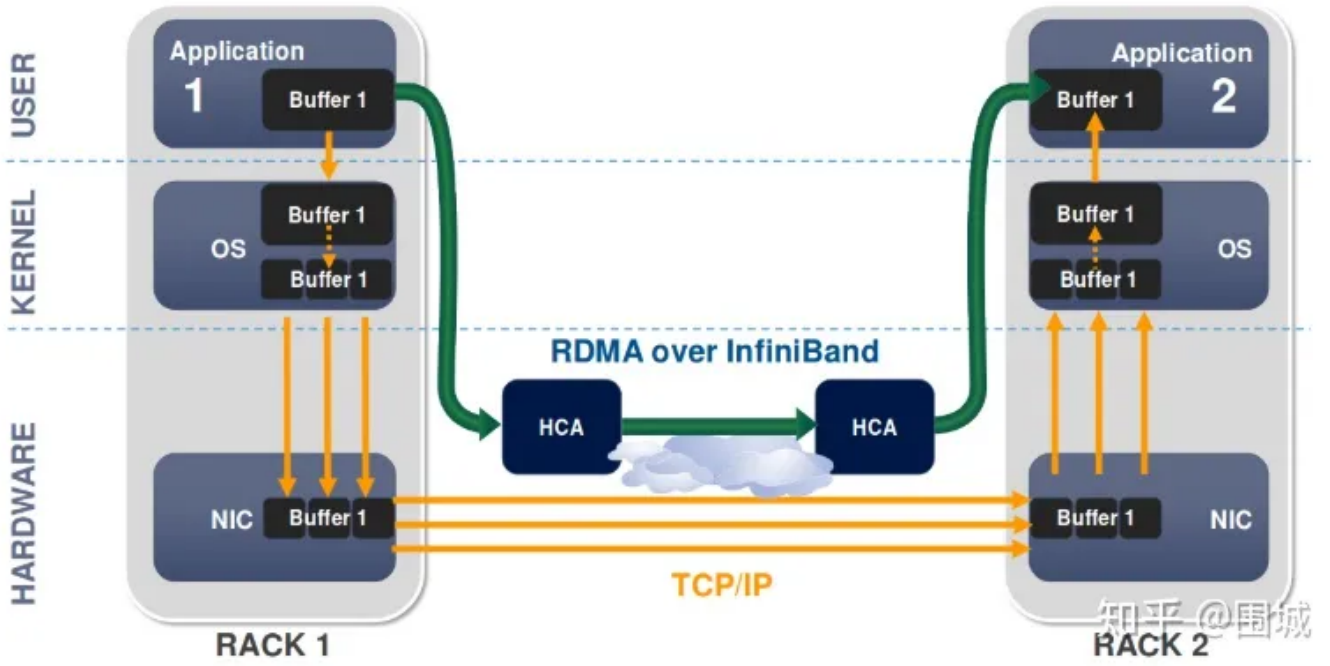
RDMA是一种概念，在两个或者多个计算机进行通讯的时候使用DMA， 从一个主机的内存直接访问另一个主机的内存。



RDMA是一种host-offload, host-bypass技术，允许应用程序(包括存储)在它们的内存空间之间直接做数据传输。具有RDMA引擎的以太网卡(RNIC)--而不是host--负责管理源和目标之间的可靠连接。

使用RNIC的应用程序之间使用专注的QP和CQ进行通讯：

1. 每一个应用程序可以有很多QP和CQ
2. 每一个QP包括一个SQ和RQ
3. 每一个CQ可以跟多个SQ或者RQ相关联



3）RDMA的优势：
传统的TCP/IP技术在数据包处理过程中，要经过操作系统及其他软件层，需要占用大量的服务器资源和内存总线带宽，数据在系统内存、处理器缓存和网络控制器缓存之间来回进行复制移动，给服务器的CPU和内存造成了沉重负担。尤其是网络带宽、处理器速度与内存带宽三者的严重"不匹配性"，更加剧了网络延迟效应。

RDMA是一种新的直接内存访问技术，RDMA让计算机可以直接存取其他计算机的内存，而不需要经过处理器的处理。RDMA将数据从一个系统快速移动到远程系统的内存中，而不对操作系统造成任何影响。

在实现上，RDMA实际上是一种智能网卡与软件架构充分优化的远端内存直接高速访问技术，通过将RDMA协议固化于硬件(即网卡)上，以及支持Zero-copy和Kernel bypass这两种途径来达到其高性能的远程直接数据存取的目标。 使用RDMA的优势如下：

- 零拷贝(Zero-copy) - 应用程序能够**直接执行数据传输**，在不涉及到网络软件栈的情况下。数据能够被直接发送到缓冲区或者能够**直接从缓冲区里接收，而不需要被复制到网络层**。
- 内核旁路(Kernel bypass) - 应用程序可以直接在用户态执行数据传输，不需要在内核态与用户态之间做上下文切换。
- 不需要CPU干预(No CPU involvement) - 应用程序**可以访问远程主机内存而不消耗远程主机中的任何CPU**。远程主机内存能够被读取而不需要远程主机上的进程（或CPU)参与。远程主机的CPU的缓存(cache)不会被访问的内存内容所填充。
- 消息基于事务(Message based transactions) - **数据被处理为离散消息而不是流**，消除了应用程序将流切割为不同消息/事务的需求。
- 支持分散/聚合条目(Scatter/gather entries support) - RDMA原生态支持分散/聚合。也就是说，**读取多个内存缓冲区然后作为一个流发出去或者接收一个流然后写入到多个内存缓冲区里去。**

在具体的远程内存读写中，RDMA操作用于读写操作的远程虚拟内存地址包含在RDMA消息中传送，远程应用程序要做的只是在其本地网卡中注册相应的内存缓冲区。远程节点的CPU除在连接建立、注册调用等之外，在整个RDMA数据传输过程中并不提供服务，因此没有带来任何负载。

4）RDMA的三种不同的硬件实现：
RDMA作为一种host-offload, host-bypass技术，使低延迟、高带宽的直接的内存到内存的数据通信成为了可能。目前支持RDMA的网络协议有：

1. InfiniBand(IB): 从一开始就支持RDMA的新一代网络协议。由于这是一种新的网络技术，因此需要支持该技术的网卡和交换机。
2. RDMA过融合以太网(RoCE): 即RDMA over Ethernet, 允许通过以太网执行RDMA的网络协议。这允许在标准以太网基础架构(交换机)上使用RDMA，只不过网卡必须是支持RoCE的特殊的NIC。
3. 互联网广域RDMA协议(iWARP): 即RDMA over TCP, 允许通过TCP执行RDMA的网络协议。这允许在标准以太网基础架构(交换机)上使用RDMA，只不过网卡要求是支持iWARP(如果使用CPU offload的话)的NIC。否则，所有iWARP栈都可以在软件中实现，但是失去了大部分的RDMA性能优势。

5）名词解释：

1. RNIC：

   > 具有RDMA引擎的以太网卡

2. Fabric：

   > 支持RDMA的局域网(LAN)

3. CA：(Channel Adapter,通道适配器)

   > 是将系统连接到Fabric的硬件组件。
   > 在IBTA中，一个CA就是IB子网中的一个终端结点(End Node)。分为两种类型，一种是HCA, 另一种叫做TCA, 它们合称为xCA。
   > 其中， HCA(Host Channel Adapter)是支持"verbs"接口的CA, TCA(Target Channel Adapter)可以理解为"weak CA", 不需要像HCA一样支持很多功能。
   > 而在IEEE/IETF中，CA的概念被实体化为RNIC（RDMA Network Interface Card）, iWARP就把一个CA称之为一个RNIC。

4. Verbs：

   > 大致可以理解为访问RDMA硬件的“一组标准动作”，每一个Verb可以理解为一个Function

5. MR：（Memory Registration | 内存注册）

   > RDMA 就是用来对内存进行数据传输。那么怎样才能对内存进行传输，很简单，注册。 因为RDMA硬件对用来做数据传输的内存是有特殊要求的。
   > - 在数据传输过程中，应用程序不能修改数据所在的内存。
   > - 操作系统不能对数据所在的内存进行page out操作 -- 物理地址和虚拟地址的映射必须是固定不变的。
   >   注意无论是DMA或者RDMA都要求物理地址连续，这是由DMA引擎所决定的。 那么怎么进行内存注册呢?
   > - 创建两个key (local和remote)指向需要操作的内存区域
   > - 注册的keys是数据传输请求的一部分
   >   注册一个Memory Region之后，这个时候这个Memory Region也就有了它自己的属性：
   > - context : RDMA操作上下文
   > - addr : MR被注册的Buffer地址
   > - length : MR被注册的Buffer长度
   > - lkey：MR被注册的本地key
   > - rkey：MR被注册的远程key
   >   对Memrory Registration：Memory Registration只是RDMA中对内存保护的一种措施，只有将要操作的内存注册到RDMA Memory Region

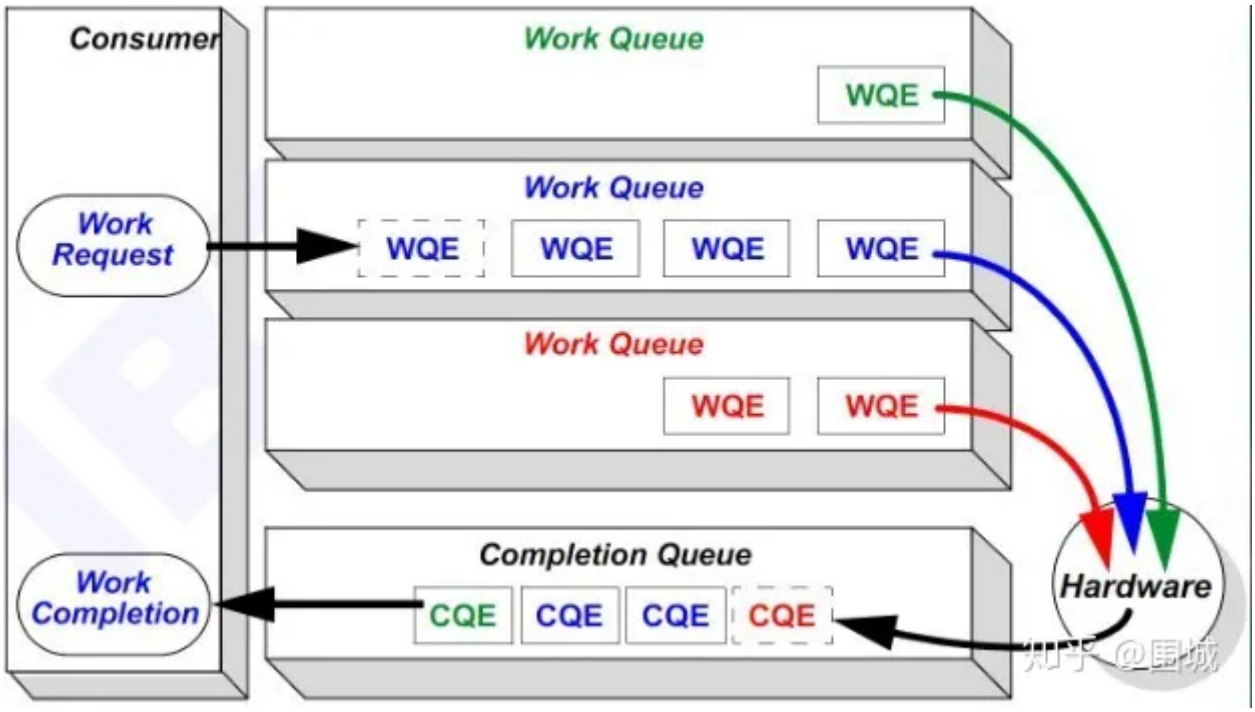
> 中，这快操作的内存就交给RDMA 保护域来操作了。这个时候我们就可以对这快内存进行操作，至于操作的起始地址、操作Buffer的长度，可以根据程序的具体需求进行操作。我们只要保证接受方的Buffer 接受的长度大于等于发送的Buffer长度

6. Queues： QP与CQ // SQ与RQ

> RDMA一共支持三种队列，发送队列(SQ)和接收队列(RQ)，完成队列(CQ)。**其中，SQ和RQ通常成对创建，被称为Queue Pairs(QP)**。

RDMA是基于消息的传输协议，数据传输都是异步操作。 RDMA操作其实很简单，可以理解为：

1. Host提交工作请求(WR)到工作队列(WQ): 工作队列包括发送队列(SQ)和接收队列(RQ)。工作队列的每一个元素叫做WQE, 也就是WR。
2. Host从完成队列(CQ）中获取工作完成(WC): 完成队列里的每一个叫做CQE, 也就是WC。
3. 具有RDMA引擎的硬件(hardware)就是一个队列元素处理器。 RDMA硬件不断地从工作队列(WQ)中去取工作请求(WR)来执行，执行完了就给完成队列(CQ)中放置工作完成(WC)。从生产者-消费者的角度理解就是：
4. Host生产WR, 把WR放到WQ中去
5. RDMA硬件消费WR
6. RDMA硬件生产WC, 把WC放到CQ中去
7. Host消费WC



7. PCP (Payload Compression Protocol)
【1】PCP，全称IP Payload Compression Protocol （IP载荷压缩协议，简称PCP），是一个减少IP数据报长度的协议。
【2】通过压缩数据包，这个协议将在一对通信主机/网关（“节点”）之间提升整体通信性能。倘若节点有足够的计算能力，透过CPU功能或者一个压缩协处理器，在慢速或者拥挤的链路上通信。
【3】IP数据报加密时，IP有效载荷压缩特别有用。加密IP数据报使得数据看起来很随机，在较低协议层压缩效率低（例如，PPP压缩控制协议[RFC-1962]）。如果同时要求压缩和加密，压缩必须在加密之前进行。

6）RDMA数据传输：
6.1 RDMA Send | RDMA发送(/接收)操作 （Send/Recv）

> 跟TCP/IP的send/recv是类似的，不同的是RDMA是基于消息的数据传输协议（而不是基于字节流的传输协议），所有数据包的组装都在RDMA硬件上完成的，也就是说**OSI模型中的下面4层(传输层，网络层，数据链路层，物理层)都在RDMA硬件上完成**。

6.2 RDMA Read | RDMA读操作 (Pull)

> RDMA读操作本质上就是Pull操作, 把远程系统内存里的数据拉回到本地系统的内存里。

6.3 RDMA Write | RDMA写操作 (Push)

> RDMA写操作本质上就是Push操作，把本地系统内存里的数据推送到远程系统的内存里。

6.4 RDMA Write with Immediate Data | 支持立即数的RDMA写操作

> 支持立即数的RDMA写操作本质上就是给远程系统Push(推送)带外(OOB)数据, 这跟TCP里的带外数据是类似的。

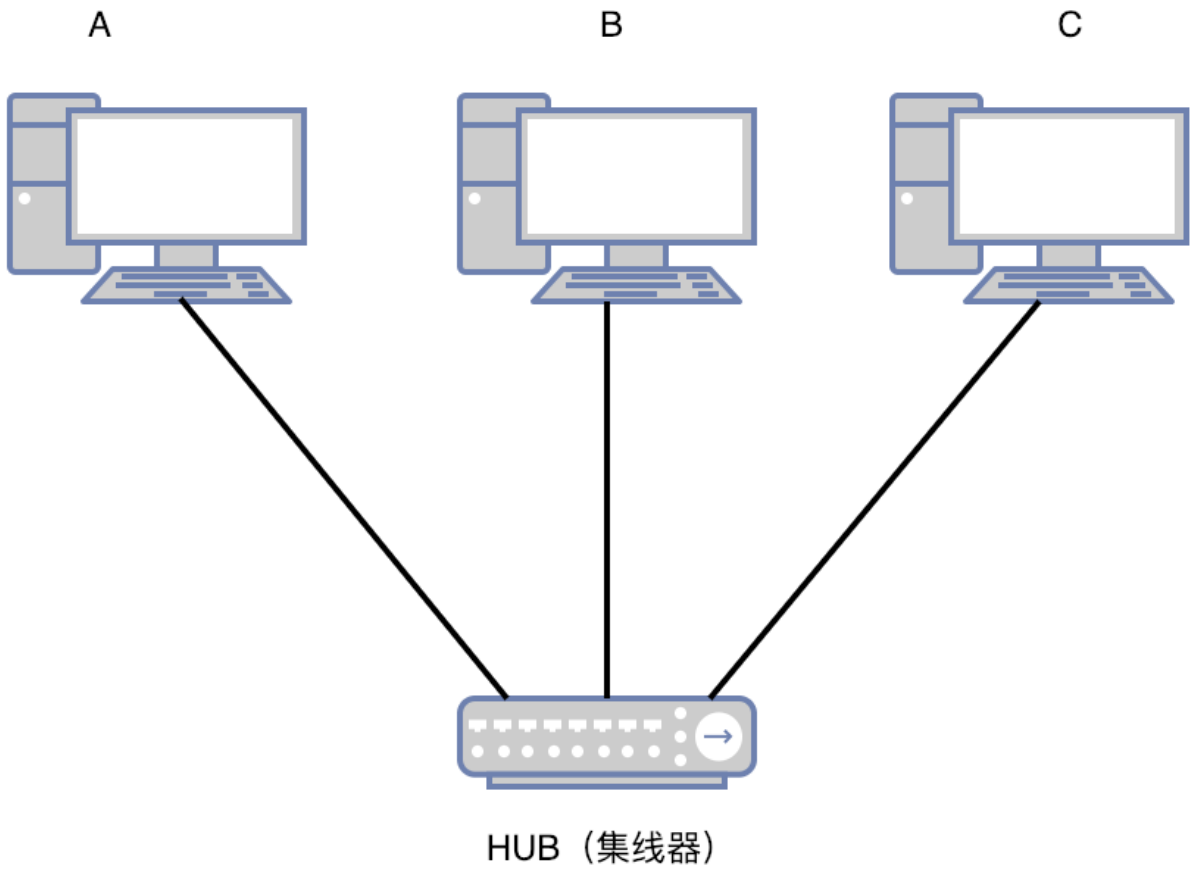
commodity Ethernet (RoCEv2)：

聚合以太网RDMA（**RoCE**）或**以太网InfiniBand（IBoE**）1(https://en.wikipedia.org/wiki/RDMA_over_Converged_Ethernet#cite_note-1)是一种允许通过[以太网](https://en.wikipedia.org/wiki/Ethernet "以太网")网络进行远程直接内存访问（RDMA）的网络协议。它通过以太网封装InfiniBand（IB）传输数据包来做到这一点。

有两个RoCE版本，RoCE v1和RoCE v2。RoCE v1是一种以太网链路层协议，因此允许在同一以太网广播域中的任何两台主机之间进行通信。RoCE v2是一种互联网层协议，这意味着可以路由RoCE v2数据包。虽然RoCE协议受益于聚合以太网网络的特性，但该协议也可以在传统或非聚合以太网网络上使用

VLAN：

1. LAN 本地局域网

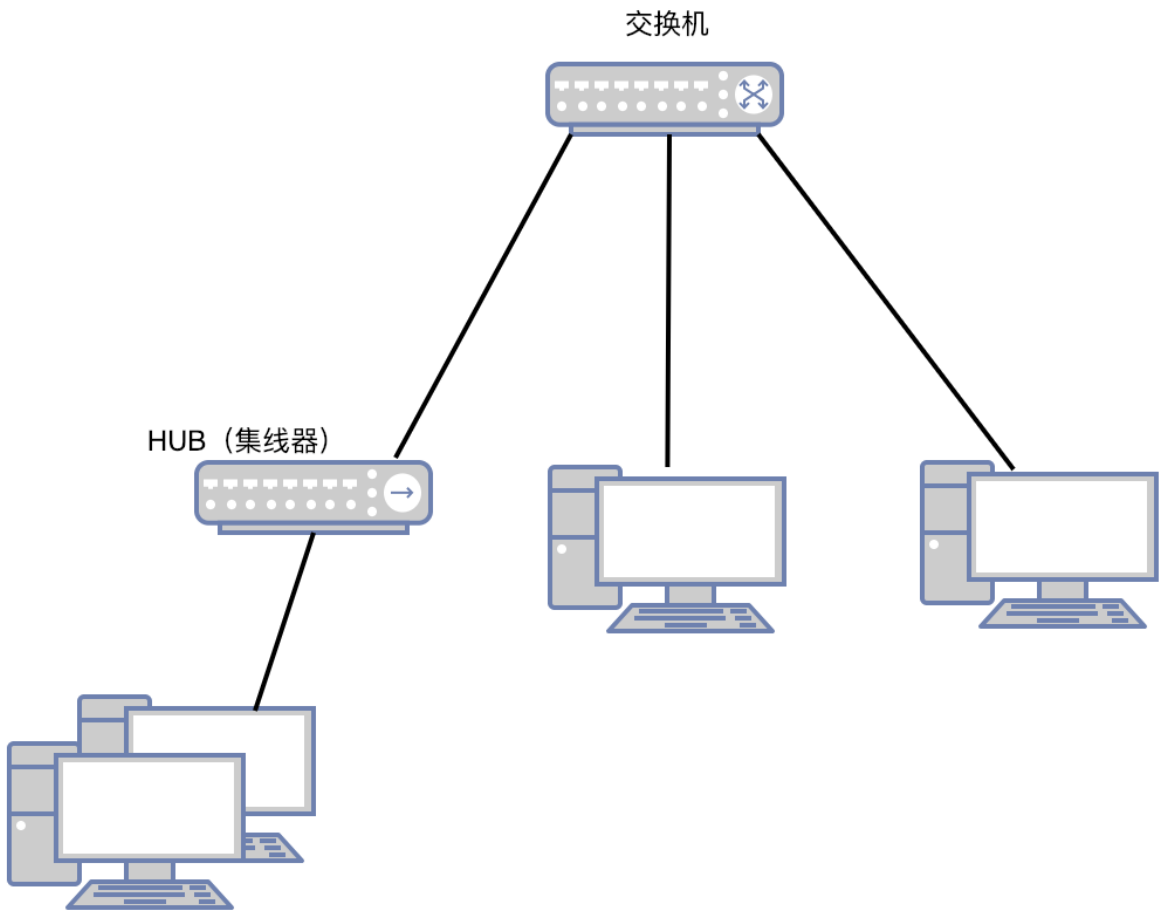
> LAN 表示 Local Area Network，本地局域网。
> 一个 LAN 表示一个广播域，含义是：LAN 中的所有成员都会收到任意一个成员发出的广播包。
>
> 
>

最基本的LAN布局
上图为最基本的LAN布局。如果设备间想要通讯，必须要获取到对方的MAC地址。

> 举例：A 发信息给 C，A 并不知道 C 的 MAC 地址。此时通过 ARP 协议（Address Resolution Protocol；地址解析协议；）获取 C 的 MAC 地址，A 先要广播一个包含目标 IP 地址的 ARP 请求到链接在集线器上的所有设备上，C 接受到广播后返回 MAC 地址给 A，其他设备则丢弃信息。至此已经建立设备间通信的准备条件。

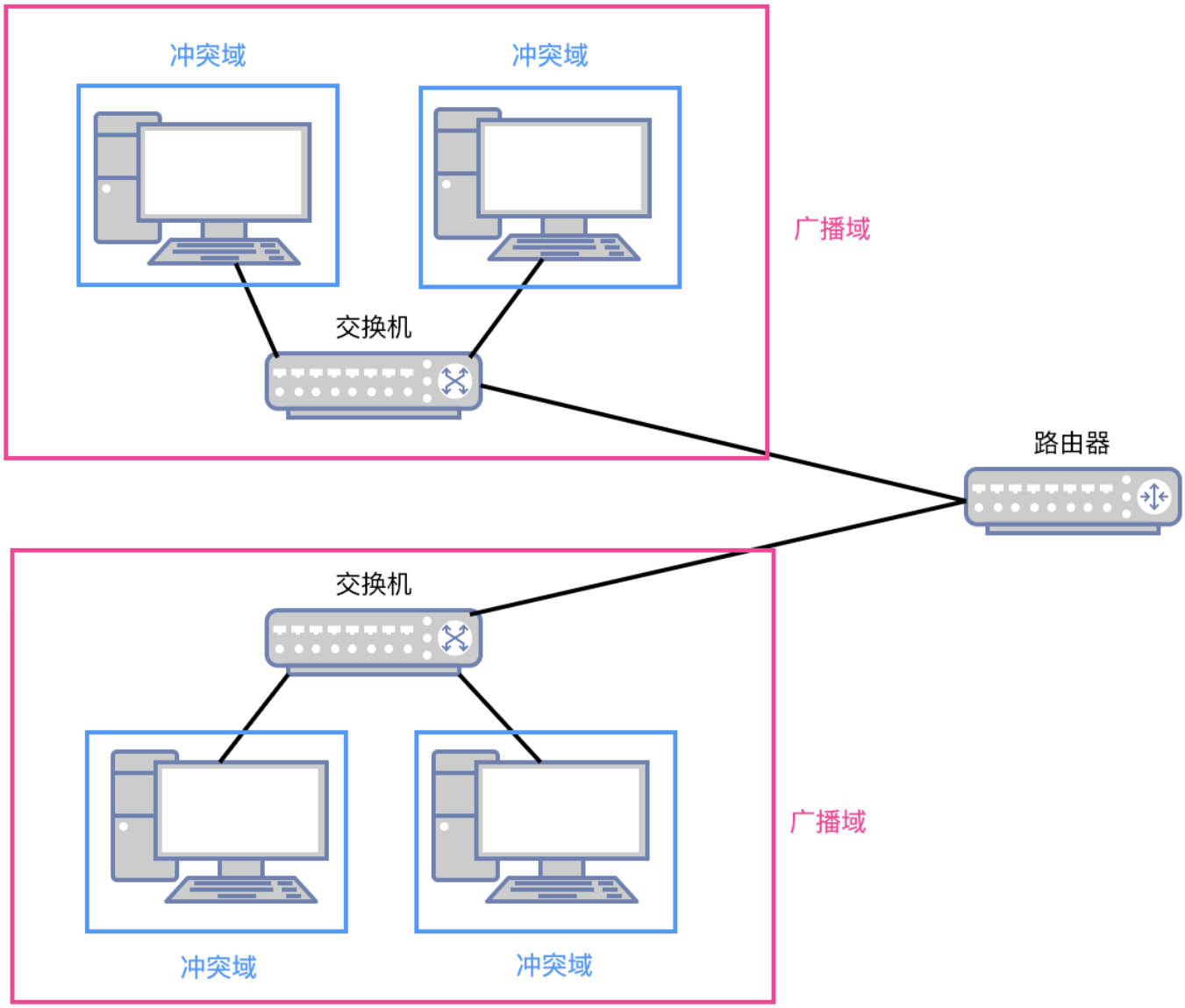
- 链接在集线器中的设备都在同一个冲突域和广播域中。此时的冲突域就是广播域。简单理解就是在这种布局中，一次只能一台设备发送信号且其他设备都能接受该信号。
- 集线器是物理层（OSI第一层）设备，主要作用是将信号进行**接受-恢复放大-发送**，双绞线、光纤在传输信号的时候，随着距离的增大，信号会减弱造成失真，借助集线器可以让信号传播更远的距离；同时集线器上有很多接口，能够扩展终端数量扩大 LAN 的规模。
- 同一集线器上的所有设备共享带宽，如果设备数量过多的话，会造成链路拥堵，严重的会产生广播风暴。

- 使用**交换机**可以把一个大的冲突域划分成多个小的冲突域，这样可以缩减冲突域的范围，降低数据拥堵。
- 下图添加一个交换机连接几个冲突域，交换机的一个端口对应一个单独的冲突域。这样一来一个大的广播域就分成了多个小的冲突域。但注意的是，**这整个网络仍是一个广播域**。



冲突域的相互隔离

- 上面已经把冲突域进行了隔离，当设备越来越多的时候每个设备都发送一个广播，交换机需要把每个广播复制下发到所有设备，这个开销就很可怕了。
- 下图我们使用**路由器**对广播域进行隔离。



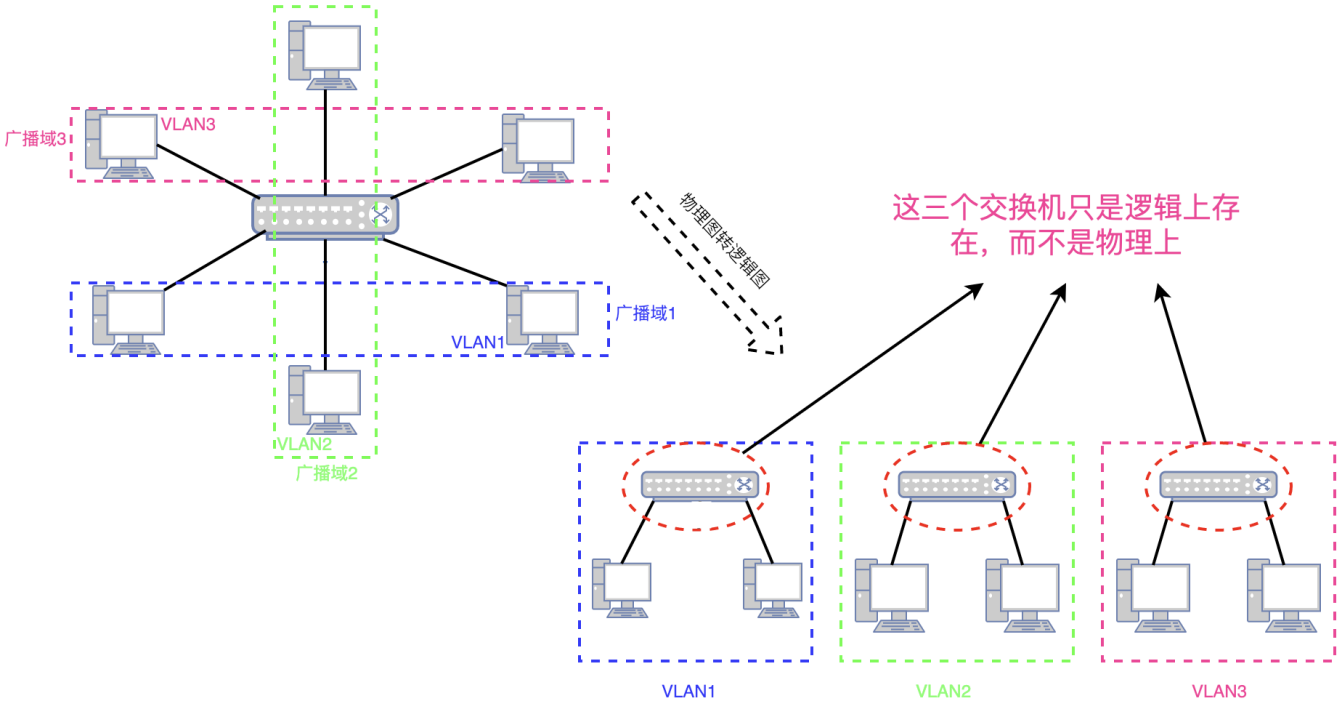
广播域的隔离

- 当路由器收到广播时，会把它自动丢弃，不会转发到路由器的其他端口，实现了广播域的分割
- 这也是路由器存在的意义，如果将全网都集中在一个广播域中则容易引发广播，进而导致全网瘫痪

2. VLAN 虚拟局域网

> VLAN 表示 Virtual Local Area Network，虚拟本地局域网。

- 虚拟局域网（VLAN）是在局域网（LAN）的逻辑上划分成多个广播域，每一个广播域就是一个 VLAN。
- 下图为交换机划分虚拟局域网。交换机把一个广播域划分成了3个广播域，物理上这些设备在一个交换机上，但是逻辑上已经分别划分到三个交换机上，所以会有三个局域网（虚拟局域网），三个广播域。



交换机划分虚拟局域网

> **交换机划分 VLAN 说明，VLAN1 这个虚拟局域网编号，一般工作于管理组，所以普通 VLAN 都是从 2 开始编号，默认情况下，所有虚拟局域网都隶属于 VALN1。**

- 在上图中不同的 VLAN 相互间是不能通讯的。为了解决这个问题，引进了**三层交换机（或路由器)**等 OSI 的三层设备实现跨网段通信。

3. VLAN 实现原理

1. 静态 VLAN

- 静态 VLAN 又被称为基于端口的 VLAN（PortBased VLAN）。是为了明确指定哪个 Port 属于哪个 VLAN ID。
- 在 VLAN 管理员最初配置交换机 Port 和 VLAN ID 的对应关系时，就已经固定了这种对应关系，即**一个 Port 只能对应一个 VLAN ID**，之后无法进行更改，除非管理员再重新配置。
- 当一台设备接到这个 Port 上的时候，怎么判断该主机的 VLAN ID 与 Port 对应呢，这里是根据 IP 配置决定的，我们知道每个 VLAN 都有一个子网号，并对应着哪些 Port，如果设备要求的 IP 地址和该 Port 对应的 VLAN 的子网号不匹配，则连接失败，该设备将无法正常通信。所以除了连接到正确的 Port 外，也必须给设备分配属于该 VLAN 网络段的 IP 地址，这样才能加入到该 VLAN 中。
- 由于需要一个个端口地指定，因此当网络中的计算机数目超过一定数字（比如数百台）后，设定操作就会变得烦杂无比。并且，客户机每次变更所连端口，都必须同时更改该端口所属 VLAN 的设定——这显然不适合那些需要频繁改变拓补结构的网络。

2. 动态 VLAN

- 动态 VLAN 则是根据每个端口所连的计算机，随时改变端口所属的 VLAN。这就可以避免上述的更改设定之类的操作。动态 VLAN 可以大致分为 3 类：
- （1）基于 MAC 的 VLAN
  基于 MAC 地址的 VLAN，就是通过查询并记录端口所连的计算机网卡的 MAC 地址来决定端口的所属。假定有一个 MAC 地址 “A” 被交换机设定为属于 VLAN 10，那么不论 MAC 地址为 “A” 的这台计算机连在交换机哪个端口，该端口都会被划分到 VLAN 10 中去。计算机连在端口 1 时，端口 1 属于 VLAN 10；而计算机连在端口 2 时，则是端口 2 属于 VLAN 10。

基于 MAC 地址的 VLAN，在设定时必须调查所连接的所有计算机的 MAC 地址并加以登录。而且如果计算机交换了网卡，还是需要更改设定。

- （2）基于 IP 的 VLAN
  基于子网的 VLAN，则是通过所连计算机的 IP 地址，来决定端口所属 VLAN 的。不像基于 MAC 地址的 VLAN，即使计算机因为交换了网卡或是其他原因导致 MAC 地址改变，只要它的 IP 地址不变，就仍可以加入原先设定的 VLAN。

因此，与基于 MAC 地址的 VLAN 相比，能够更为简便地改变网络结构。IP 地址是 OSI 参照模型中第三层的信息，所以我们可以理解为基于子网的 VLAN 是一种在 OSI 的第三层设定访问链接的方法。

- （3）基于用户的 VLAN
  基于用户的 VLAN，则是根据交换机各端口所连的计算机上当前登录的用户，来决定该端口属于哪个 VLAN。这里的用户识别信息，一般是计算机操作系统登录的用户，比如可以是 Windows 域中使用的用户名。这些用户名信息，属于 OSI 第四层以上的信息。

DSCP：

DSCP差分服务代码点（Differentiated Services Code Point），IETF于1998年12月发布了Diff-Serv（Differentiated Service）的QoS分类标准。它在每个数据包IP头部的服务类别TOS标识字节中，利用已使用的6比特和未使用的2比特，通过编码值来区分优先级。

它在每个数据包IP头部的服务类别TOS标识字节中，利用已使用的6比特和未使用的2比特，通过编码值来区分优先级.
DSCP 使用6个bit，DSCP的值得范围为0~63。
DSCP 是“IP 优先”和“服务类型”字段的组合。为了利用只支持“IP 优先”的旧路由器，会使用 DSCP 值，因为 DSCP 值与“IP 优先”字段兼容。
用通俗一点的语言解释，其实DSCP就是为了保证通信的QoS，在数据包IP头部的8个标识字节进行编码，来划分服务类别，区分服务的优先级。
每一个DSCP编码值都被映射到一个已定义的PHB（Per-Hop-Behavior）标识码。
通过键入DSCP值，电话、Windows客户和服务器等终端设备也可对流量进行标识。

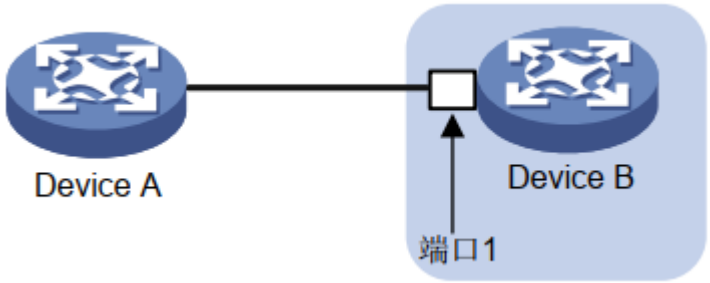
PFC：

PFC 是构建无损以太网的必选手段之一， 能够逐跳提供基于优先级的流量控制。 设备在进行报文转发时，根据报文的优先级进入对应映射关系的队列中进行调度转发。 当某一优先级报文发送速率超过接收速率，导致接收方可用数据缓冲空间不足时， 设备通过PFC PAUSE 帧反馈给上一跳设备，上一跳设备收到 PAUSE 帧报文后停止发送本优先级报文，直到再收到PFC XON帧或经过一定的老化时间后才能恢复流量发送。通过使用 PFC功能，使得某种类型的流量拥塞不会影响其他类型流量的正常转发，从而达到同一链路上不同类型的报文互不影响。

1. PFC 工作机制：

   > PFC使用PFC帧控制优先级队列的发送与停止。

2. PFC PAUSE 帧生成机制：



PFC 功能 PAUSE 帧产生示意图

3. PAUSE 帧产生过程如图1所示：

   > （1）Device B 的端口1收到来自 Device A 的报文后， MMU（Memory Manage Unit，存储器管理单元）会为该报文分配 cell 资源（cell 资源：用来存储数据包的内容，端口会根据报文的实际大小占用相应大小的 cell 资源。比如一个 cell 资源是 208 字节，当发送的报文是 128 字节时，端口会给它分配一个 cell 资源，当发送的报文是 300 字节时，端口会给它分配两个 cell 资源。）， 当设备的 PFC 功能处于开启状态时，会根据报文中的 dot1p优先级统计占用的 cell 资源。
   > （2）**当Device B端口1的某个优先级的报文占用的 cell 资源统计计数达到设置的门限后**， 再收到新的该优先级报文后，端口1会发送对应优先级的**PFC PAUSE帧**给Device A。
   > （3） Device A收到该优先级的PFC PAUSE帧后停止发送对应优先级的报文，**对该优先级的报文进行缓存**，如果触发了**缓存门限**，则也向其上游设备发送 PFC PAUSE 帧，如图2所示。

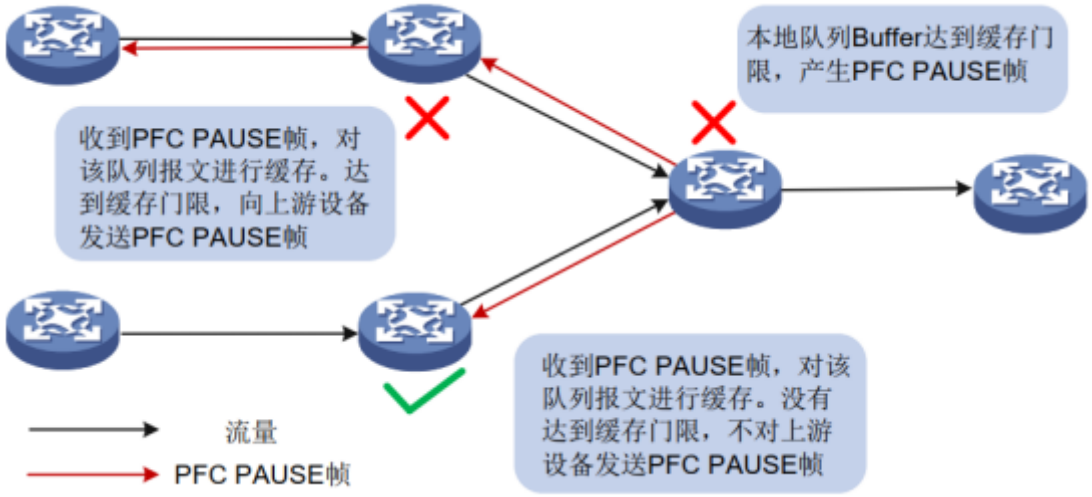


图2 多跳设备之间的PFC PAUSE帧处理

4. 报文优先级与队列映射关系：
   设备在进行报文转发时，将不同优先级的报文放入不同的队列中进行调度转发。 报文优先级与队列映射关系与设备配置的优先级映射方式有关。设备支持的优先级映射配置方式包括： 优先级信任模式方式、端口优先级方式。

4.1 优先级信任模式方式：

配置端口的优先级信任模式后，设备将信任报文自身携带的优先级。通过优先级映射表，使用所信任的报文携带优先级进行优先级映射，根据映射关系完成对报文优先级的修改，以及实现报文在设备内部的调度。
端口的优先级信任模式分为：

> dot1p：信任报文自带的 802.1p 优先级，以此优先级进行优先级映射。

> dscp：信任 IP 报文自带的 DSCP 优先级，以此优先级进行优先级映射。

4.2 端口优先级方式：
未配置端口的优先级信任模式时，设备会将端口优先级作为报文自身的优先级。通过优先级映射表，对报文进行映射。用户可以配置端口优先级，通过优先级映射，使不同端口收到的报文进入对应的队列，以此实现对不同端口收到报文的差异化调度。

接口配置 PFC 功能时，必须配置接口信任报文自带的 802.1p优先级或DSCP 优先级。 接口收到以太网报文，根据优先级信任模式和报文的802.1Q标签状态， 设备为不同优先级的报文标记不同的本地优先级（LP），根据本地优先级进行队列调度，具体过程如图3所示。

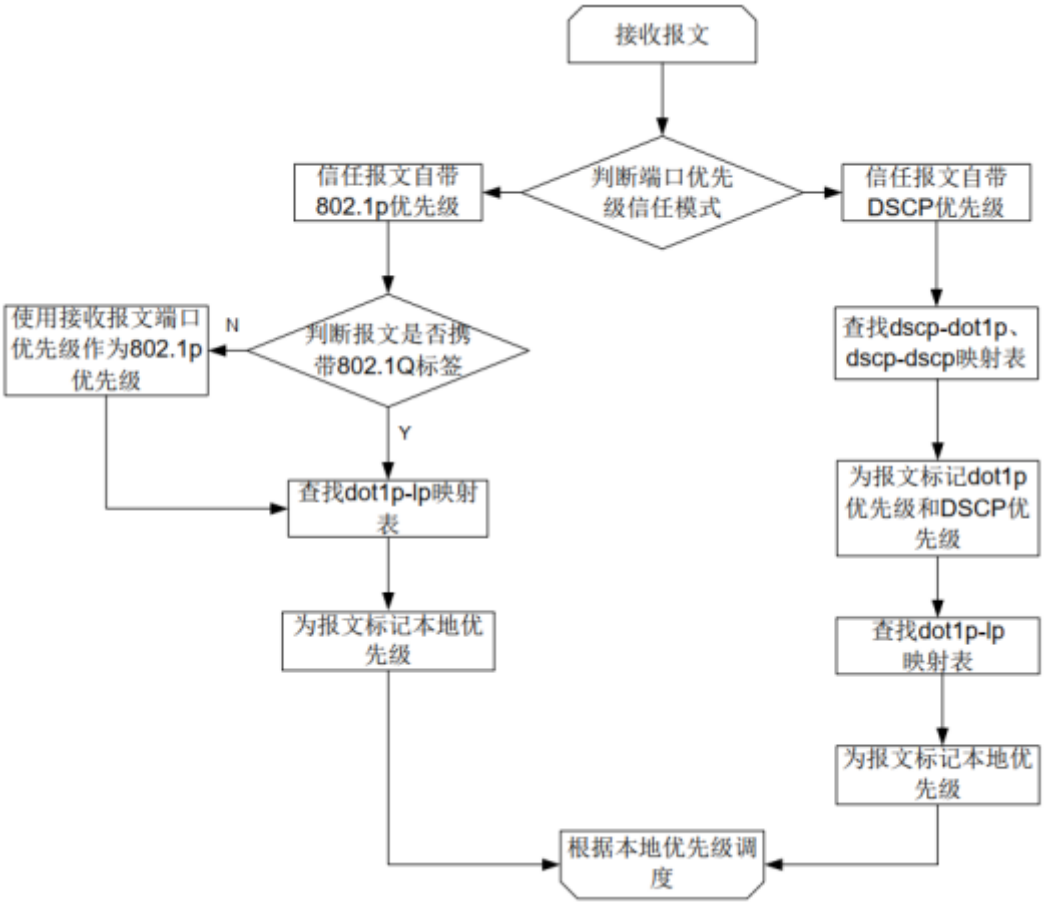


图3 报文优先级与队列映射关系

5. PFC 门限配置：
   通过配置 PFC 缓存门限可以有效解决因缓冲空间不足和入流量队列数量过大， 导致发送数据缓冲区尾丢弃等问题。
   我们先来了解一下接口的缓冲空间设置。接口的缓冲空间分为以下几种：

> **Guaranteed存储空间**： 固定缓冲区，为每一个优先级队列和端口提供最小的缓存保证。 系统会根据用户的配置给队列预留指定大小的空间，即便该队列没有报文存储需求，其他队列也不能抢占。给队列预留的空间均分给每个端口的，即使某端口的某队列没有报文存储需求，其他端口也不能抢占。

> **Shared存储空间**： 共享缓冲区，当端口或优先级的固定缓冲区不够用时， 使用 Shared 存储空间，系统会根据用户配置以及实际需要收发报文的数量决定每个队列实际可占用的缓冲区的大小。如果某个队列没有报文存储需求，则其他队列会抢占该队列的配额。对于某个队列的缓冲区，所有端口接收或发送的报文采用抢占的方式，先到先得，如果资源耗尽，则后到达的报文将被丢弃。

> **Headroom存储空间**： Headroom 缓冲区，当端口PFC功能生效并触发PFC反压帧门限后，本端设备发送PFC PAUSE帧到对端设备让对端停止流量发送的过程中，已经在途的这部分流量的缓存空间，设备需要这些缓冲空间来保证 PFC 流程的不丢包。

PFC 目前提供以下门限设置：

> **Headroom 缓存门限**： Headroom 存储空间中某 802.1p优先级报文的最大使用cell资源。当达到使用的 cell 资源后，该接口会丢弃收到的报文。

> **反压帧触发门限**： Shared 存储空间中某 802.1p 优先级报文在该存储空间使用 cell 资源上限。达到上限后，会触发 PFC 功能发送 PAUSE 帧。 反压帧触发门限又分为动态反压帧触发门限和静态反压帧触发门限：1）**动态反压帧触发门限**：设置某 802.1p 优先级报文触发 PFC PAUSE 帧的可用 cell 资源的百分比。2）**静态反压帧触发门限**：设置某 802.1p 优先级报文触发 PFC PAUSE 帧的可用 cell 资源门限为一个固定值。

> **反压帧停止门限与触发门限间的偏移量**：当某 802.1p优先级报文使用的 cell 资源减小了一个固定值时， 停止发送 PFC PAUSE 帧， 使对端设备恢复流量发送。

> **PFC预留门限**： Guaranteed存储空间中为某 802.1p优先级报文预留的cell资源。

> **Headroom最大可用的 cell 资源**：配置某缓存池（pool，产品具体支持的poolID与产品型号有关，请以设备的实际情况为准） 中，分配给Headroom存储空间的cell资源的大小。

NIC：网络接口控制器

A network interface controller (**NIC**, also known as a **network interface card**,**network adapter**, **LAN adapter** or **physical network interface**, and by similar terms) is a computer hardware component that connects a computer to a computer network

Early network interface controllers：
were commonly implemented on expansion cards that plugged into a computer bus. The low cost and ubiquity of the Ethernet standard means that most newer computers have a network interface built into the motherboard, or is contained into a USB-connected dongle.

Modern network interface controllers：
offer advanced features such as interrupt and DMAinterfaces to the host processors, support for multiple receive and transmit queues, partitioning into multiple logical interfaces, and on-controller network traffic processing such as the TCP offload engine.

Conclusion

> In this paper, we have presented our practices and experiences in deploying RoCEv2 safely at large-scale in Microsoft data centers. Our practices include the introducing of DSCP-based PFC which scales RoCEv2 from layer-2 VLAN to layer-3 IP and the step-by-step onboarding and deployment procedure. Our experiences include the discoveries and resolutions of the RDMA transport livelock, the RDMA deadlock, the NIC PFC storm and the slow-receiver symptom. With the RDMA management and monitoring in place, some of our highlyreliable, latency-sensitive services have been running RDMA for over one and half years

> 在本文中，我们介绍了在 Microsoft 数据中心大规模安全部署 RoCEv2 的实践和经验。我们的实践包括引入基于 DSCP 的 PFC，将 RoCEv2 从第 2 层 VLAN 扩展到第 3 层 IP，以及逐步的加入和部署过程。我们的经验包括发现并解决 RDMA 传输活锁、RDMA 死锁、NIC PFC 风暴和缓慢接收器症状。随着 RDMA 管理和监控到位，我们的一些高度可靠、延迟敏感的服务已经运行 RDMA 超过一年半了

Layer-2 VLAN & Layer-3 IP

背景引入

RDMA 允许用户态的应用程序直接读取和写入远程内存，避免了数据拷贝和上下文切换；并将网络协议栈从软件实现 offload 到网卡硬件，实现了高吞吐量、超低时延和低 CPU 开销的效果。

当前 RDMA 在以太网上的传输协议是 RoCEv2，RoCEv2 是基于无连接协议的 UDP 协议，相比面向连接的 TCP 协议，UDP 协议更加快速、占用 CPU 资源更少，但其传输是不可靠的，*一旦出现丢包会导致 RDMA 的传输效率降低*，这是由 RDMA 的 Go-back-N 重传机制决定的。RDMA 接收方网卡发现丢包时，会丢弃后续接收到的数据包，发送方需要重发之后的所有数据包，这导致性能大幅下降。所以**要想 RDMA 发挥出其性能，需要为其搭建一套不丢包的无损网络环境**

差异化流量分类

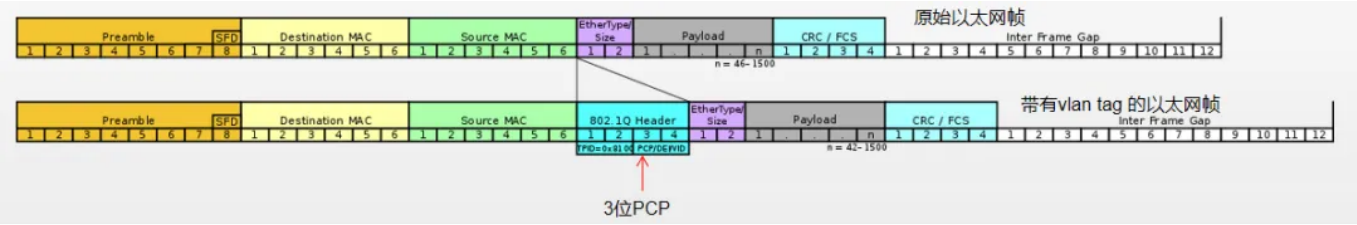
构建无损网络，首先需要对网络流量进行分类，然后针对不同类别流量采用具体流控策略，实现精确控制，避免相互影响。

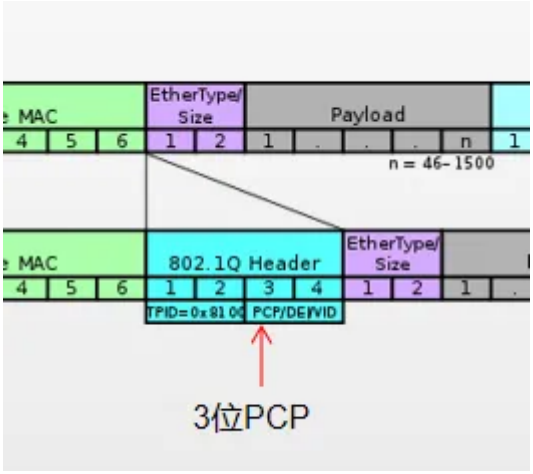
流量分类有两种不同的分类方法：传输层（Layer 2）和网络层（Layer 3）。

【1】Layer2 通过 vlan header（802.1q）里的 PCP（802.1p）位进行分类，对应 CoS（Class of Service）
【2】Layer3 通过 IP header 里的 DSCP 进行分类，对应 DSCP

Layer 2 流量分类

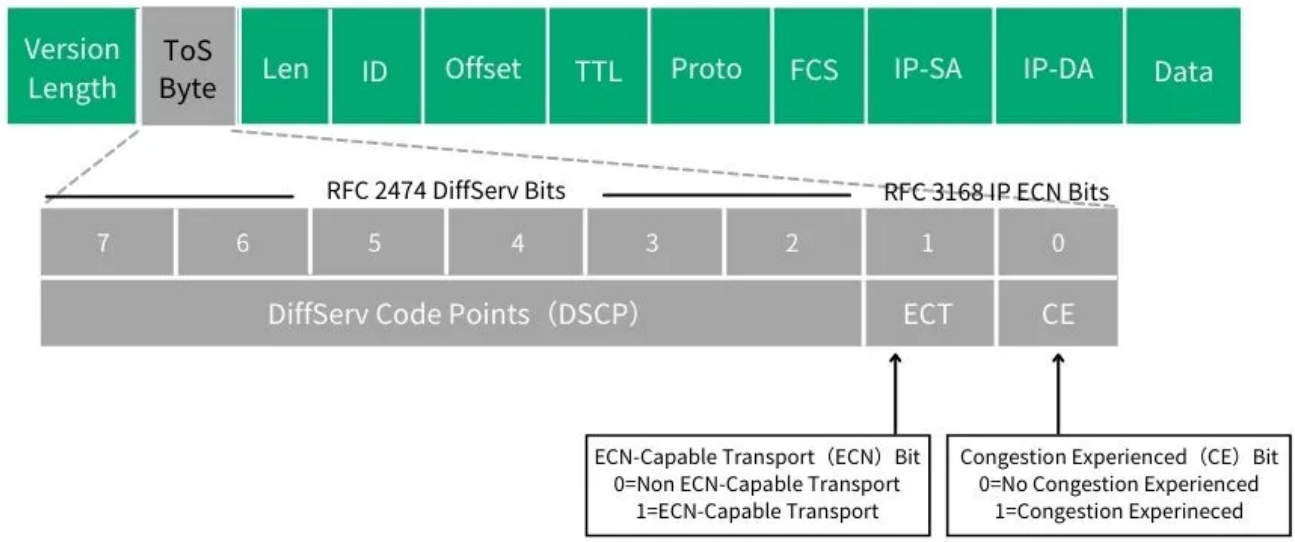
Layer2 层流量分类依据的是 vlan tag 中的 3 位 PCP bit，总共有 8 个类别。3 个 bit 是 Header 中第 3 个 byte 的前三位，如下图。在使用 Layer 2 流量分类时，主机端发出的包需要带有 vlan tag。因此要对网卡配置 vlan，并且设置优先级。因为 L2 层 PFC 需要依靠 vlan，因此包经过三层交换机时可能存在 tag 失效等问题。





Layer 3 流量分类

Layer3 使用 IP 包头中的 TOS 前 6 位（DSCP），支持 64 种不同的流量分类方式，TOS 的后两位用作 Explicit Congestion Notification（ECN）Field，ECN 是一种端到端的流控方式，后面会有介绍。



选择 Layer2 还是 Layer3 层流控

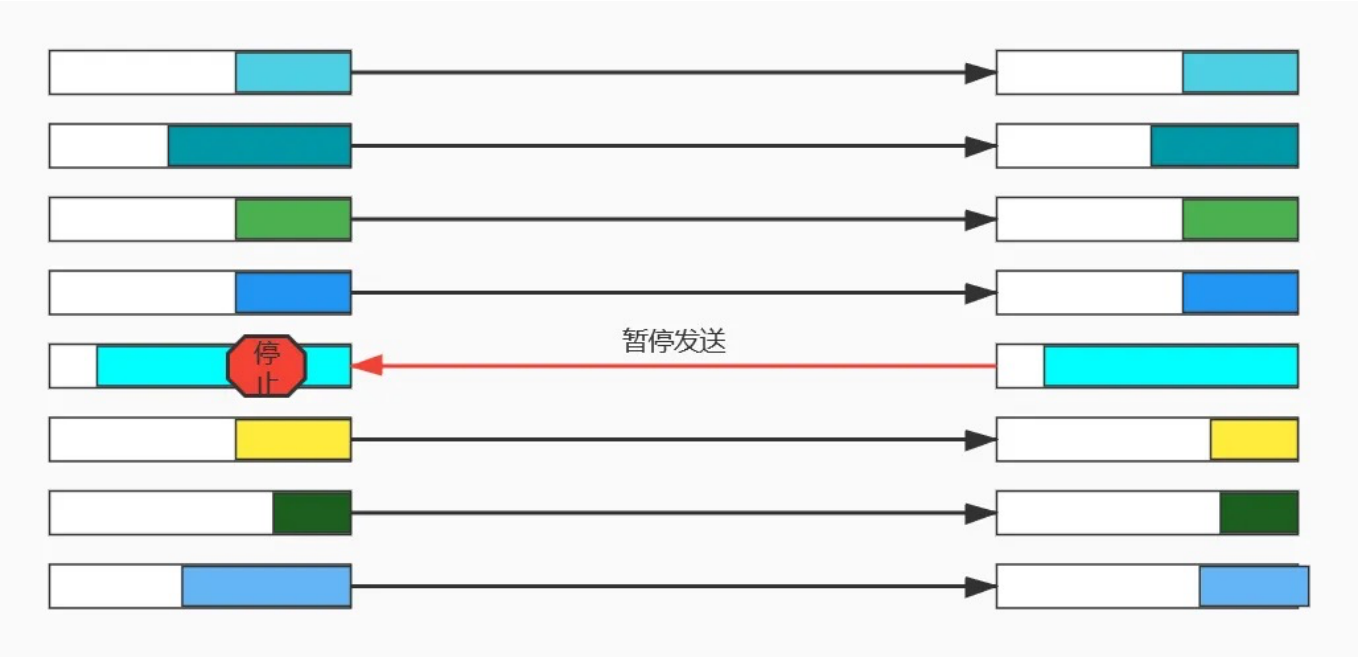
在交换机支持 DSCP 的条件下，建议使用 L3 层的流控方式。从前面的介绍可以看到，**L3 层的控制方式可跨多层交换机，DSCP 值在端到端的传输过程中不会发生变化**。RoCE 使用 UDP 报文进行数据传输，建议 RoCE 的流控使用基于 DSCP 的方式。

构建无损网络—基于 DSCP 或 PCP 的 PFC 流控机制

IEEE 802.1Qbb（Priority-based Flow Control，基于优先级的流量控制）简称 PFC，是 IEEE 数据中心桥接（Data Center Bridge）协议族中的一个技术，是流量控制的增强版。

我们先看一下 IEEE 802.3X（Flow Control）流控的机制：当接收者没有能力处理接收到的报文时，为了防止报文被丢弃，接收者需要通知报文的发送者暂时停止发送。IEEE 802.3X 协议存在一个**缺点：一旦链路被暂停，发送方就不能再发送任何数据包，如果是因为某些优先级较低的数据流引发的暂停，结果却让该链路上其他更高优先级的数据流也一起被暂停了，这是得不偿失的。**

PFC 在基础流控 IEEE 802.3X 基础上进行扩展，【1】允许在一条以太网链路上创建 8 个虚拟通道，并为每条虚拟通道指定相应优先级，【2】允许单独暂停和重启其中任意一条虚拟通道，同时允许其它虚拟通道的流量无中断通过。【3】PFC 将流控的粒度从物理端口细化到8个虚拟通道，分别对应 Smart NIC 硬件上的 8 个硬件发送队列，如下图。



对比二层与三层流控

在二层网络中，PFC 使用 vlan 中的 PCP 位来对数据流进行区分；在三层网络中，PFC 既可以使用 PCP，也可以使用 DSCP，使得不同数据流可以享受到独立的流控制。

当下数据中心因多采用三层网络，且 DSCP 值在端到端的传输过程中不会发生变化，故推荐使用 DSCP。

RDMA 无损网络中利用 PFC 流控机制，实现了交换机端口缓存溢出前暂停对端流量，阻止了丢包现象发生，但因为需要一级一级反压，效率较低，而且存在不公平问题和 Head-of-Line 堵塞问题。此外，PFC 是通过下游网络设备对上游设备的控制方式达到不丢包的目的，但最有效的流控应该是控制产生数据的源端主机的发送速度，使得主机往网络中注入数据速度放缓，这是解决问题的根本方法。