“架构师技术联盟”微信公众号专注技术架构和行业解决方案，构建专业交流平台，分享一线技术实践。

《RDMA 原理分析、对比和技术实现解析》

公众号作者牺牲业余休息时间，把历史原创文章进行了总结、归类和细化，梳理成电子书，重点书目如下所示(持续更新…)：

《数据备份和副本管理技术全面解析》
《容器技术架构、网络和生态详解》
《闪存技术、产品和发展趋势全面解析》
《虚拟化技术最详细解析》
《传统企业存储知识完全解析》
《IO 知识和系统性能深度调优全解》
《业界主流数据中心存储双活完全解析》
《Ceph 技术架构、生态和特性详细对比分析》
《数据中心大二层交换技术详解》
《VMware 云数据中心(私有云)解决方案详解》
《大数据时代数据重删技术详解》
《高性能计算 HPC 技术、方案和行业全面解析》
《RDMA 原理分析、对比和技术实现解析》

说明：免费电子书下载地址(实时更新防止链接失效)->请关注“架构师技术联盟”微信号或在“架构师电子书店”首页，按照提示语或说明获取下载地址。

“架构师技术联盟”微信公众号

架构师电子书店

目录

摘要：远程直接内存访问(即 Remote Direct Memory Access)是一种直接内存访问技术，它将数据直接从一台计算机的内存传输到另一台计算机，无需双方操作系统的介入。

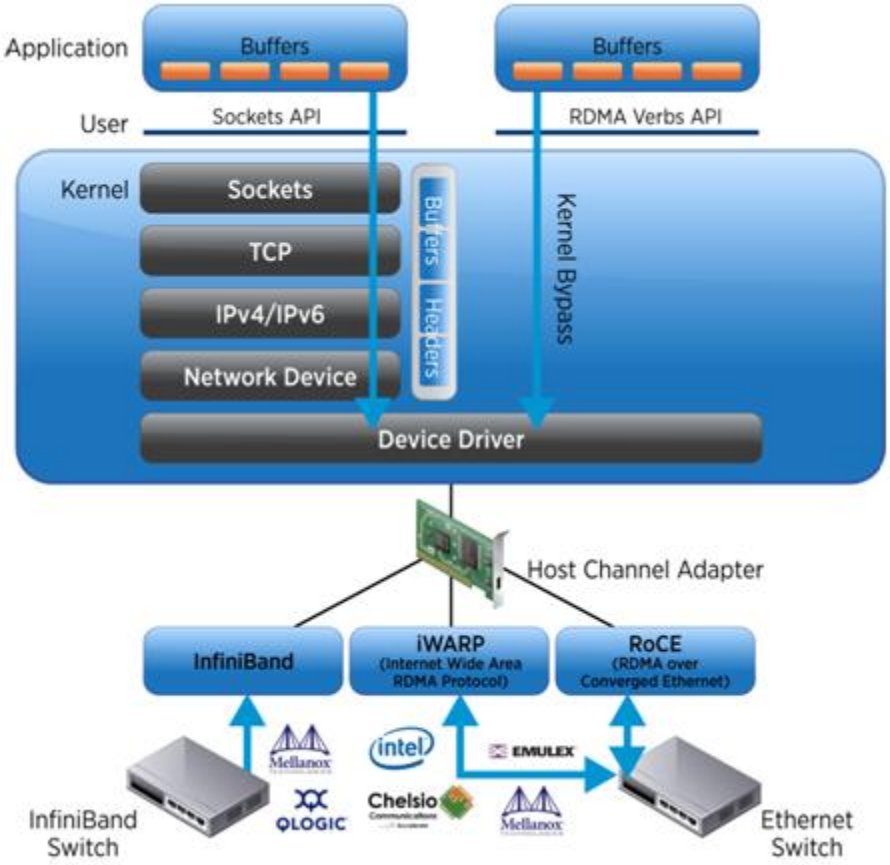
第1章 RDMA 背景简介

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。

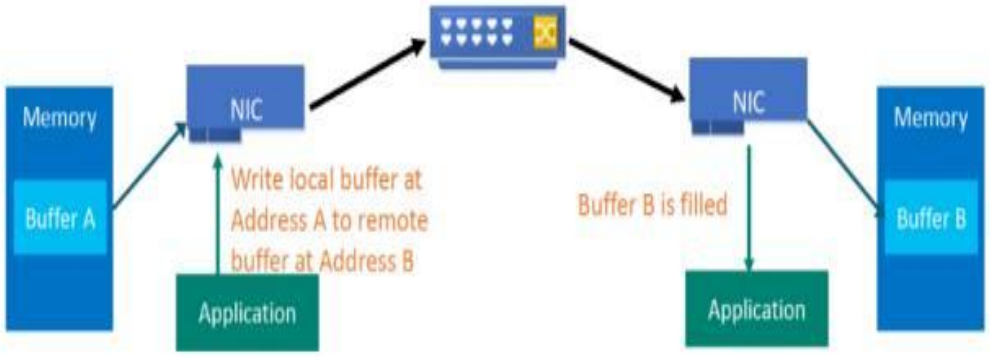
1.1 RDMA 技术背景

RDMA 技术最早出现在 Infiniband 网络，用于 HPC 高性能计算集群的互联。传统的基于 socket 套接字（TCP/IP 协议栈）的网络通信，需要经过操作系统软件协议栈，数据在系统 DRAM、处理器 CACHE 和网卡 buffer 之间来回拷贝搬移，因此占用了大量的 CPU 计算资源和内存总线带宽，也加大了网络延时。举例来说，40Gbps 的 TCP 流，能耗尽主流服务器的所有 CPU 资源。RDMA 解决了传统 TCP/IP 通信的技术痛点：40Gbps 场景下，CPU 占用率从 100%下降到 5%，网络延时从 ms 级降低到 10us 以下。

RDMA 是一种新的内存访问技术，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耗时的处理。RDMA 将数据从一个系统快速移动到远程系统存储器中，而不对操作系统造成任何影响。RDMA 技术的原理及其与 TCP/IP 架构的对比如下图所示。



因此，RDMA 可以简单理解为利用相关的硬件和网络技术，服务器 1 的网卡可以直接读写服务器 2 的内存，最终达到高带宽、低延迟和低资源利用率的效果。如下图所示，应用程序不需要参与数据传输过程，只需要指定内存读写地址，开启传输并等待传输完成即可。



1.2 RDMA 标准组织

2001 年 10 月，Adaptec、Broadcom、Cisco、Dell、EMC、HP、IBM、Intel、Microsoft 和 NetApp 公司宣布成立"远程直接内存访问(RDMA)联盟"。RDMA 联盟是个独立的开放组织，其制定实施能提供 TCP/IP RDMA 技术的产品所需的体系结构规范，鼓励其它技术公司积极参与新规范的制定。该联盟将负责为整个 RDMA 解决方案制定规范，包括 RDMA、DDP(直接数据放置)和 TCP/IP 分帧协议。

RDMA 联盟是 Internet 工程任务组(IETF)的补充，IETF 是由网络设计师、运营商、厂商和研究公司组成的大型国际组织。其目的是推动 Internet 体系结构的发展，并使 Internet 的运作更加顺畅。RDMA 联盟的成员公司和个人都是 IETF 的积极参与者。另外，IETF 还认识到了 RDMA 在可行网络方案中的重要性，并计划在以后几个月里成立"Internet 协议套件 RDMA"工作组。RDMA 联盟协议规定，联盟将向相应的 IETF 工作组提交规范草案，供 IETF 考虑。

TCP/IP RDMA 体系结构规范的 1.0 版本于 2002 年 10 月由 RDMA 联盟成员发布， TCP/IP RDMA 的最终规范将由 RDMA 联盟的业界合作伙伴及相应的业界标准组织派出的代表共同确定。

第2章 哪些网络协议支持 RDMA

2.1 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。

2.2 RDMA 过融合以太网(RoCE)

一种允许通过以太网进行 RDMA 的网络协议。其较低的网络头是以太网头，其上网络头(包括数据)是 InfiniBand 头。这允许在标准以太网基础架构(交换机)上使用 RDMA。只有 NIC 应该是特殊的，并支持 RoCE。

2.3 互联网广域 RDMA 协议(iWARP)

允许通过 TCP 执行 RDMA 的网络协议。在 IB 和 RoCE 中存在功能，iWARP 不支持这些功能。这允许在标准以太网基础架构(交换机)上使用 RDMA。只有 NIC 应该是特殊的，并支持 iWARP(如果使用 CPU 卸载)，否则所有 iWARP 堆栈都可以在 SW 中实现，并且丢失了大部分的 RDMA 性能优势。

第3章 RDMA 技术优势

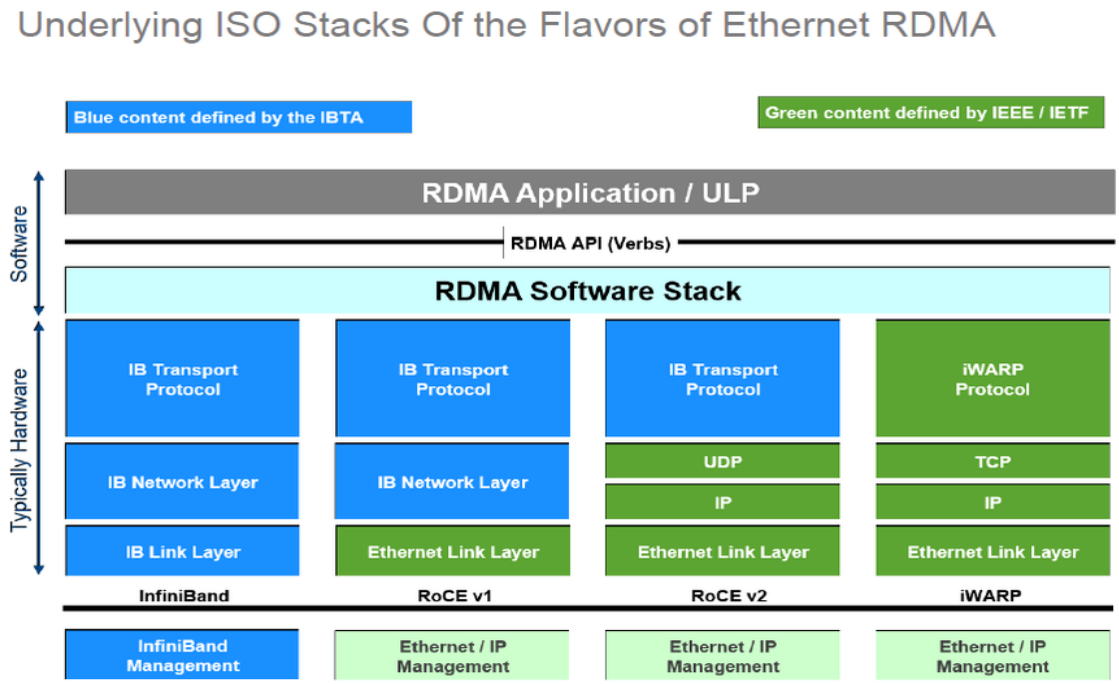
在实现上，RDMA 实际上是一种智能网卡与软件架构充分优化的远端内存直接高速访问技术，通过在网卡上将 RDMA 协议固化于硬件，以及支持零复制网络技术和内核内存旁路技术这两种途径来达到其高性能的远程直接数据存取的目标。

- （1）**零复制：**零复制网络技术使网卡可以直接与应用内存相互传输数据，从而消除了在应用内存与内核之间复制数据的需要。因此，传输延迟会显著减小。
- （2）**内核旁路：**内核协议栈旁路技术使应用程序无需执行内核内存调用就可向网卡发送命令。在不需要任何内核内存参与的条件下，RDMA 请求从用户空间发送到本地网卡并通过网络发送给远程网卡，这就减少了在处理网络传输流时内核内存空间与用户空间之间环境切换的次数。
- （3）**没有 CPU 参与：**应用程序可以访问远程内存，而不占用远程机器中的任何 CPU。远程存储器将被读取，无需任何干预的远程进程(或处理器)。远程 CPU 中的缓存将不会被访问的内存内容填满。
- （4）**基于消息的事务：**数据被作为离散消息处理，而不是作为流，这消除了应用将流分成不同消息/事务的需要。
- （5）**分散/收集条目支持：**RDMA 支持本地处理多个分散/收集条目，即读取多个内存缓冲区并将其作为一个流或获取一个流并将其写入多个内存缓冲区。

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。

第4章 RDMA 有哪些不同实现

如前文所述，RDMA 最早在 Infiniband 传输网络上实现，技术先进，但是价格高昂（只有 Mellanox 和 Intel 供应商提供全套网络解决方案），应用局限在 HPC 高性能计算领域。后来业界厂家把 RDMA 移植到传统 Ethernet 以太网上，降低了 RDMA 的使用成本，推动了 RDMA 技术普及。在 Ethernet 以太网上，根据协议栈融合度的差异，分为 iWARP 和 RoCE 两种技术，而 RoCE 又包括 RoCEv1 和 RoCEv2 两个版本（RoCEv2 的最大改进是支持 IP 路由）。各 RDMA 网络协议栈的对比如下图所示。



其中，InfiniBand 是最早实现 RDMA 的网络协议，被广泛应用到高性能计算中。但是 InfiniBand 和传统 TCP/IP 网络的差别非常大，需要专用的硬件设备，承担昂贵的价格。鉴于此，这里不对 InfiniBand 做过多的讨论。

以 RoCEv2 为例,如下图所示,RoCEv2 的协议栈包括 IB 传输层、TCP/UDP、IP 和 Ethernet，其中后面三层都使用了 TCP/IP 中相应层次的封包格式。UDP 包头中,目的端口号为 4791 即代表是 RoCEv2 帧。IB BTH 即 InfiniBand Base Transport Header，定义了 IB 传输层的相应头部字段。IB Payload 即为消息负载。ICRC 和 FCS 分别对应冗余检测和帧校验。

- IB BTH 格式和字段定义如下图。其中：

  1. Opcode 用于表明该包的 type 或 IB payload 中更高层的协议类型。

  2. S 是 Solicited Event 的缩写，表明回应者应该产生一个事件。

  3. M 是 MigReq 的缩写，一般用于迁移状态。

  4. Pad 表明有多少补齐字节被填充到 IB payload 中。

  5. TVer 即 Transport Header Version,表明该包的版本号。

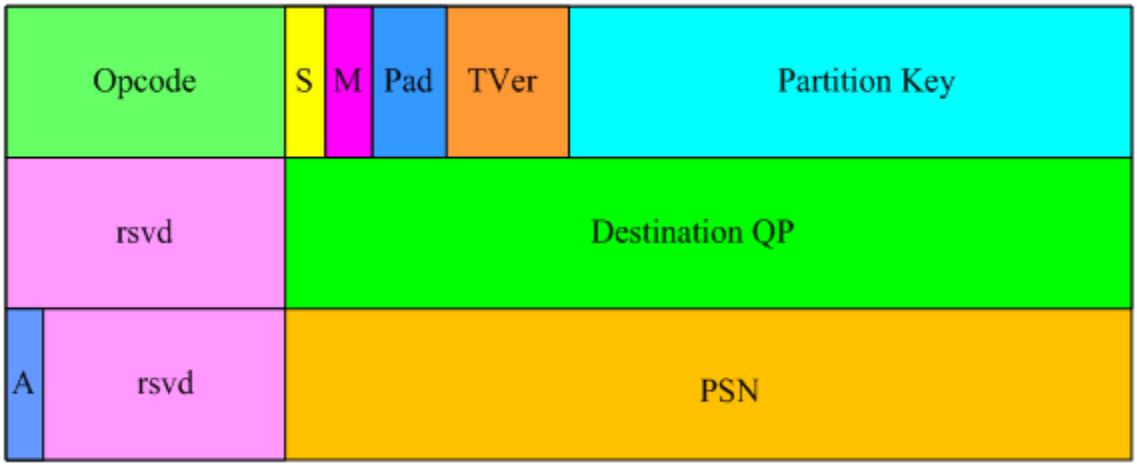
  6. Partition Key 用来表征与本 packet 关联的逻辑内存分区。

  7. rsvd 为保留字段。

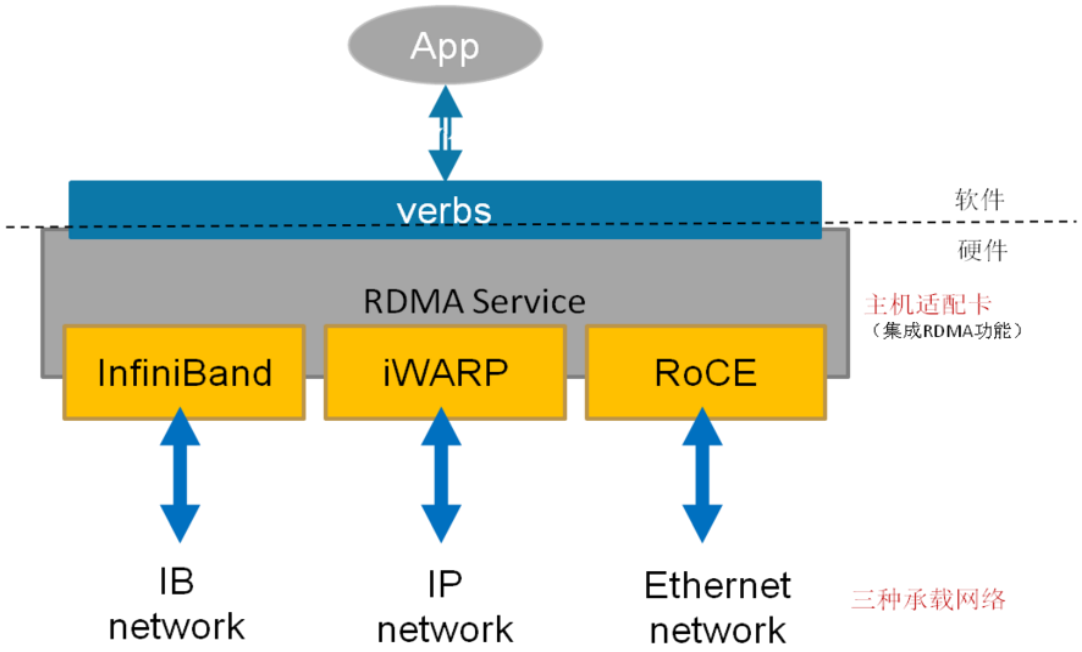
  8. Destination QP 表明目的端 Queue Pair 序号。

  9. A 是 Acknowledge Request，表示该 packet 的应答可由响应者调度。

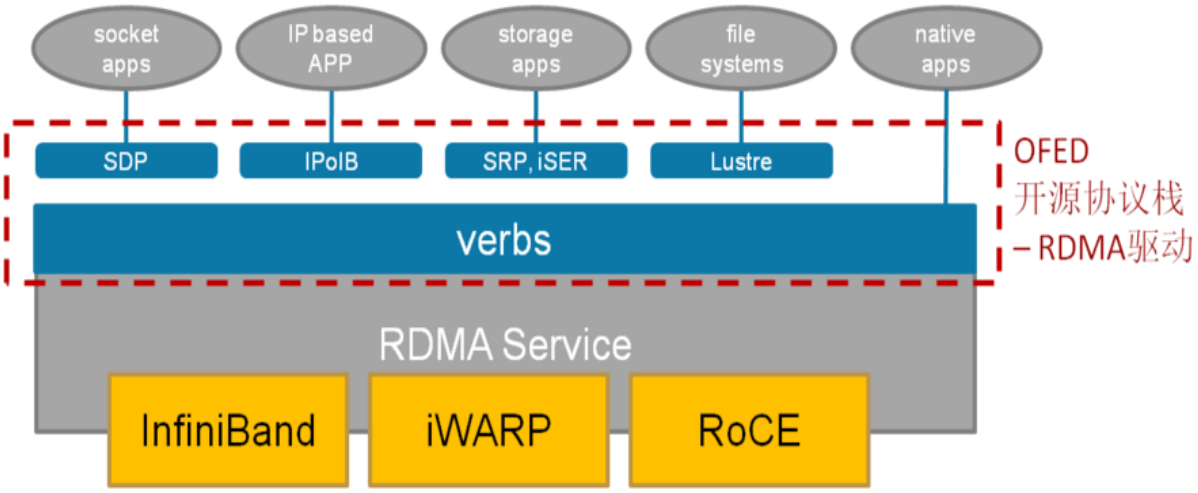
  10. PSN 是 Packet Sequence Number，用来检测丢失或重复的数据包。



在 RoCEv2 协议栈中，IB BTH、UDP、IP 以及 Ethernet Layer 全是固化在网卡上的。应用程序 APP 通过 OFA Stack（亦或其他组织编写的 RDMA stack）提供的 verbs 编程接口（比如 WRITE、READ、SEND 等）打包 IB payload，接下来便直接进入硬件，由 RDMA 网卡实现 payload 的层层封装，软硬件协议栈如下图所示。

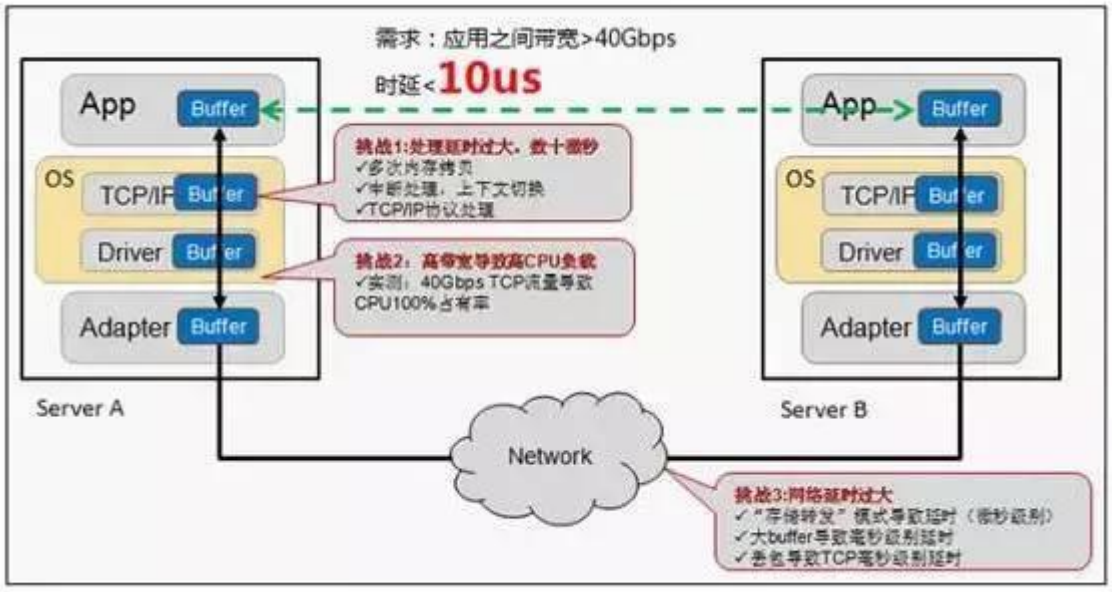


值得注意的是，**第三方组织 OpenFabric 联盟推出了 OFED 协议栈扩展 RDMA verbs 接口，兼容现有应用**，使得现有应用无需重构即可受益于 RDMA（性能成倍提升）。



第5章 RDMA 其他相关标准组织

面对高性能计算、大数据分析和浪涌型 IO 高并发、低时延应用，现有 TCP/IP 软硬件架构和应用高 CPU 消耗的技术特征根本不能满足应用的需求。这要有体现在**处理延时过大，数十微秒；多次内存拷贝、中断处理，上下文切换、复杂的 TCP/IP 协议处理、网络延时过大、存储转发模式和丢包导致额外延时**。接下来我们继续讨论 **RDMA 技术、原理和优势**，看完文章你就会发现为什么 RDMA 可以更好的解决这一系列问题。

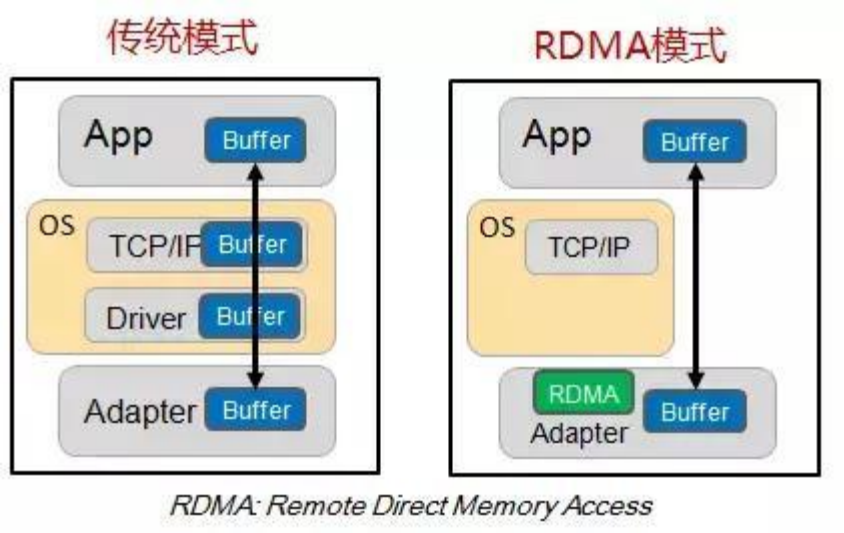


RDMA 最早专属于 Infiniband 架构，随着在网络融合大趋势下出现的 RoCE 和 iWARP，这使高速、超低延时、极低 CPU 使用率的 RDMA 得以部署在目前使用最广泛的以太网上。

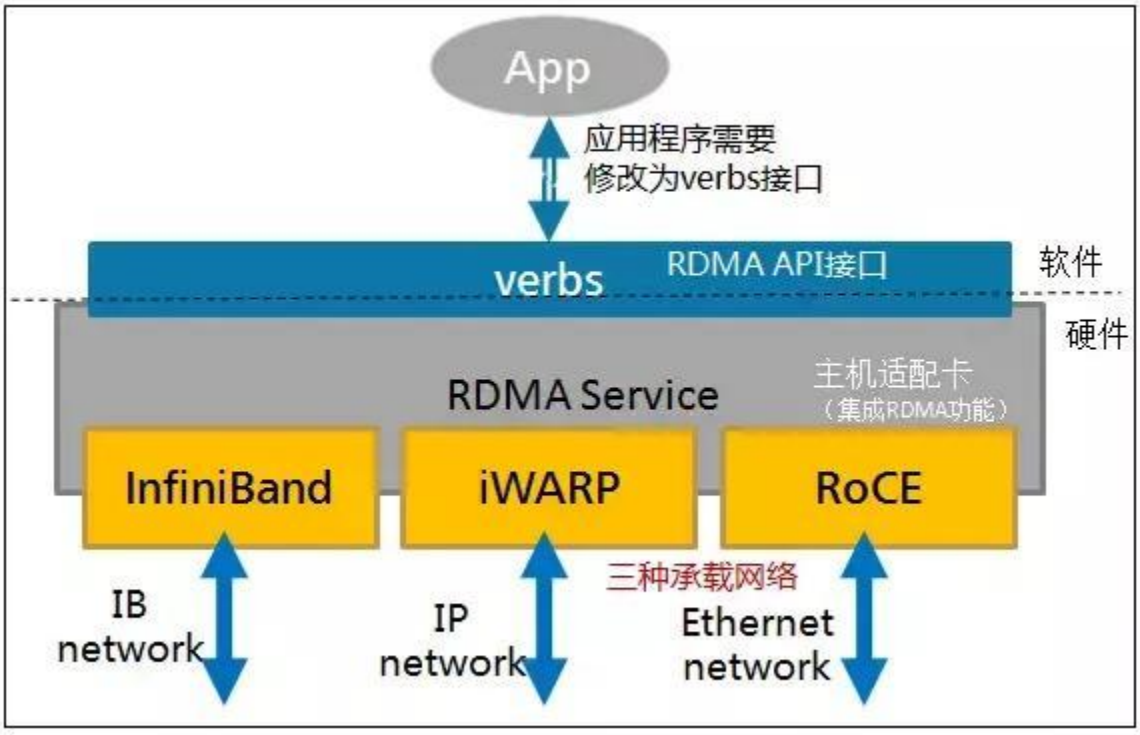
RDMAC(RDMA Consortium) 和 IBTA(InfiniBand Trade Association)主导了 RDMA 发展，RDMAC 是 IETF 的一个补充并主要定义的是 iWRAP 和 iSER，IBTA 是 infiniband 的全部标准制定者，并补充了 RoCE v1 v2 的标准化。IBTA 解释了 RDMA 传输过程中应具备的特性行为，而传输相关的 Verbs 接口和数据结构原型是由另一个组织 OFA(Open Fabric Alliance)来完成。

相比传统 DMA 的内部总线 IO，RDMA 通过网络在两个端点的应用软件之间实现 Buffer 的直接传递；相比比传统的网络传输，RDMA 又无需操作系统和协议栈的介入。RDMA 可以轻易实现端点间的超低延时、

超高吞吐量传输，而且基本不需要 CPU、OS 等资源介入，也不必再为网络数据的处理和搬移耗费过多其他资源。

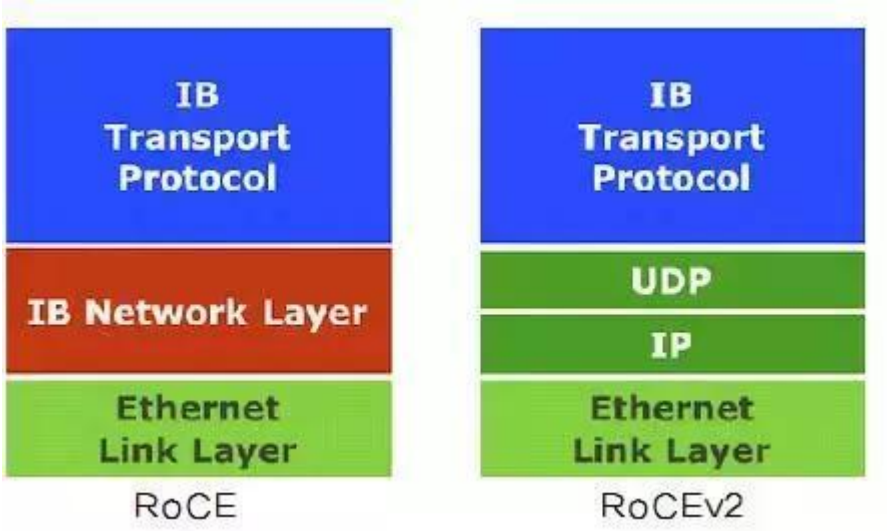


InfiniBand 通过以下技术保证网络转发的低时延(**亚微秒级**)，采用 Cut-Through 转发模式，减少转发时延；基于 Credit 的流控机制，保证无丢包；硬件卸载；Buffer 尽可能小，减少报文被缓冲的时延 。



iWARP(RDMA over TCP/IP) 利用成熟的 IP 网络；继承 RDMA 的优点；TCP/IP 硬件实现成本高，但如果采用传统 IP 网络丢包对性能影响大。

RoCE 性能与 IB 网络相当；DCB 特性保证无丢包；需要以太网支持 DCB 特性；以太交换机时延比 IB 交换机时延要稍高一些。



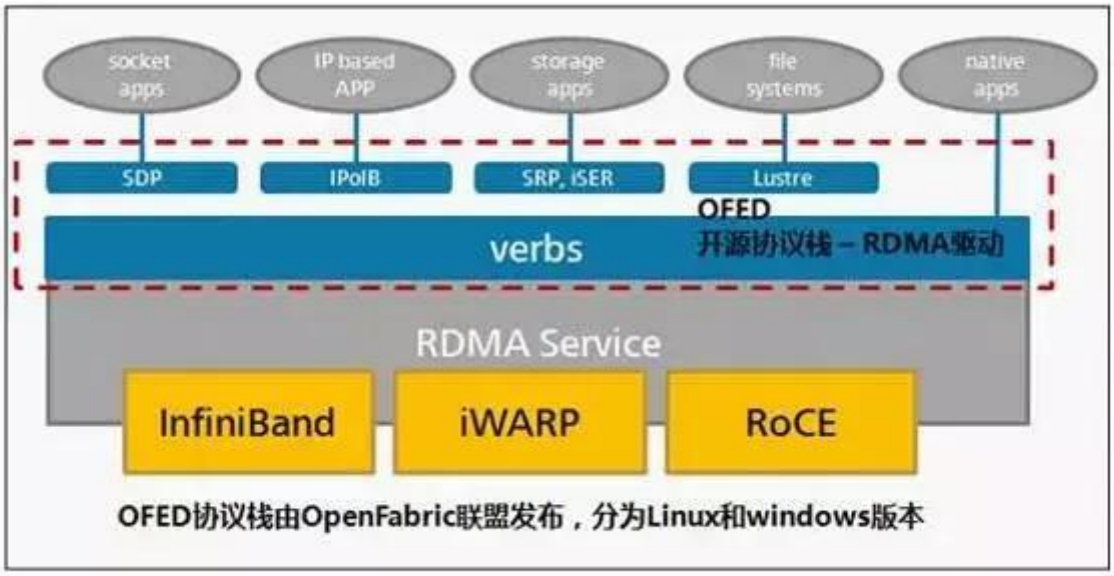
RoCEv2 针对 RoCE 进行了一些改进,如引入 IP 解决扩展性问题,可以跨二层组网; 引入 UDP 解决 ECMP 负载分担等问题。

	InfiniBand（IB）	iWARP	RoCE
标准组织	IBTA	IETF	IBTA
性能	最好	稍差（受TCP影响）	与IB相当
成本	高	中	低
网卡厂商	Mellanox 40Gbps	Chelsio 10Gbps	Mellanox-40Gbps Emulex-10/40Gbps

基于 InfiniBand 的 RDMA 是在 2000 年发布规范,属于原生 RDMA; 基于 TCP/IP 的 RDMA 称作 iWARP, 在 2007 年形成标准, 主要包括 MPA/ DDP/ RDMAP 三层子协议; 基于 Ethernet 的 RDMA 叫做 RoCE, 在 2010 年发布协议,基于增强型以太网并将传输层换成 IB 传输层实现。

第6章 应用和 RNIC 传输接口层

扩展 RDMA API 接口以兼容现有协议/应用, **OFED(Open Fabrics Enterprise Distribution)协议栈由 OpenFabric 联盟发布, 分为 Linux 和 windows 版本, 可以无缝兼容已有应用**。通过使已有应用与 RDMA 结合后, 性能成倍提升。



应用和 RNIC(RDMA-aware Network Interface Controller)之间的传输接口层(Software Transport Interface)被称为 Verbs。OFA(Open Fabric Alliance)提供了 RDMA 传输的一系列 Verbs API，开发了 OFED(Open Fabric Enterprise Distribution)协议栈，支持多种 RDMA 传输层协议。

OFED 向下除了提供 RNIC(实现 RDMA 和 LLP(Lower Layer Protocol))基本的队列消息服务外，向上还提供了 ULP(Upper Layer Protocols)，通过 ULP 上层应用不需直接和 Verbs API 对接，而是借助于 ULP 与应用对接，这样使得常见的应用不需要做修改就可以跑在 RDMA 传输层上。RDMA API (Verbs)，RDMA 的两种 Verbs：

6.1 内存 Verbs（Memory Verbs）

也叫 One-Sided RDMA，包括：RDMA reads，RDMA writes，RDMA atomic。这种模式下的 RDMA 访问完全不需要远端机的任何确认。

6.2 消息 Verbs（Messaging Verbs）

也叫 Two-Sided RDMA，包括：RDMA send，RDMA receive。这种模式下的 RDMA 访问需要远端机 CPU 的参与。

第7章 RDMA 传输分类方式

- reliable 和 unreliable：其中，reliable 方式中 NIC 使用确认形式来保证消息的按顺序传递。而 unreliable 不会进行确认。
- connected 和 unconnected（也叫 datagram）：connected 的传输需要队列对（Queue Pairs）之间的一对一连接，而 unconnected 的方式是一个 QP 可以与多个 QP 进行通信。
- RDMA 不同传输方式支持的 Verbs 类型：如下图（RC 指 reliable connected 传输方式，UC 指 unreliable connected 传输方式，UD 指 unreliable datagram。没有 reliable datagram 的传输方式）

	SEND/RECV	WRITE	READ/ATOMIC
RC	✓	✓	✓
UC	✓	✓	✗
UD	✓	✗	✗

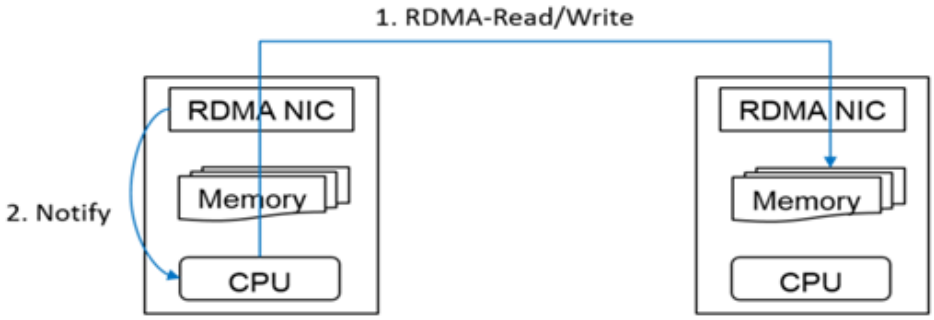
Table 1: Verbs supported by each transport type. RC, UC, and UD stand for Reliable Connected, Unreliable Connected, and Unreliable Datagram, respectively.

原语	RC	UC	UD
SEND/RECEIVE	√	√	√
WRITE	√	√	×
READ	√	×	×

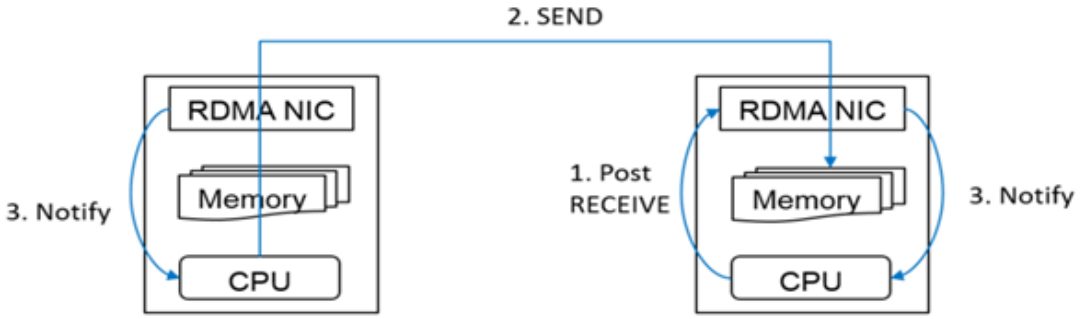
在 Infiniband/RDMA 的模型中，核心是如何实现应用之间最简单、高效和直接的通信。RDMA 提供了基于消息队列的点对点通信，每个应用都可以直接获取自己的消息，无需操作系统和协议栈的介入。

7.1 RDMA 原语

RDMA 提供 RDMA 原语（RDMA verbs）供开发者编程，原语分为两大类：单边（one-side）原语和双边（two-side）原语，最常用的为 RDMA-Read、RDMA-Write、SEND 和 RECEIVE 四个。



单边RDMA原语



双边RDMA原语

单边原语：RDMA-Read & RDMA-Write 被称作单边原语。本端主机上的应用通过单边原语访问远端主机的内存时，远端主机的 CPU 完全不参与流程（被旁路），只有本端主机的 CPU 参与，即只有一边的 CPU 在工作。单边原语是 RDMA 原语中最具有吸引力的原语，可以提供最低的延迟和最高的吞吐。通常所说的 RDMA 都是指单边 RDMA 原语。

双边原语：SEND & RECEIVE 被称作双边原语。双边原语支持数据在传输的过程中完全旁路 CPU，但却要求参与数据传输的本端主机 CPU 和远端主机 CPU 都要工作，即双边的 CPU 都未被旁路。更具体地来说，如果本端主机想要通过 SEND 将数据传输到远端主机的内存中，远端主机必须先要调用 RECEIVE，否则本地调用 SEND 就会失败。

双边原语的工作模式类似于传统的 socket 编程，两端的 CPU 都要参与工作，整体性能要略低于单边原语。

7.2 RDMA 队列对(QP)

RDMA 网卡在处理单边和双边原语时,需要使用队列对(Queue Pair)。一个队列对 QP 包含两个队列，一个发送队列，一个接收队列，都由 RDMA 网卡负责维护。当调用 RDMA-Read、RDMA-Write 和 SEND 时，原语被发送到发送队列，并在发送队列里被发送到远端主机的网卡；当调用 RECEIVE 时，原语被发送到接收队列，并在接收队列里接收来自远端主机的 SEND 请求。

7.3 RDMA 完成事件

每个队列对都对应一个完成事件队列（Completion Queue）。当队列对中的某个队列的原语被操作完成时，例如，发送队列中的发送原语发送到远程机器,或者接收队列中的接收原语接收到远程机器的 SEND 原语，那么 RDMA 网卡会对应产生一个完成事件（Completion Event），并将该事件存入到完成事件队列中。开发者可以通过相关的接口来获取完成事件队列中的完成事件,以判断其之前调用的 RDMA 原语操作的完成状态。但是，如果每个 RDMA 原语操作完成后，都对应产生一个完成事件，势必会增加 RDMA 网卡的负担。RDMA 网卡提供了一种称为 Selective Signaling 的机制来对其进行优化：开发者通过 Selective Signaling 来告诉 RDMA 网卡有多少 RDMA 原语操作不需

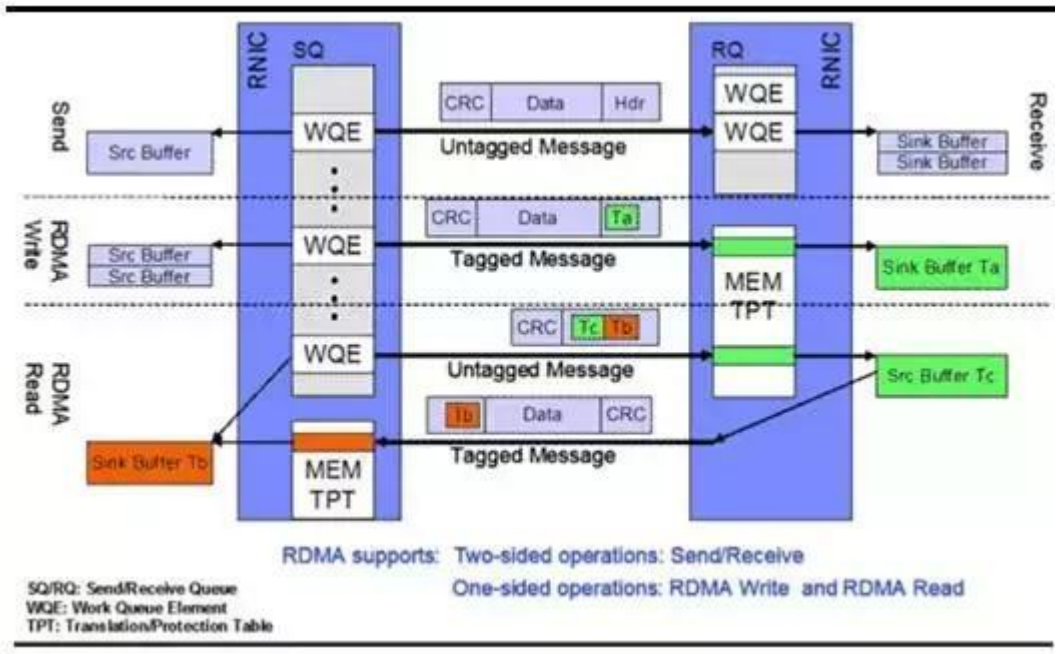
要对应产生完成事件（假设这个数值为 N），那么 RDMA 网卡会对发送队列中完成的连续 N 个 RDMA 原语操作不产生完成事件，而在完成第 N+1 个 RDMA 原语操作时才会产生一个完成事件。这种机制可以大幅提升发送 RDMA-Read、RDMA-Write 和 SEND 原语操作的性能，进而提升系统的整体性能。需要注意的是，Selective Signaling 机制不能用于接收队列，因此对 RECEIVE 原语操作不起作用。

7.4 RDMA 传输类型

RDMA 的传输类型分为两大类：连接模式和非连接模式，前者支持可靠连接（Reliable Connection，RC）和非可靠连接（Unreliable Connection，UC）两种传输类型，后者支持不可靠报文（Unreliable Datagram，UD）传输类型。假设主机 A 和主机 B 通过连接模式进行通信，那么 A 和 B 之间的每一个连接都需要在 A 上对应一个唯一的队列对 QP，在 B 上也对应一个唯一的队列对，这两个队列对 QP 通过该连接绑定在一起。假设主机 A 和主机 B、C 通过非连接模式进行通信，那么 A 机只需要一个队列对就可以和主机 B、C 建立任意数目的连接。不同传输类型支持的原语类型如下图所示。

消息服务建立在通信双方**本端和远端应用之间创建的 Channel-IO 连接之上**。当应用需要通信时，就会创建一条 Channel 连接，每条 Channel 的首尾端点是两对 **Queue Pairs(QP)，每对 QP 由 Send Queue(SQ)和 Receive Queue(RQ)构成**，这些队列中管理着各种类型的消息。QP 会被映射到应用的虚拟地址空间，使得应用直接

通过它访问 RNIC 网卡。除了 QP 描述的两种基本队列之外，RDMA 还提供一种队列 Complete Queue(CQ)，CQ 用来知会用户 WQ 上的消息已经被处理完。



RDMA 提供了一套软件传输接口，方便用户创建传输请求 Work Request(WR)，WR 中描述了应用希望传输到 Channel 对端的消息内容，WR 通知 QP 中的某个队列 Work Queue(WQ)。在 WQ 中，用户的 WR 被转化为 Work Queue Ellement(WQE)的格式，等待 RNIC 的异步调度解析，并从 WQE 指向的 Buffer 中拿到真正的消息发送到 Channel 对端。

RDMA 中 SEND/RECEIVE 是双边操作，即必须要远端的应用感知参与才能完成收发。READ 和 WRITE 是单边操作，只需要本端明确信息的源和目的地址，远端应用不必感知此次通信，数据的读或写都

通过 RDMA 在 RNIC 与应用 Buffer 之间完成，再由远端 RNIC 封装成消息返回到本端。在实际中，**SEND /RECEIVE 多用于连接控制类报文，而数据报文多是通过 READ/WRITE 来完成的**。

7.5 RDMA 双边操作解析

对于双边操作为例，主机 A 向主机 B(下面简称 A、B)发送数据的流程如下

- 1. 首先，A 和 B 都要创建并初始化好各自的 QP，CQ
- 2. A 和 B 分别向自己的 WQ 中注册 WQE，对于 A，WQ=SQ，WQE 描述指向一个等到被发送的数据；对于 B，WQ=RQ，WQE 描述指向一块用于存储数据的 Buffer。
- 3. A 的 RNIC 异步调度轮到 A 的 WQE，解析到这是一个 SEND 消息，从 Buffer 中直接向 B 发出数据。数据流到达 B 的 RNIC 后，B 的 WQE 被消耗，并把数据直接存储到 WQE 指向的存储位置。
- 4. AB 通信完成后，A 的 CQ 中会产生一个完成消息 CQE 表示发送完成。与此同时，B 的 CQ 中也会产生一个完成消息表示接收完成。每个 WQ 中 WQE 的处理完成都会产生一个 CQE。

双边操作与传统网络的底层 Buffer Pool 类似，收发双方的参与过程并无差别，区别在零拷贝、Kernel Bypass，实际上对于 RDMA，这是一种复杂的消息传输模式，多用于传输短的控制消息。

7.6 RDMA 单边操作解析

对于单边操作，以存储网络环境下的存储为例(A 作为文件系统，B 作为存储介质)，数据的流程如下

- 1. 首先 A、B 建立连接，QP 已经创建并且初始化。
- 2. 数据被存档在 A 的 buffer 地址 VA，注意 VA 应该提前注册到 A 的 RNIC,并拿到返回的 local key,相当于 RDMA 操作这块 buffer 的权限。
- 3. A 把数据地址 VA，key 封装到专用的报文传送到 B，这相当于 A 把数据 buffer 的操作权交给了 B。同时 A 在它的 WQ 中注册进一个 WR，以用于接收数据传输的 B 返回的状态。
- 4. B 在收到 A 的送过来的数据 VA 和 R_key 后，RNIC 会把它们连同存储地址 VB 到封装 RDMA READ，这个过程 A、B 两端不需要任何软件参与，就可以将 A 的数据存储到 B 的 VB 虚拟地址。
- 5. B 在存储完成后，会向 A 返回整个数据传输的状态信息。

单边操作传输方式是 RDMA 与传统网络传输的最大不同，只需提供直接访问远程的虚拟地址，无须远程应用的参与其中，这种方式适用于批量数据传输。

7.7 RDMA 技术简单总结

- Infiniband 的成功取决于两个因素，一是主机侧采用 RDMA 技术，可以把主机内数据处理的时延从几十微秒降低到几微秒，同时不占用

CPU；二是 InfiniBand 网络的采用高带宽(40G/56G)、低时延(几百纳秒)和无丢包特性

- 随着以太网的发展，也具备高带宽和无丢包能力，在时延方面也能接近 InfiniBand 交换机的性能，所以 RDMA over Ethernet(RoCE)成为必然，且 RoCE 组网成本更低。未来 RoCE、iWARP 和 Infiniband 等基于 RDMA 技术产品都会得到长足的发展。
- RDMA 相对于传统以太网的优势，以太网的网卡在硬件层面是不提供可靠传输功能的，因此，需要设计上层的软件协议（如经典的 TCP/IP 协议）来保证数据传输的顺序性和可靠性。而 RDMA 网卡（此处 RDMA 网卡指 InfiniBand 提供的网卡或支持 RoCE 的网卡）通过硬件的重传机制，在硬件层面保证了传输的可靠性。另外，在参与通信的两个机器中，RDMA 网卡是支持完全内核旁路的，而以太网卡却需要数据从用户态拷贝到内核态。因此，RDMA 的延迟要远远好于以太网。通常情况下，RDMA 的延迟只有 1 到 3 微秒，而以太网的延迟却达到 10 到几十微秒。
- 本文介绍的 RDMA 特点是利用 RDMA 优化系统的重要基础，它们是任何 RDMA 优化策略所必须要参考的最重要因素之一（另一个最重要因素是内存的管理）。关于这些特点的更多细节，以及它们的不同使用方式会对系统带来什么样的性能提升，笔者会在后续的 wiki 中进行进一步分析。

第8章　InfiniBand 技术和协议架构分析

Infiniband 开放标准技术简化并加速了服务器之间的连接,同时支持服务器与远程存储和网络设备的连接。

8.1　InfiniBand 技术的发展

1999 年开始起草规格及标准规范，2000 年正式发表，但发展速度不及 Rapid I/O、PCI-X、PCI-E 和 FC，加上 Ethernet 从 1Gbps 进展至 10Gbps。所以直到 2005 年之后,InfiniBand Architecture(IBA)才在集群式超级计算机上广泛应用。全球 Top 500 大效能的超级计算机中有相当多套系统都使用上 IBA。

随着越来越多的大厂商正在加入或者重返到它的阵营中来，包括 Cisco、IBM、HP、Sun、NEC、Intel、LSI 等。InfiniBand 已经成为目前主流的高性能计算机互连技术之一。为了满足 HPC、企业数

据中心和云计算环境中的高 I/O 吞吐需求，新一代高速率 56Gbps 的 FDR (Fourteen Data Rate) 和 EDR InfiniBand 技术已经出现。

8.2 InfiniBand 技术的优势

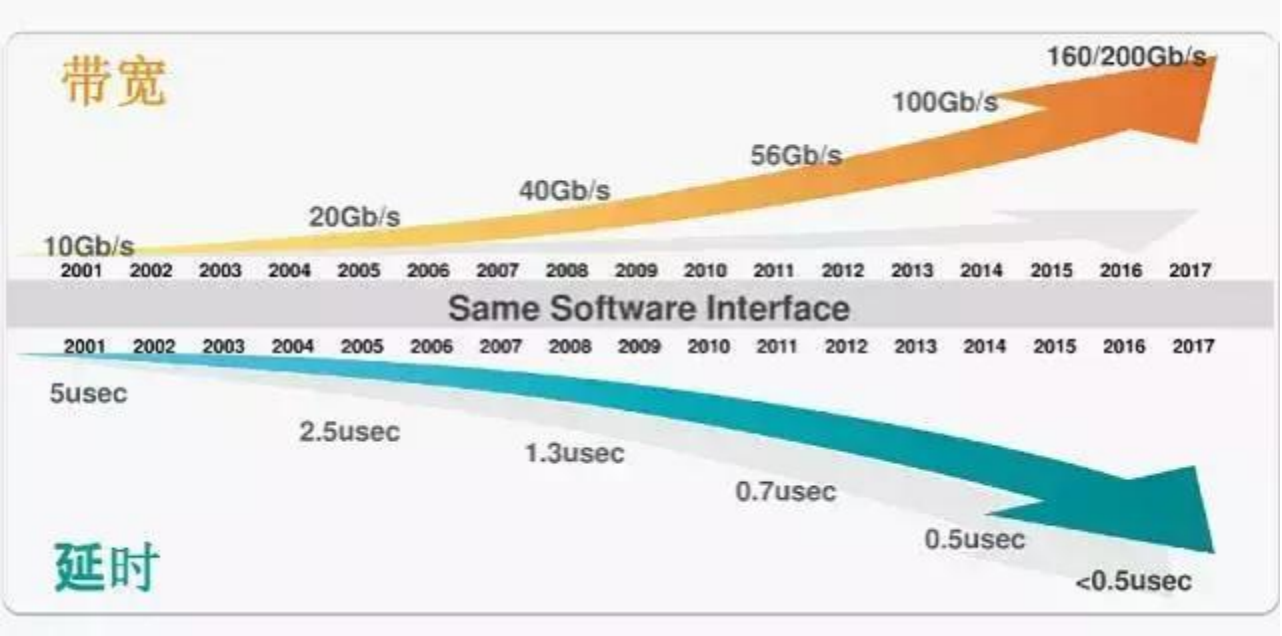
Infiniband 大量用于 FC/IP SAN、NAS 和服务器之间的连接,作为 iSCSI RDMA 的存储协议 iSER 已被 IETF 标准化。目前 EMC 全系产品已经切换到 Infiniband 组网,IBM/TMS 的 FlashSystem 系列,IBM 的存储系统 XIV Gen3,DDN 的 SFA 系列都采用 Infiniband 网络。

相比 FC 的优势主要体现在性能是 FC 的 3.5 倍,Infiniband 交换机的延迟是 FC 交换机的 1/10，支持 SAN 和 NAS。

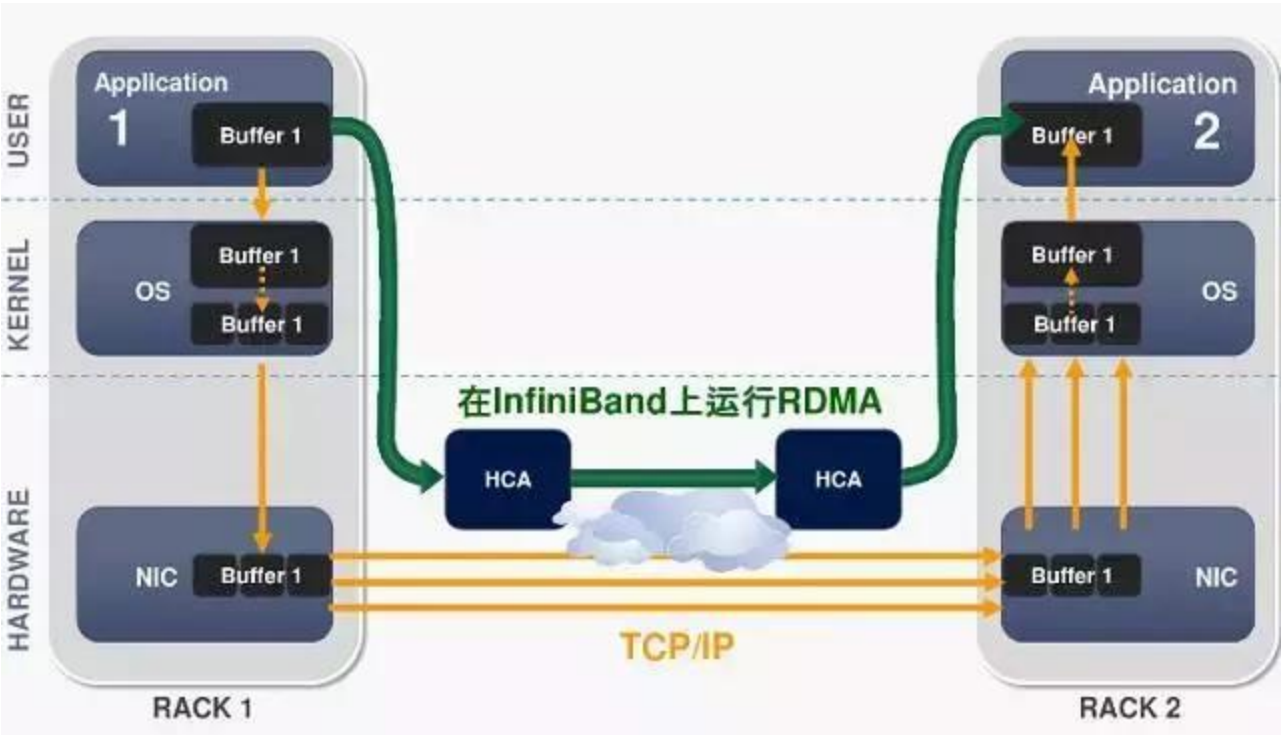
存储系统已不能满足于传统的 FC SAN 所提供的服务器与裸存储的网络连接架构。HP SFS 和 IBM GPFS 是在 Infiniband fabric 连接起来的服务器和 iSER Infiniband 存储构建的并行文件系统，完全突破系统的性能瓶颈。

Infiniband 采用 PCI 串行高速带宽链接,从 SDR、DDR、QDR、FDR 到 EDR HCA 连接，可以做到 1 微妙、甚至纳米级别极低的时延，基于链路层的流控机制实现先进的拥塞控制。

InfiniBand 采用虚通道(VL 即 Virtual Lanes)方式来实现 QoS，虚通道是一些共享一条物理链接的相互分立的逻辑通信链路，每条物理链接可支持多达 15 条的标准虚通道和一条管理通道(VL15)。



RDMA 技术实现内核旁路，可以提供远程节点间 RDMA 读写访问，完全卸载 CPU 工作负载，基于硬件传出协议实现可靠传输和更高性能。



相比 TCP/IP 网络协议，IB 使用基于信任的、流控制的机制来确保连接的完整性，数据包极少丢失，接受方在数据传输完毕之后，返回信号来标示缓存空间的可用性，所以 IB 协议消除了由于原数据包丢失而带来的重发延迟，从而提升了效率和整体性能。

TCP/IP 具有转发损失的数据包的能力，但是由于要不断地确认与重发,基于这些协议的通信也会因此变慢,极大地影响了性能。

8.3 InfiniBand 基本概念

IB 是以通道为基础的双向、串行式传输，在连接拓朴中是采用交换、切换式结构(Switched Fabric)，在线路不够长时可用 IBA 中继器(Repeater)进行延伸。每一个 IBA 网络称为子网(Subnet)，每个子网内最高可有 65,536 个节点(Node)，IBA Switch、IBARepeater 仅适用于 Subnet 范畴，若要通跨多个 IBASubnet 就需要用到 IBA 路由器(Router)或 IBA 网关器(Gateway)。

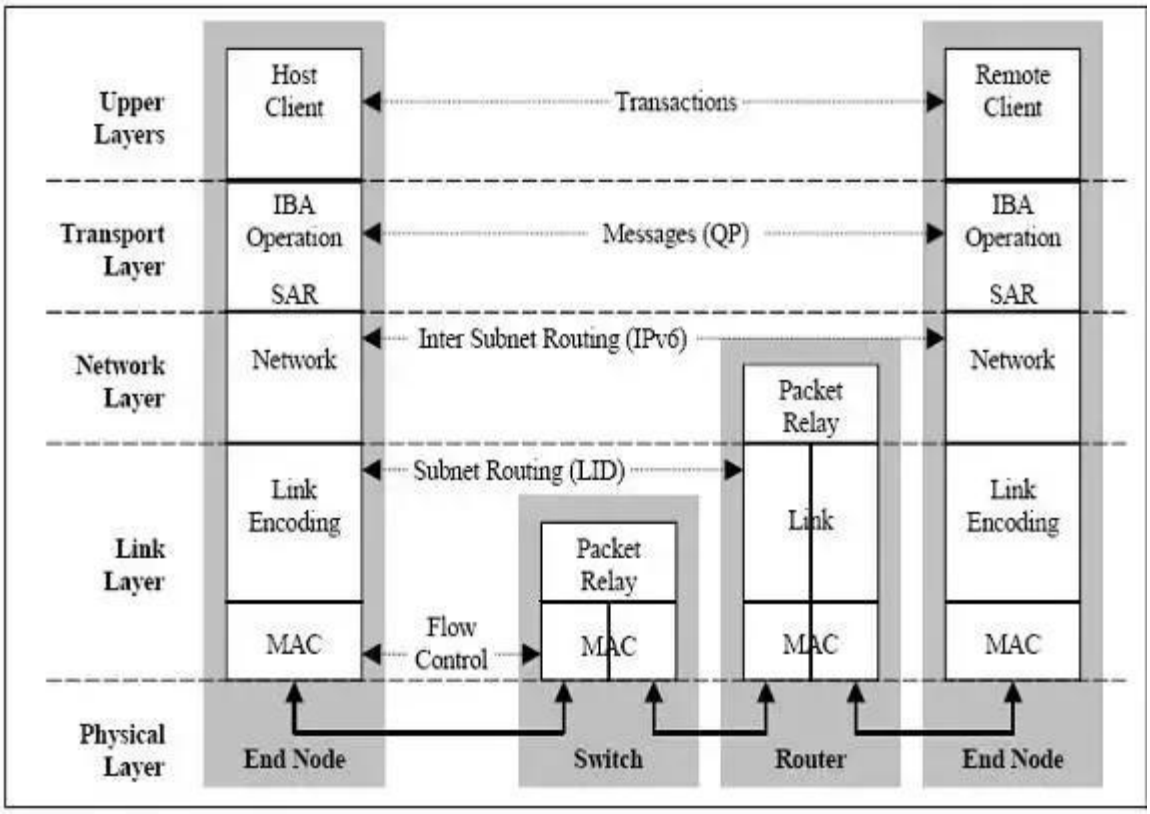
每个节点(Node) 必须透过配接器(Adapter)与 IBA Subnet 连接，节点 CPU、内存要透过 HCA(Host Channel Adapter)连接到子网；节点硬盘、I/O 则要透过 TCA(TargetChannel Adapter)连接到子网，这样的一个拓扑结构就构成了一个完整的 IBA。

IB 的传输方式和介质相当灵活，在设备机内可用印刷电路板的铜质线箔传递(Backplane 背板)，在机外可用铜质缆线或支持更远光纤介质。若用铜箔、铜缆最远可至 17m，而光纤则可至 10km，同

时 IBA 也支持热插拔，及具有自动侦测、自我调适的 Active Cable 活化智能性连接机制。

8.4 InfiniBand 协议简介

InfiniBand 也是一种分层协议(类似 TCP/IP 协议)，每层负责不同的功能，下层为上层服务，不同层次相互独立。 IB 采用 IPv6 的报头格式。其数据包报头包括本地路由标识符 LRH，全局路由标示符 GRH，基本传输标识符 BTH 等。



8.4.1 物理层

物理层定义了电气特性和机械特性，包括光纤和铜媒介的电缆和插座、底板连接器、热交换特性等。定义了背板、电缆、光缆三种物理端口。

并定义了用于形成帧的符号(包的开始和结束)、数据符号(DataSymbols)、和数据包直接的填充(Idles)。详细说明了构建有效包的信令协议，如码元编码、成帧标志排列、开始和结束定界符间的无效或非数据符号、非奇偶性错误、同步方法等。

8.4.2 链路层

链路层描述了数据包的格式和数据包操作的协议，如流量控制和子网内数据包的路由。链路层有链路管理数据包和数据包两种类型的数据包。

8.4.3 网络层

网络层是子网间转发数据包的协议，类似于 IP 网络中的网络层。实现子网间的数据路由，数据在子网内传输时不需网络层的参与。

数据包中包含全局路由头 GRH，用于子网间数据包路由转发。全局路由头部指明了使用 IPv6 地址格式的全局标识符(GID)的源端口和目的端口，路由器基于 GRH 进行数据包转发。GRH 采用 IPv6

报头格式。GID 由每个子网唯一的子网 标示符和端口 GUID 捆绑而成。

8.4.4 传输层

传输层负责报文的分发、通道多路复用、基本传输服务和处理报文分段的发送、接收和重组。传输层的功能是将数据包传送到各个指定的队列(QP)中，并指示队列如何处理该数据包。当消息的数据路径负载大于路径的最大传输单元(MTU)时，传输层负责将消息分割成多个数据包。

接收端的队列负责将数据重组到指定的数据缓冲区中。除了原始数据报外，所有的数据包都包含 BTH，BTH 指定目的队列并指明操作类型、数据包序列号和分区信息。

8.4.5 上层协议

InfiniBand 为不同类型的用户提供了不同的上层协议，并为某些管理功能定义了消息和协议。InfiniBand 主要支持 SDP、SRP、iSER、RDS、IPoIB 和 uDAPL 等上层协议。

- SDP(SocketsDirect Protocol)是 InfiniBand Trade Association (IBTA)制定的基于 infiniband 的一种协议，它允许用户已有的使用 TCP/IP 协议的程序运行在高速的 infiniband 之上。
- SRP(SCSIRDMA Protocol)是 InfiniBand 中的一种通信协议，在 InfiniBand 中将 SCSI 命令进行打包，允许 SCSI 命令通过 RDMA(远程

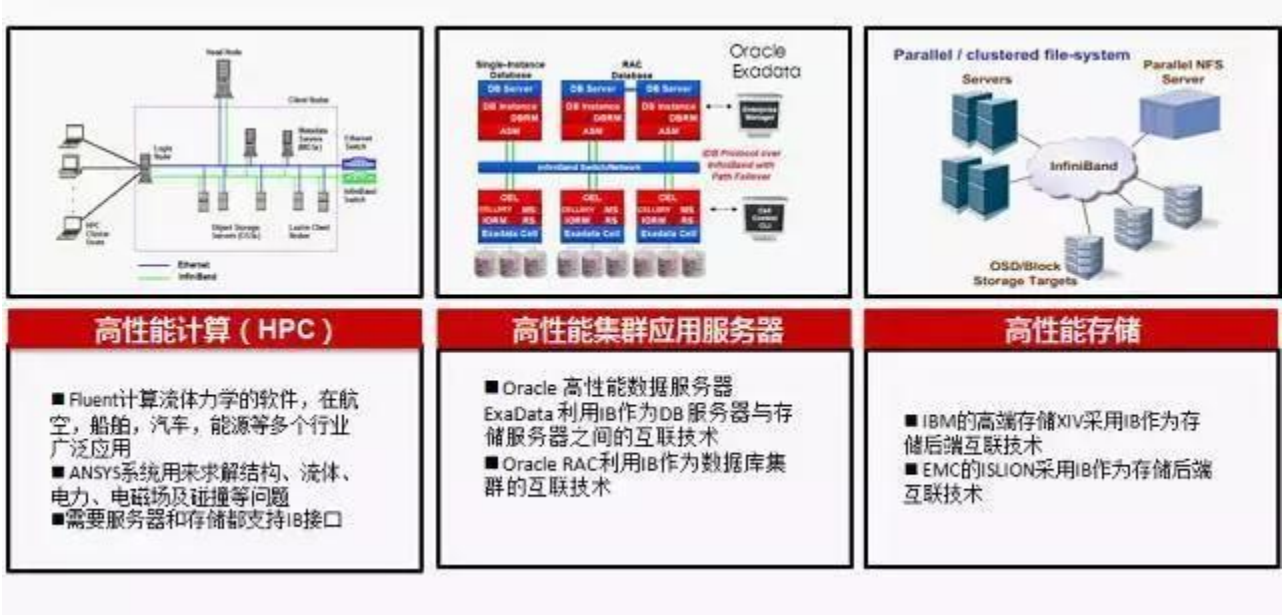
直接内存访问)在不同的系统之间进行通信，实现存储设备共享和RDMA 通信服务。

- iSER(iSCSIRDMA Protocol)类似于 SRP(SCSI RDMA protocol)协议，是 IB SAN 的一种协议 ，其主要作用是把 iSCSI 协议的命令和数据通过 RDMA 的方式跑到例如 Infiniband 这种网络上，作为 iSCSI RDMA 的存储协议 iSER 已被 IETF 所标准化。
- RDS(ReliableDatagram Sockets)协议与 UDP 类似，设计用于在 Infiniband 上使用套接字来发送和接收数据。实际是由 Oracle 公司研发的运行在 infiniband 之上，直接基于 IPC 的协议。
- IPoIB(IP-over-IB)是为了实现 INFINIBAND 网络与 TCP/IP 网络兼容而制定的协议，基于 TCP/IP 协议，对于用户应用程序是透明的，并且可以提供更大的带宽，也就是原先使用 TCP/IP 协议栈的应用不需要任何修改就能使用 IPoIB。
- uDAPL(UserDirect Access Programming Library)用户直接访问编程库是标准的 API，通过远程直接内存访问 RDMA 功能的互连（如 InfiniBand）来提高数据中心应用程序数据消息传送性能、伸缩性和可靠性。

8.5 IB 应用场景

Infiniband 灵活支持直连及交换机多种组网方式，主要用于 HPC 高性能计算场景，大型数据中心高性能存储等场景，HPC 应用

的共同诉求是低时延(<10 微秒)、低 CPU 占有率(<10%)和高带宽(主流 56 或 100Gbps)



一方面 Infiniband 在主机侧采用 RDMA 技术释放 CPU 负载，可以把主机内数据处理的时延从几十微秒降低到 1 微秒；另一方面 InfiniBand 网络的高带宽(40G、56G 和 100G)、低时延(几百纳秒)和无丢包特性吸取了 FC 网络的可靠性和以太网的灵活扩展能力。

第9章　InfiniBand 主流厂商和产品分析

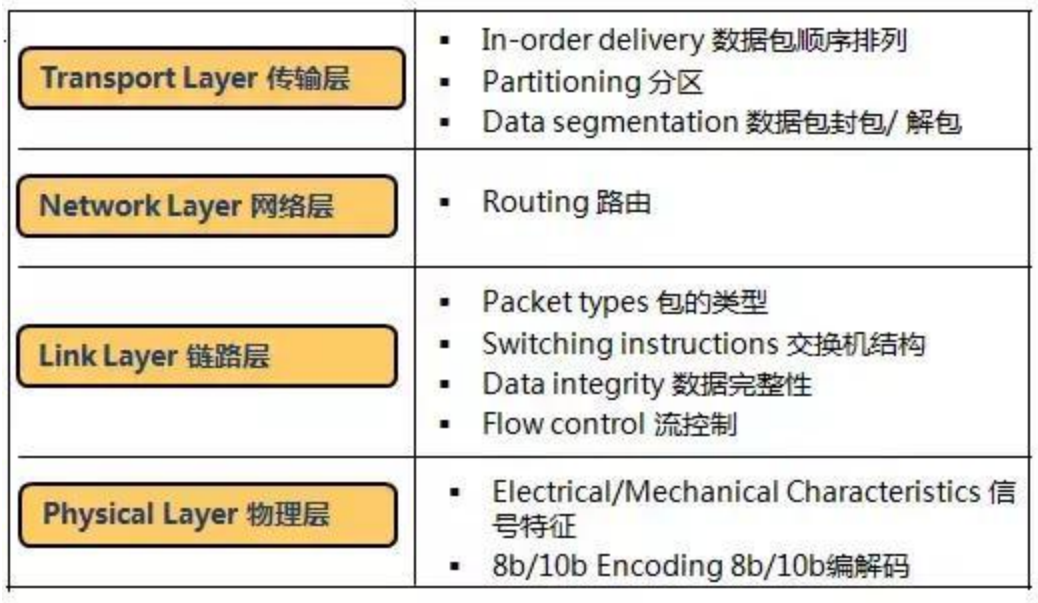
Mellanox 成立于 1999 年，总部设在美国加州和以色列，Mellanox 公司是服务器和存储端到端连接 InfiniBand 解决方案的领先供应商。2010 年底 Mellanox 完成了对著名 Infiniband 交换机厂

商 Voltaire 公司的收购工作，使得 Mellanox 在 HPC、云计算、数据中心、企业计算及存储市场上获得了更为全面的能力。

还有一家 InfiniBand 技术厂商就是 Intel，Intel 拿出 1.25 亿美元收购 QLogic 的 InfiniBand 交换机和适配器产品线发力于高性能计算领域，但今天我们重点讨论 Mellanox 的产品、技术和趋势。

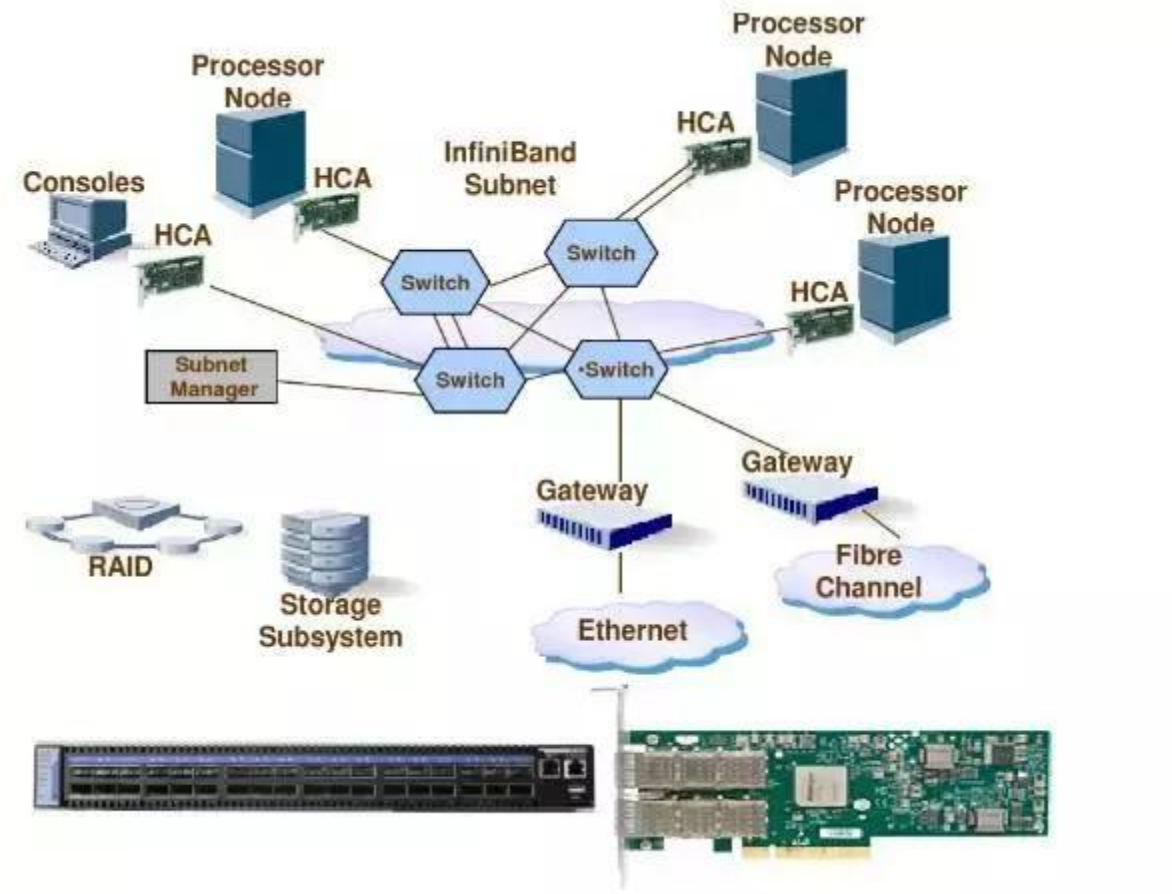
9.1 InfiniBand 网络和拓扑

InfiniBand 结构基于信道的串口替代共用总线，从而使 I/O 子系统和 CPU/内存分离。所有系统和节点可通过信道适配器逻辑连接到该结构，它们可以是主机、适配器(HCA)或目标适配器(TCA)，还包括 InfiniBand 交换机和路由器扩展，从而满足不断增长的需求。



InfiniBand 也是一种分层协议(类似 TCP/IP 协议)，每层负责不同的功能，下层为上层服务，不同层次相互独立，每一层提供

相应功能。InfiniBand 协议可满足各种不同的需求，包括组播、分区、IP 兼容性、流控制和速率控制等。

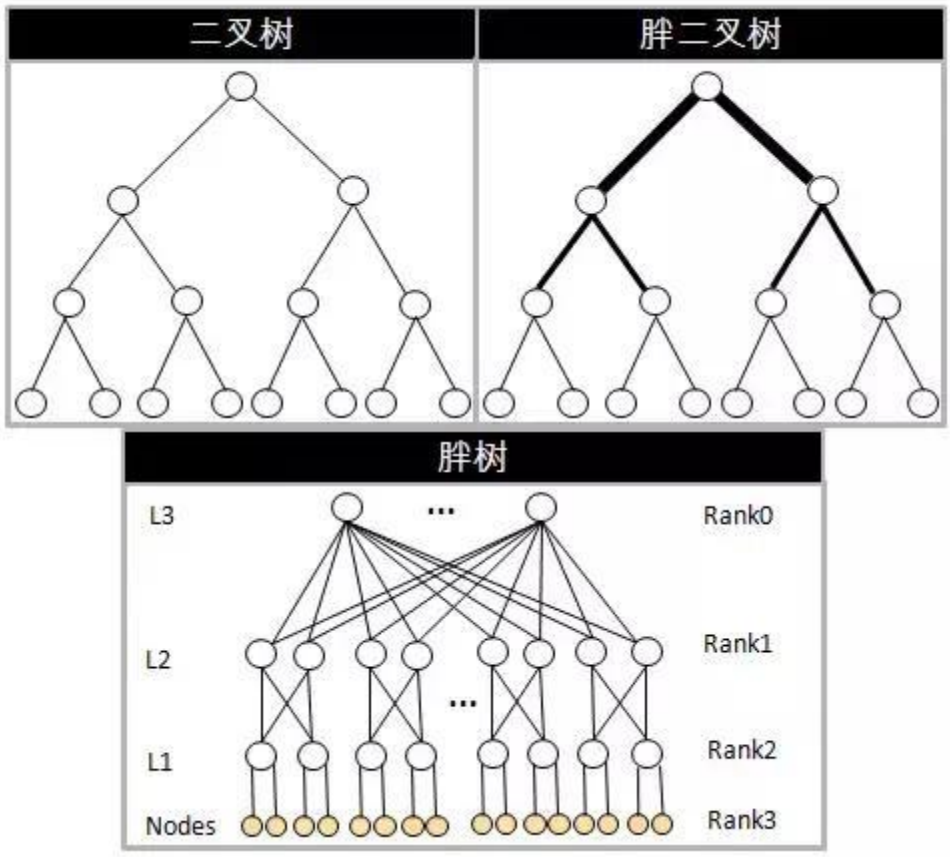


InfiniBand 网络路由算法包括最**短路径算法、基于 Min Hop 的 UPDN 算法和基于 Fat Tree 组网 FatTree 算法**。

算法在一定程度上也决定了 InfiniBand 网络拓扑结构，尤其在高性能计算、大型集群系统，必须要考虑网络之间的拓扑结构，网络上行和下行链路阻塞情况也决定着整个网络性能。由于树形拓扑结构具备清晰、易构建和管理的有点，故而胖树网络拓扑结构常常被

采用，以便能够发挥出 InfiniBand 网络优势，也通常应用在无阻塞或阻塞率很低的应用场景，所以我们下面我们重点讨论下。

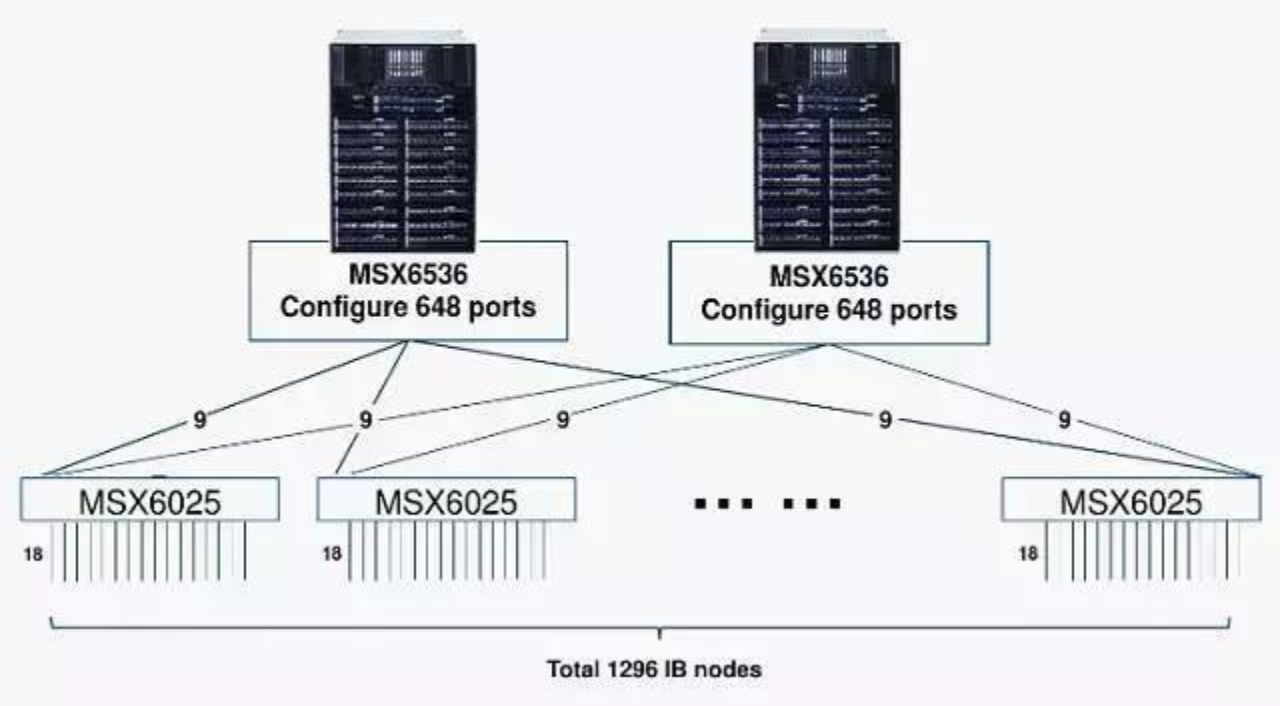
在传统的三层组网架构中(二层架构也经常用到)，由于接入层节点数量庞大，所以要求汇聚层或核心层的网络带宽和处理能力与之匹配，否则设计出来的网络拓扑结构就会产生一定的阻塞比。



为了解决这一问题，在汇聚层和核心层就要采用胖节点组网(如果采用瘦节点就一定发生阻塞，且三层组网阻塞比二层组网更加严重)，如上图胖二叉树事例，胖节点(Fat Tree)必须提供足够的网络端口和带宽与叶子节点匹配。

采用胖树拓扑网络的结构一般由叶子(Leaf)和主干(Spine)交换机组成，叶子交换机与服务器或存储等信道适配卡相连，分配一部分端口给节点，另一部分端口被接入网络中。在 InfiniBand 网络中 Fat Tree 组网结构具有下面几个特点。

- 连接到同一端 Switch 的端口叫端口组，同一 Rank 级别的 Switch 必须有相同的上行端口组,且根 Rank 没有上行端口组;除了 Leaf Switch，同一 Rank 的 Switch 必须有相同的下行端口组。
- 同一 Rank 的每个上行端口组中端口个数相同;且同一 Rank 的每个下行端口组中端口个数也相同。
- 所有终端节点的 HCA 卡都在同一 Rank 级别上。

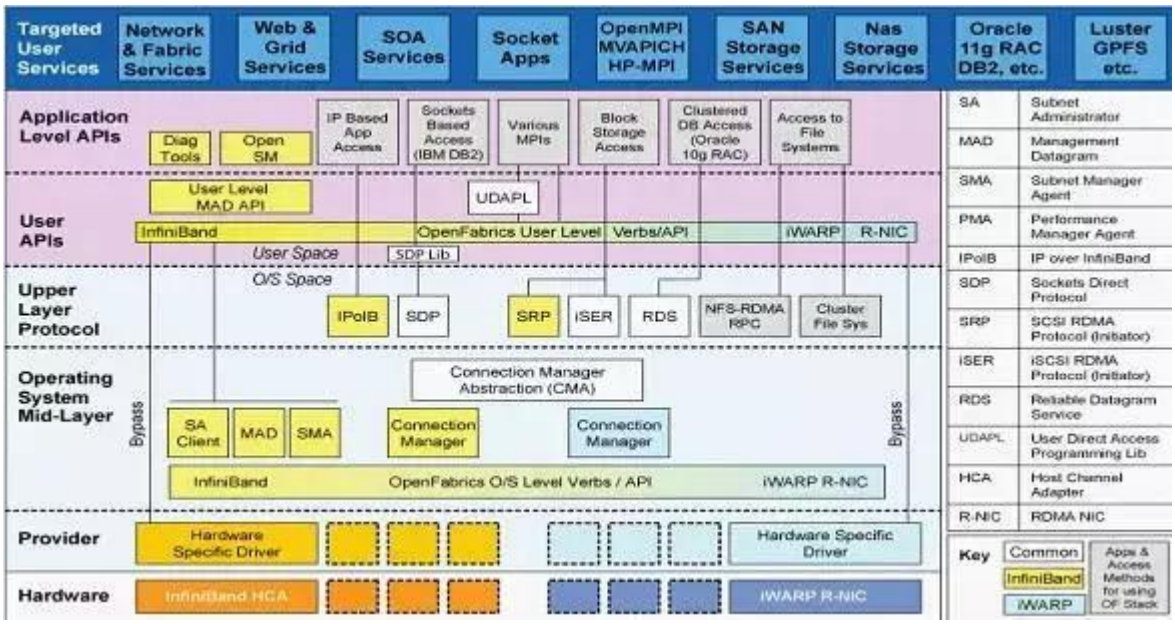


上图是一个采用二层架构的无阻塞 Fat Tree 组网事例，接入层下行提供 1296 个 IB 端口给服务器或存储适配卡，上行也提供适

配器给汇聚层。但从一个接入 IB 交换机来看，上行和下行分别提供 18 个接口实现无阻塞组网。胖树拓扑结构一方面提供费阻塞数据传输，另一方面提供网络冗余增强网络可靠性。

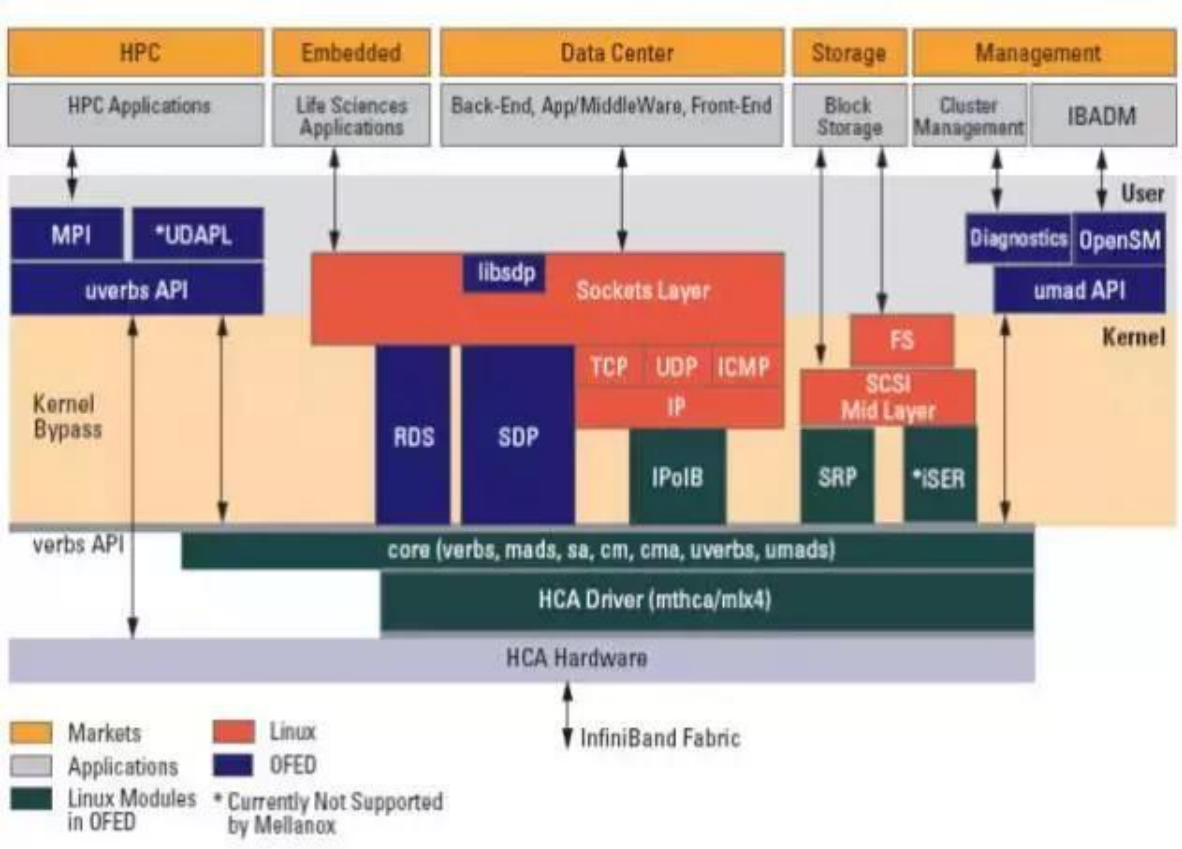
9.2 OpenFabrics Alliance 和 OFED

为服务器和存储集群提供低延迟和高带宽的企业数据中心(EDC)，高性能计算(HPC)和嵌入式应用环境。 Mellanox 所有适配卡与基于 Open Fabrics 的 RDMA 协议和软件兼容。2004 年 OpenFabrics Alliance 成立，该组织致力于促进 RDMA 网络交换技术的发展。2005 年，OpenFabrics Alliance 发布了第一个版本的 OFED(OpenFabrics Enterprise Distribution)。



Mellanox OFED 是一个单一的软件堆栈，包括驱动、中间件、用户接口，以及一系列的标准协议 IPoIB、SDP、SRP、iSER、RDS、DAPL(Direct Access Programming Library)，支持 MPI、Lustre/NFS

over RDMA 等协议，并提供 Verbs 编程接口；Mellanox OFED 由开源 OpenFabrics 组织维护。



如果前面的软件堆栈逻辑图过于复杂，可以参考上面的简明介绍图。Mellanox OFED for Linux (MLNX_OFED_LINUX) 作为 ISO 映像提供，每个 Linux 发行版，包括源代码和二进制 RPM 包、固件、实用程序、安装脚本和文档。

9.3 InfiniBand 网络管理

OpenSM 软件是符合 InfiniBand 的子网管理器(SM)，运行在 Mellanox OFED 软件堆栈进行 IB 网络管理，管理控制流走业务通道，属于带内管理方式。

OpenSM 包括子网管理器、背板管理器和性能管理器三个组件,绑定在交换机内部的必备部件。提供非常完备的管理和监控能力,如设备自动发现、设备管理、Fabric 可视化、智能分析、健康监测等等。

9.4 并行计算集群能力

MPI (Message Passing Interface) 用于并行编程的一个规范,并行编程即使用多个 CPU 来并行计算,提升计算能力。Mellanox OFED for Linux 的 InfiniBand MPI 实现包括 Open MPI 和 OSU MVAPICH。

Open MPI 是基于 Open MPI 项目的开源 MPI-2 实现, OSU MVAPICH 是基于俄亥俄州立大学的 MPI-1 实施。下面列出了一些有用的 MPI 链接。

MPI Standard	http://www-unix.mcs.anl.gov/mpi
Open MPI	http://www.open-mpi.org
MVAPICH MPI	http://nowlab.cse.ohio-state.edu/projects/mpi-iba/
MPI Forum	http://www.mpi-forum.org

RDS (Reliable Datagram Socket)是一种套接字 API，在 sockets over RC or TCP/IP 之间提供可靠的按顺序数据报传送，RDS 适用于 Oracle RAC 11g。

9.5 基于 Socket 网络应用能力

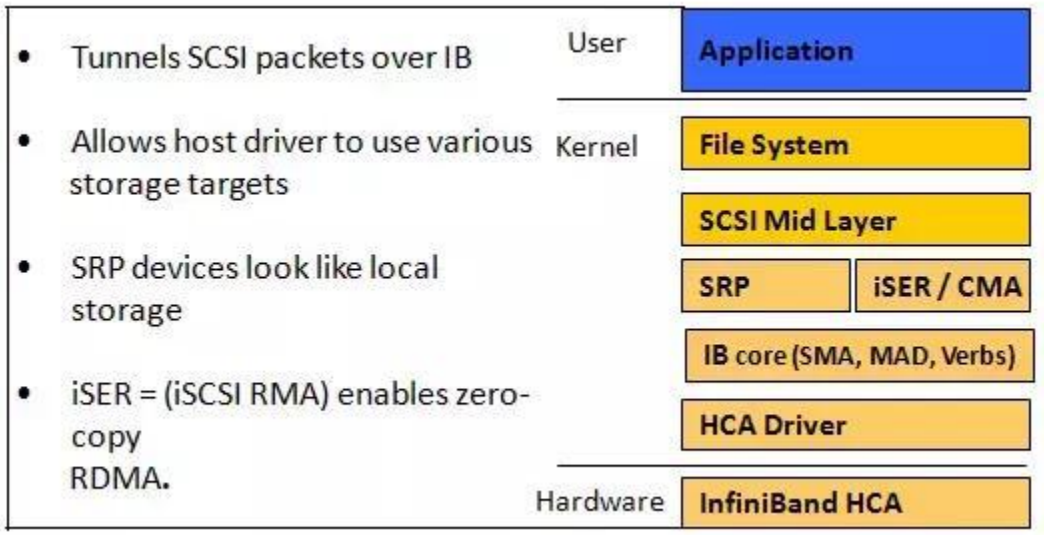
IPoIB/ EoIB (IP/Eth over InfiniBand) 是通过 InfiniBand 实现的网络接口实现，IPoIB 封装 IP 数据报通过 InfiniBand 连接或数据报传输服务。

SDP (Socket Direct Protocol) 是提供 TCP 的 InfiniBand 字节流传输协议流语义，利用 InfiniBand 的高级协议卸载功能，SDP 可以提供更低的延迟更高带宽。

9.6 存储支持能力

支持 iSER (iSCSI Extensions for RDMA)和 NFSoRDMA (NFS over RDMA)，SRP (SCSI RDMA Protocol) 是 InfiniBand 中的一种通信协议，在 InfiniBand 中将 SCSI 命令进行打包，允许 SCSI 命令

通过 RDMA(远程直接内存访问)在不同的系统之间进行通信，实现存储设备共享和 RDMA 通信服务。

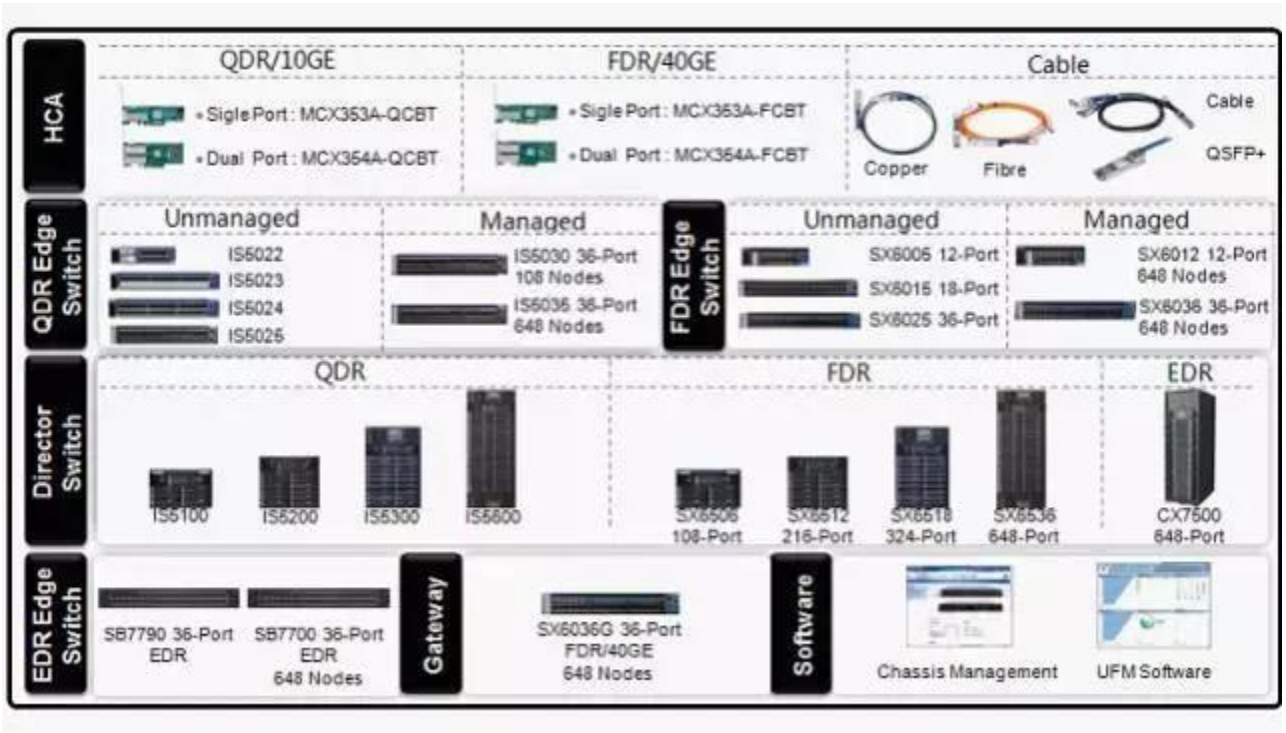


RDMA (Remote Direct Memory Access)技术是为了解决网络传输中服务器端数据处理的延迟而产生的。RDMA 通过网络把数据直接传入计算机的存储区，将数据从一个系统快速移动到远程系统存储器中，而不对操作系统造成任何影响，这样就不需要用到多少计算机的处理功能。它消除了外部存储器复制和文本交换操作，因而释放内存带宽和 CPU 周期用于改进应用系统性能。

9.7 Mellanox 产品介绍

Mellanox 是服务器和存储端到端连接解决方案的领先供应商，一直致力于 InfiniBand 和以太网互联产品的研发工作，也是业界公认的超高速网络典型代表。下面我们重点看看 InfiniBand 和相关产品介绍。

InfiniBand 产品搭配先进的 VPI 技术使得单端口适配业务需求，主要产品包括 VPI 系列网卡、交换机。芯片产品也是保障所有系列产品的可靠基石。种类丰富的线缆是实现高速互联网络的重要保证。除了硬件外，InfiniBand 配套加速软件和统一管理软件丰富整个产品家族。



9.8 Infiniband 交换机

在 IB 网络内提供点到点高速通信;基于 LID 技术将数据从一个端口送到另外一个端口 ，当前单个交换机支持从 18 到 864 节点等规模不等,支持 SDR(10Gbps)、DDR(20Gbps)、QDR(40Gbps)、FDR10（40Gbps）、FDR(56Gbps)等。

从 SwitchX 到 Switch IB，SwitchX 是支持 10、20、40、56 G IB 主流的芯片，下一代芯片 Switch IB 支持 IB EDR 100Gb/s，并且向前兼容，后面还有 SwitchX3 支持 100G 和 IB EDR。

基于ConnectX系列网卡和SwitchX交换机可以实现以太网和IB网络的虚拟协议互联(VPI)，实现链路协议显示或自动适配，一个物理交换机实现多种技术支持。虚拟协议互联支持整机VPI、端口VPI和VPI桥接，整机VPI实现交换机所有端口运行在InfiniBand或以太网模式，端口VPI实现交换机部分端口运行InfiniBand、部分端口运行以太网模式，VPI桥接模式实现InfiniBand和以太网桥接。

Edge Switches

	IS5022	IS5023	IS5024	IS5025	SX6025	SB7790
Ports	8	18	36	36	36	36
Height	1U	1U	1U	1U	1U	1U
Switching Capacity	640Gb/s	1.44Tb/s	2.88Tb/s	2.88Tb/s	4.032Tb/s	7.2Tb/s
Link Speed	40Gb/s	40Gb/s	40Gb/s	40Gb/s	56Gb/s	100Gb/s
Interface Type	QSFP	QSFP	QSFP	QSFP	QSFP+	QSFP28
Management	No	No	No	No	No	No
PSU Redundancy	No	No	No	Yes	Yes	Yes
Fan Redundancy	No	No	No	Yes	Yes	Yes
Integrated Gateway	-	-	-	-	-	-

	SX6005	SX6012	SX6015	SX6018	IS5030	IS5035	4036E	SX6036	SB7700
Ports	12	12	18	18	36	36	34 + 2Eth	36	36
Height	1U	1U	1U	1U	1U	1U	1U	1U	1U
Switching Capacity	1.3 Tb/s	1.3 Tb/s	2.016 Tb/s	2.016Tb/s	2.88Tb/s	2.88Tb/s	2.72Tb/s	4.032Tb/s	7.2Tb/s
Link Speed	56 Gb/s	56 Gb/s	56 Gb/s	56Gb/s	40Gb/s	40Gb/s	40Gb/s	56Gb/s	100Gb/s
Interface Type	QSFP+	QSFP+	QSFP+	QSFP+	QSFP	QSFP	QSFP	QSFP+	QSFP28
Management	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
	-	648	No	648 nodes	108 nodes	648 nodes	648 nodes	648 nodes	2048 nodes
Management Ports	-	1	-	2	1	2	1	2	2
PSU Redundancy	No	Optional	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fan Redundancy	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Integrated Gateway	-	Optional	-	Optional	-	-	Yes	Optional	-

边缘(机架) InfiniBand 交换机系统支持 8 到 36 端口，提供非阻塞 40 到 100Gb 端口，在 1U 的空间可提供 7.2Tb 的带宽，这些边缘交换机是组成中小型费阻塞网络集群 Leaf 节点的理想选择。边缘交换机使用先进的 InfiniBand 交换技术(如自适应路由、拥塞控制和服务质量等)旨在构建最有效的交换矩阵。

Director Switches

	SX6506	SX6512	CS7520	SX6518	SX6536	CS7500
Ports	108	216	216	324	648	648
Height	6U	9U	12U	16U	29U	28U
Switching Capacity	12.12Tb/s	24.24Tb/s	43.2Tb/s	36.36Tb/s	72.52Tb/s	130Tb/s
Link Speed	56Gb/s	56Gb/s	100Gb/s	56Gb/s	56Gb/s	100Gb/s
Interface Type	QSFP+	QSFP+	QSFP28	QSFP+	QSFP+	QSFP28
Management	648 nodes	648 nodes	2048 nodes	648 nodes	648 nodes	2048 nodes
Management HA	Yes	Yes	Yes	Yes	Yes	Yes
Console Cables	Yes	Yes	Yes	Yes	Yes	Yes
Spine Modules	3	6	6	9	18	18
Leaf Modules (Max)	6	12	6	18	36	18
PSU Redundancy	YES (N+N)	YES (N+N)	YES (N+N)	YES (N+N)	YES (N+N)	YES (N+N)
Fan Redundancy	Yes	Yes	Yes	Yes	Yes	Yes

核心 InfiniBand 交换机系统支持 108 至 648 端口，提供全双向 40 至 100Gb 端口， InfiniBand 核心交换机系统提供高密的解决方案，在一个机框内带宽可以 8.4Tb 至 130Tb 之间灵活扩展，可达数千个端口。针对关键任务应用，InfiniBand 核心交换机提供核心级可用性，系统所有部件都采用冗余技术设计。

9.9 InfiniBand 适配器

Inifiniband 的主机信道适配器 HCA(网络接口卡)，通常通过 PCIE 接口与主机连接，插在或集成在服务器内;支持 PCI-E 8X 插槽(双端口和单端口)。提供 Inifiniband 的网络链路接入能力。等同于以太网的 NIC。HCA 包含三代芯片：目前主流的 QDR，FDR 使用的芯片为 ConnectX3，OSCA 使用的也是 ConnectX3

Channel Adapter(CA)
分为Host Channel Adapter(HCA)和Target Channel Adapter(TCA)

Host Channel Adapter(HCA)
主机通道适配器

如：**Mellanox 产品**

Target Channel Adapter(TCA)
目标通道适配器
用于IB交换机、存储系统的IO接口

型 号	速 率	系统说明	接 口
Mellanox MH 系列	10-40Gbps	PCIe 2.0	2个 IB 铜口 / QSHP

目标信道适配器(TCA)提供InfiniBand到I/O设备的连接，绑定在存储或网关设备等外设。

9.10 Infiniband路由器和网关设备

Infiniband 路由器完成不同子网的 infiniband 报文的转发。Mellanox 的 SB7780 是基于 Switch-IB 交换机 ASIC 实现的 InfiniBand 路由器，提供 EDR 100Gb s 端口可以连接不同类型的拓扑。因此，它能够使每个子网拓扑最大化每个应用程序的性能。例如，存储子网可以使用 Fat Tree 拓扑，而计算子网可以使用最适合本地应用程序的环路拓扑。

SX6036G 是采用 Mellanox 第六代 SwitchX 2 InfiniBand 构建的交换机网关设备，提供高性能、低延迟的 56Gb FDR Infiniband 到 40Gb 以太网的网关，支持 InfiniBand 和以太网连接的虚拟协议互连(VPI)技术，VPI 通过一个硬件平台能够在同一机箱上运行 InfiniBand 和以太网网络协议。

9.11 Infiniband 线缆和收发器

Mellanox LinkX 互连产品包括 10、25、40、50 和 100 Gb/s 丰富铜缆、有源光缆以及针对单模光纤和多模光纤应用的收发器。

LinkX 系列提供 200Gb/s 和 400Gb/s 电缆和收发器等关键组件,对于 InfiniBand 互连基础设施来说,让端到端的 200Gb/s 解决方案成为可能。

9.12 InfiniBand 主要构件总结

Host Channel Adapter, HCA 为一个 IB 终端节点（例如一个服务器）连接到 IB 网络中提供了连接点。这类似于以太网卡（NIC），但比 NIC 做更多的事情。HCA 在操作系统的控制下提供了地址翻译机制，允许应用程序直接使用 HCA。相同地利用这种地址翻译机制，HCA 能够在用户态程序下对内存进行使用。应用程序直接使用 虚拟地址进行操作，HCA 能将这些虚拟地址翻译为物理地址，这样就实现了消息的有效传输。

Range Extenders，IB 将传输通路加入到 WAN 中，来实现范围的网络范围的扩展。同时，增加了足够的 buffer credit (BBC)来确保在 WAN 上能够满带宽运行。

Subnet Manager，IB 子网管理器给每个连接到子网中的端口分配一个局部的 ID（LID），并且基于这些 LID，构建了一 个路由表。IB 子网管理器是一个软件定义网络（SDN）的概念，SDN 能够消除网络连接的复杂性，使 创建超大规模的计算和存储体系成为可能。

Switches，IB 交换机与标准的网络中的交换中在概念上是相似的，但它的设计是为了迎合 IB 的性能需求。IB 交 换机在 IB 的链路层实施了流控制，一是为了防止数据包的丢失，二是为了支持拥塞控制和自适应路 由，同时也为了保证良好的服务质量。很多 IB 交换机都包含一个子网管理器。配置一个 IB 网络， 至少需要一个子网管理器。

9.13 InfiniBand 对现有应用和 ULPs 支持

IB 对现有应用的支持和 ULPs（Upper Layer Protocols 上层协议）支持。基于 IP 的应用可以在 IB 网络中运行，需要用到 IP over IB(IPoIB)或者 Ethernet over IB(EoIB)或者 RDS ULPs。

存储型的应用可以利用 iSER,SRP,RDS,NFS,ZFS,SMB 等等。MPI 和网络管理也是支持的 ULPs，但不在本文讨论范围之内。

参考资料

- IBTA Intro to IB for End Users
- Mellanox InfiniBandFAQ_FQ_100.pdf
- Mellanox WP_2007_IB_Software_and_Protocols.pdf

第10章 RDMA over TCP(iWARP)协议和工作原理

随着网络带宽和速度的发展和大数据量数据的迁移的需求，网络带宽增长速度远远高于处理网络流量时所必需的计算节点的能力和对内存带宽的需求，数据中心网络架构已经逐步成为计算和存储技术的发展的瓶颈，迫切需要采用一种更高效的数据通讯架构。

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，所产生严重的延迟来自系统庞大的开销、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，如图 1.1 所示，给服务器的 CPU 和内存造成了沉重负担。特别是面对网络带宽、处理器速度与内存带宽三者的严重"不匹配性"，更造成了网络延迟效应的加剧。处理器速度比内存速度快得越多，等待相应数据的延迟就越多。而且，处理每一数据包时，数据必须在系统内存、处理器缓存和网络控制器缓存之间来回移动，因此延迟并不是一次性的，而是会对系统性能持续产生负面影响。

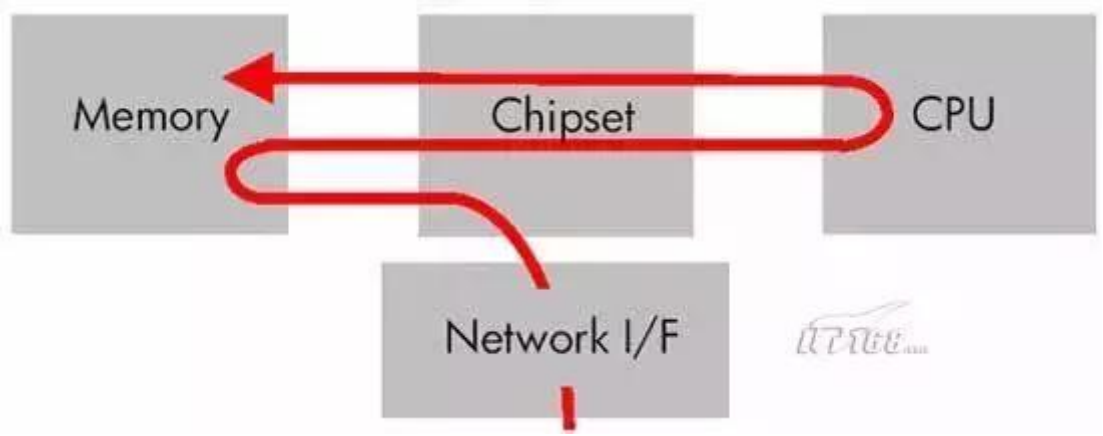


图 1.1 主机接收传统以太网数据的典型数据流示意图

10.1 RDMA 相关简介

这样，以太网的低投入、低运营成本优势就难以体现。为充分发挥万兆位以太网的性能优势，必须解决应用性能问题。系统不能以软件方式持续处理以太网通信；主机 CPU 资源必须释放专注于应用处理。业界最初的解决方案是采用 TCP/IP 负荷减轻引擎(TOE)。TOE 方案能提供系统性能，但协议处理不强；它能使 TCP 通信更快速，但还达不到高性能网络应用的要求。解决这类问题的关键，是要消除主机 CPU 中不必要的频繁数据传输，减少系统间的信息延迟。

RDMA(Remote Direct Memory Access)全名是"远程直接数据存取"，RDMA 让计算机可以直接存取其它计算机的内存，而不需要经过处理器耗时的传输，如图 1.2 所示。RDMA 是一种使一台计算机可以直接将数据通过网络传送到另一台计算机内存中的特性，将数据从一个系统快速移动到远程系统存储器中，而不对操作系统造成任何影响，这项技术通过消除外部存储器复制和文本交换操作，因而能腾出总线

空间和 CPU 周期用于改进应用系统性能，从而减少对带宽和处理器开销的需要，显著降低了时延。

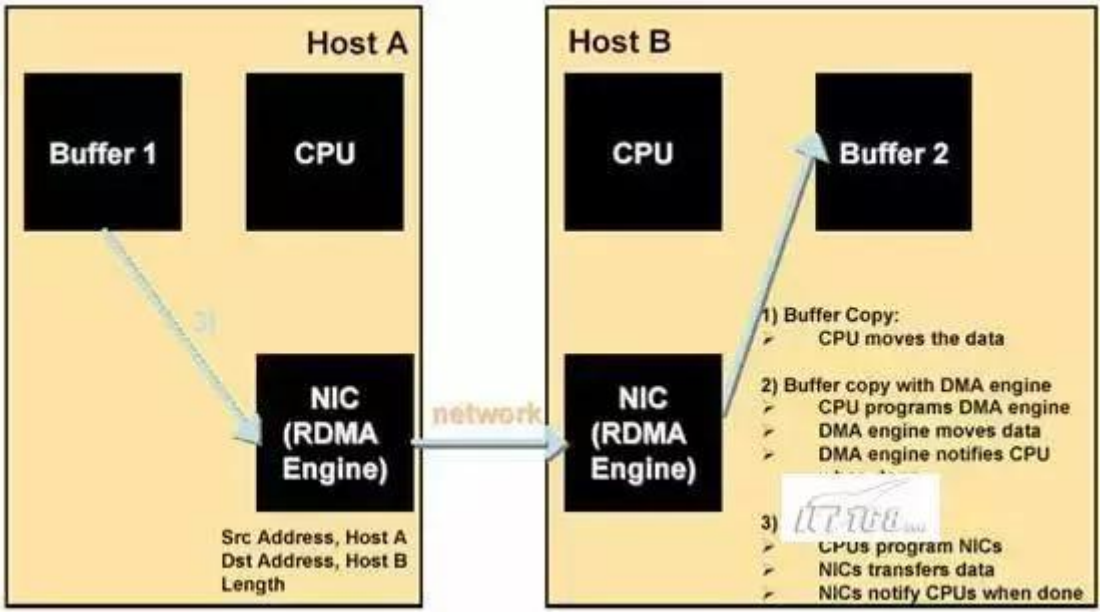


图 1.2 RDMA 数据流传输示意图

RDMA 对以太网来说还是"新生事物"，但以不同形式存在已有十多年时间，它是 Infiniband 技术的基础。产业标准 API 使 RDMA 从技术走向实现成为可能。其中包括用于低时延消息处理、成就高性能计算的 MPI(消息通过接口)，以及 DAPL(直接接入供应库)。后者包括两部分： KDAPL 和 UDAPL，分别用于内核和用户(应用程序)。Linux 支持 KDAPL，其它操作系统将来也有可能支持。RDMA 在高性能计算环境广为采纳，在商务应用领域很少，但如今大多应用程序都能直接支持操作系统，透过操作系统(如 NFS)间接利用 RDMA 技术的优势是完全可能的。

10.2 RDMA 工作原理

RDMA 是一种网卡技术，采用该技术可以使一台计算机直接将信息放入另一台计算机的内存中。通过最小化处理过程的开销和带宽的需求降低时延。RDMA 通过在网卡上将可靠传输协议固化于硬件，以及支持零复制网络技术和内核内存旁路技术这两种途径来达到这一目标。RDMA 模型如图 2.1 所示。

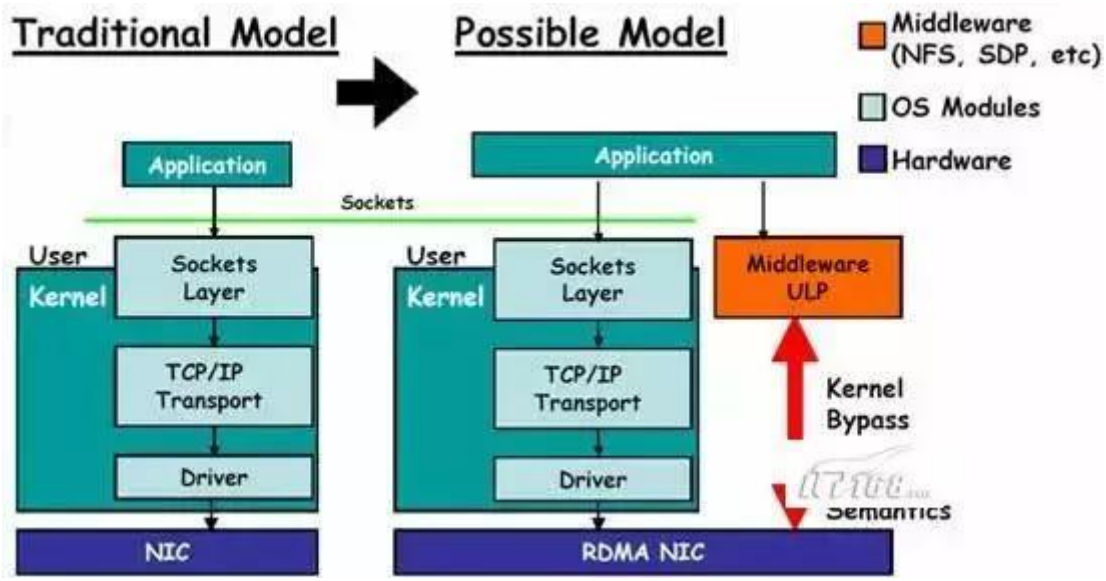


图 2.1 RDMA 模型演变

零复制网络技术使 NIC 可以直接与应用内存相互传输数据，从而消除了在应用内存与内核内存之间复制数据的需要。

内核内存旁路技术使应用程序无需执行内核内存调用就可向网卡发送命令。在不需要任何内核内存参与的条件下，RDMA 请求从用户空间发送到本地 NIC 并通过网络发送给远程 NIC，这就减少了在处理网络传输流时内核内存空间与用户空间之间环境切换的次数。

当一个应用程序执行 RDMA 读/写请求时，系统并不执行数据复制动作，这就减少了处理网络通信时在内核空间和用户空间上下文切换的次数。在不需要任何内核内存参与的条件下，RDMA 请求从运行在用户空间中的应用中发送到本地 NIC(网卡)，然后经过网络传送到远程 NIC。请求完成既可以完全在用户空间中处理(通过轮询用户级完成排列)，或者在应用一直睡眠到请求完成时的情况下通过内核内存处理。

RDMA 操作使应用可以从一个远程应用的内存中读数据或向这个内存写数据。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 在整个 RDMA 操作中并不提供服务，因此没有带来任何负载。通过类型值(键值)的使用，一个应用程序能够在远程应用程序对它进行随机访问的情况下保护它的内存。

发布 RDMA 操作的应用程序必须为它试图访问的远程内存指定正确的类型值，远程应用程序在本地网卡中注册内存时获得这个类型值。发布 RDMA 的应用程序也必须确定远程内存地址和该内存区域的类型值。远程应用程序会将相关信息通知给发布 RDMA 的应用程序，这些信息包括起始虚拟地址、内存大小和该内存区域的类型值。在发布 RDMA 的应用程序能够对该内存区域进行 RDMA 操作之前，远程应用程序应将这些信息通过发送操作传送给发布 RDMA 的应用程序。

10.3 RDMA 操作类型

具备 RNIC（RDMA-aware network interface controller）网卡的设备，不论是目标设备还是源设备的主机处理器都不会涉及到数据传输操作，RNIC 网卡负责产生 RDMA 数据包和接收输入的 RDMA 数据包，从而消除传统操作中多余的内存复制操作。

RDMA 协议提供以下 4 种数据传输操作(RDMA Send 操作、RDMA Write 操作、RDMA Read 操作和 Terminate 操作)，除了 RDMA 读操作不会产生 RDMA 消息，其他操作都会产生一条 RDMA 消息。

10.4 RDMA over TCP 详解

以太网凭借其低投入、后向兼容、易升级、低运营成本优势在目前网络互连领域内占据统治地位，目前主流以太网速率是 100 Mb/s 和 1000 Mb/s，下一代以太网速率将会升级到 10Gb/s。将 RDMA 特性增加到以太网中，将会降低主机处理器利用率，增加以太网升级到 10 Gb/s 的优点，消除由于升级到 10 Gb/s 而引入巨大开销的弊端，允许数据中心在不影响整体性能的前提下拓展机构，为未来扩展需求提供足够的灵活性。

RDMA over TCP 协议将数据直接在两个系统的应用内存之间进行交互，对操作系统内核几乎没有影响，并且不需要临时复制到系统内存的操作，数据流如图 4.1 所示。

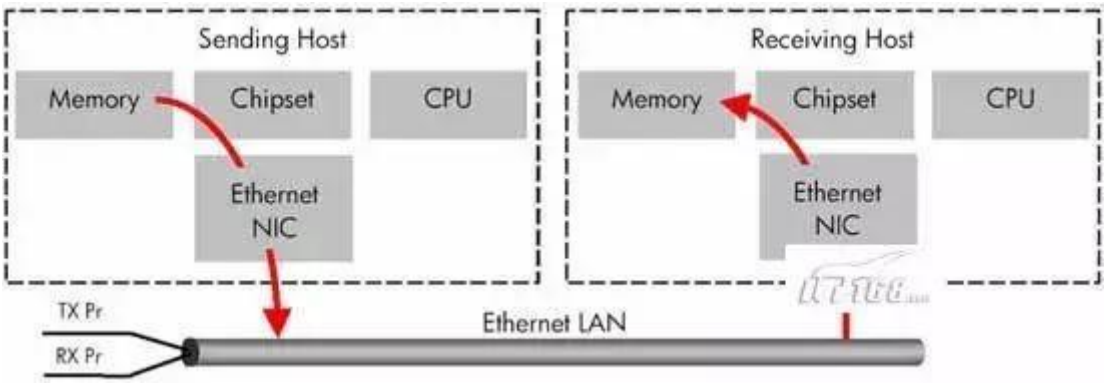


图 4.1 RDMA over TCP (Ethernet)数据流示意图

RDMA over TCP 协议能够工作在标准的基于 TCP/IP 协议的网络，如目前在各个数据中心广泛使用的以太网。注意：RDMA over TCP 并没有指定物理层信息，能够工作在任何使用 TCP/IP 协议的网络上层。RDMA over TCP 允许很多传输类型来共享相同的物理连接，如网络、I/O、文件系统、块存储和处理器之间的消息通讯。

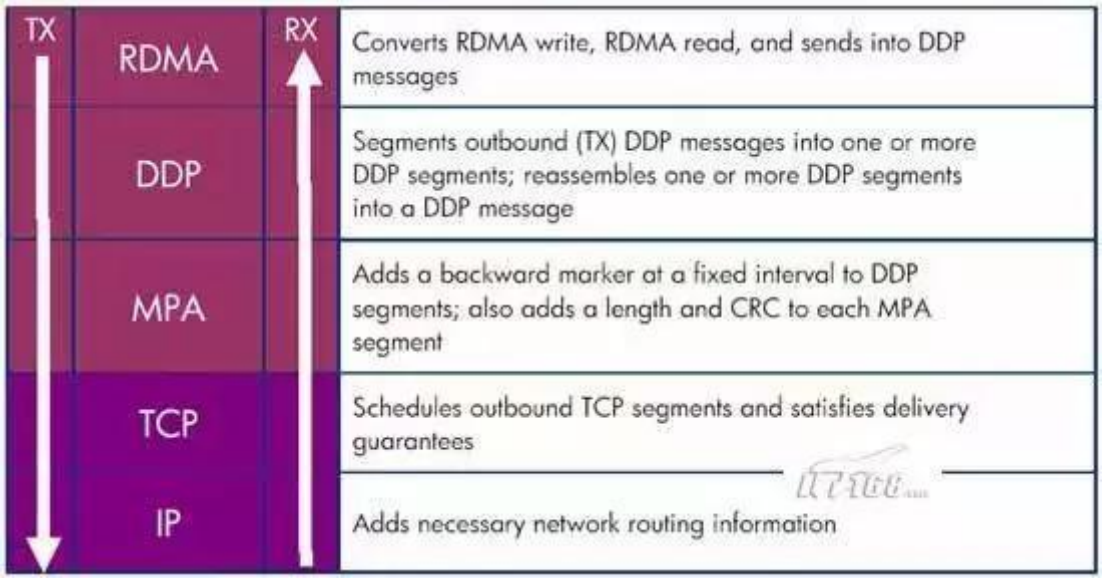


图 4.2 RDMA over TCP (Ethernet)协议栈

图 4.2 是 RDMA over TCP (Ethernet)的协议栈，最上面三层构成 iWARP 协议族，用来保证高速网络的互操作性。

RDMA 层协议负责根据 RDMA 写操作、RDMA 读操作转换成 RDMA 消息，并将 RDMA 消息传向 Direct Data Placement (DDP)层。DDP 层协议负责将过长的 RDMA 消息分段封装成 DDP 数据包继续向下转发到 Marker-based, Protocol-data-unit-Aligned (MPA)层。MPA 层在 DDP 数据段的固定间隔位置增加一个后向标志、长度以及 CRC 校验数据，构成 MPA 数据段。TCP 层负责对 TCP 数据段进行调度，确保发包能够顺利到达目标位置。IP 层则在数据包中增加必要的网络路由数据信息。

DDP 层的 PDU 段的长度是固定的，DDP 层含有一个成帧机制来分段和组合数据包，将过长的 RDMA 消息分段封装为 DDP 消息，处理过程如图 4.3 所示。

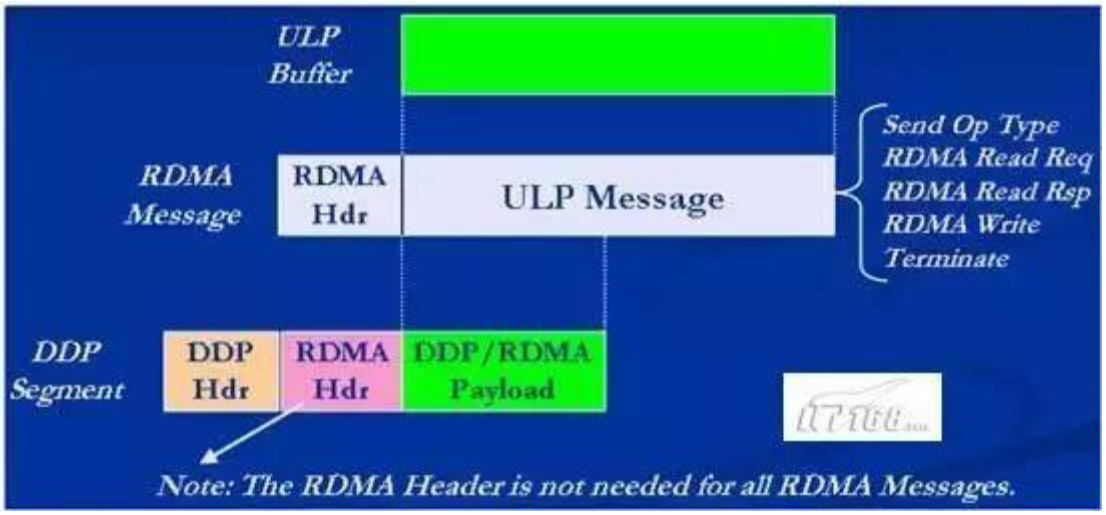


图 4.3 DDP 层拆分 RDMA 消息示意图

DDP 数据段是 DDP 协议数据传输的最小数据单元，包含 DDP 协议头和 ULP 载荷。DDP 协议头包含 ULP 数据的最终目的地址的位置和相关控制信息。DDP 层将 ULP 数据分段的原因之一就是 TCP 载荷的最大长度限制。DDP 的数据传输模式分为 2 种：tagged buffer 方式和 untagged buffer 方式。tagged buffer 方式一般应用于大数据量传输，例如磁盘 I/O、大数据结构等；而 untagged buffer 方式一般应用于小的控制信息传输，例如：控制消息、I/O 状态信息等。

MPA 层在 DDP 层传递下来的 DDP 消息中，MPA 层通过增加 MPA 协议头、标志数据以及 CRC 校验数据构成 FPDU(framed PDU)数据段，处理过程如图 4.4 所示。

MPA 层便于对端网络适配器设备能够快速定位 DDP 协议头数据，根据 DDP 协议头内设置的控制信息将数据直接置入相应的应用内存区域。MPA 层具备错序校正能力，通过使能 DDP，MPA 避免内存复制的开销，减少处理乱序数据包和丢失数据包时对内存的需求。MPA 将 FPDU 数据段传送给 TCP 层处理。

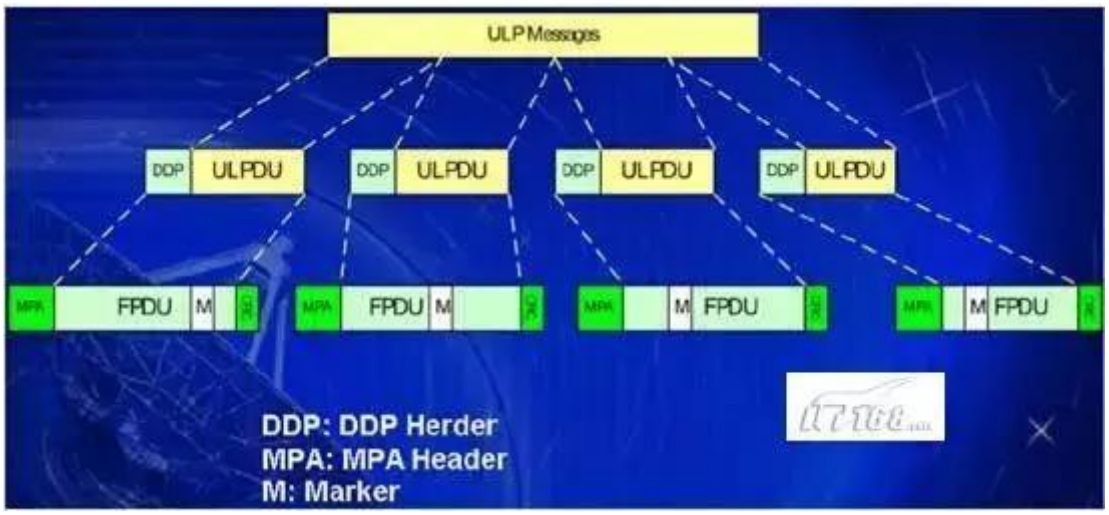


图 4.4 MPA 层拆分 DDP 消息示意图

TCP 层将 FPDU 数据段拆放置在 TCP 数据段中，确保每个 TCP 数据段中都包含 1 个单独的 FDPU。MPA 接收端重新组装为完整的 FPDU，验证数据完整性，将无用的信息段去除，然后将完整的 DDP 消息发送到 DDP 层进行处理。DDP 允许 DDP 数据段中的 ULP 协议(Upper Layer Protocol)数据，例如应用消息或磁盘 I/O 数据，不需要经过 ULP 的处理而直接放置在目的地址的内存中，即使 DDP 数据段乱序也不影响这种操作。

第11章 RoCE（RDMA over Converged Ethernet）原理

RoCE（RDMA over Converged Ethernet）是一种允许通过以太网使用远程直接内存访问（RDMA）的网络协议。华为 CE8860 交换机插入 CX4 归一化网卡（10GE/25GE） 后，立即支持 RoCE 10GE/25GE 通

信。

由于具备明显性能和成本优势，在 NAS 存储集群中采用 RoCE 协议，将逐渐成为市场主流。

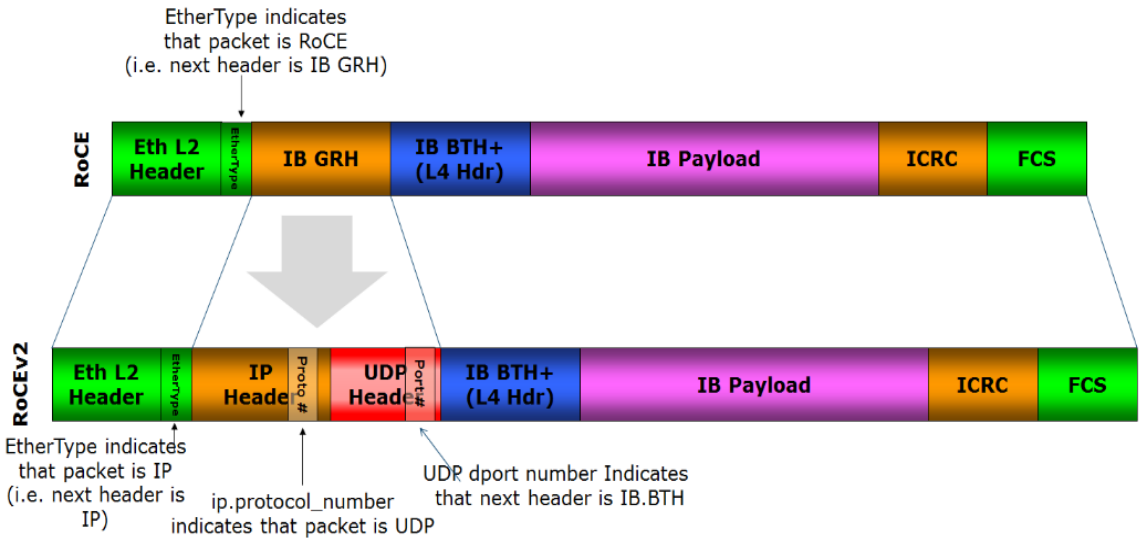
RDMA over Converged Ethernet (RoCE)是一种网络协议，允许应用通过以太网实现远程内存访问。目前 RoCE 有两个协议版本，v1 和 v2。其中 RoCE v1 是一种链路层协议，允许在同一个广播域下的任意两台主机直接访问。而 RoCE v2 是一种 Internet 层协议，即可以实现路由功能。虽然 RoCE 协议这些好处都是基于融合以太网的特性，但是 RoCE 协议也可以使用在传统以太网网络或者非融合以太网络中。

网络密集型应用，例如存储或者集群计算等，需要网络支持大带宽和低时延特性。RDMA 的优势相比其他网络应用程序接口，例如 Berkeley 接口，就是低时延，低 CPU 负载和大带宽。RoCE 可以比他的前任 iWARP 协议要实现更低的时延。目前 RoCE HCAs(Host Channel Adapter)最低时延为 1.3 微秒，而 iWARP HCA 的最低时延为 3 微秒（2011 年数据）。

RoCE v1 协议属于 ETH 链路层协议，协议类型为 0x8915。这意味着其报文长度最大不超过 1500 字节，在支持超大帧的情况下不超过

9000 字节

RoCE v2 协议使用的是 UDP/IPv4 或者 UDP/IPv6。UDP 目的端口号 4791 就是给 RoCE v2 预留的。由于 RoCE v2 报文能够支持路由功能，因此有时候也称他为可路由的 RoCE 或者 RRoCE。尽管在通常情况下 UDP 报文的传输顺序是没有保证，但是 RoCE v2 规范要求了有相同 UDP 源端口和目的地址的报文必须顺序传送。另外，RoCE v2 定义了拥塞控制机制，即使用 IP ECN 位来进行标记，同时用 CNP 帧来进行确认。当前 Mellanox OFED 2.3 及更新的版本，Linux Kernel v4.5 都能够支持 RoCE v2,总体来说支持 RoCE v2 的软件还处于新兴阶段。



第12章　不同 RDMA 技术的比较

目前，有三种支持 RDMA 的技术：IB、以太网 RoCE、以太网 iWARP。这三种技术使用本文中定义的同 一 API，但它们有着不同的物理层

和链路层。

在以太网解决方案中,RoCE 相对于 iWARP 来说有着明显的优势，这些优势体现在延时、吞吐率和 CPU 负载。RoCE 被很多主流的方案所支持，并且被包含在 Windows 服务软件中（IB 也是）。

RDMA 技术基于传统网络的概念，但与 IP 网络又有些不同。最关键的不同是 RDMA 提供了一种消息服务，利用这种服务,**应用程序(通过 Verbs)可以直接访问远程计算机上的虚拟内存**。消息服务可以用来进行网络中进程 的通信（IPC)、与远程服务器进行通信和在一些上层协议的协助下与存储设备进行数据传递。上层 协议（ULPs）有很多，例如：**iSCSI 的 RDMA 扩展（iSER)、SCSI RDMA 协议（SRP)、SMB、Samba 、Lustre、ZFS** 等等。

RDMA 利用旁路和零拷贝技术提供了低延迟的特性，同时，减少了 CPU 占用，减少了内存带宽瓶颈， 提供了很高的带宽利用率。RDMA 所带来的关键好处得益于 RDMA 消息服务呈现给应用的方式，和底层用来 发送和传递这些消息的技术。RDMA 提供了基于 IO 的通道。这种通道允许一个应用程序通过 RDMA 设备对 远程的虚拟内存进行直接的读写。

在传统的套接字网络中，应用程序要向操作系统申请使用网络资源时，要通过特定的 API 来管理程序的 相关行为。但是，RDMA 使用操作系统仅仅建设一个通道，然后就可以在不需要操作系统的干预下，应用 程序之间就能够进行直接的消息传递。消息可以是 RDMA 读或写

操作，也可以是发送/接收操作。IB 和 RoCE 也都支持多播模式。

12.1 IB 和 TCP、Ethernet 比较

IB 在链路层提供的特性有：基于信任的流控制机制用来进行拥塞控制。它也支持使用虚拟局域网（VLs），虚拟局域网能够使高层协议简单化，并且提供高质量服务。IB 在 VL 中严格保证数据在一条路径中能够 按序到达。IB 的传输层提供可靠性和交付性保障。

IB 网络层拥有的特性使它能够很简单地在应用程序的虚拟内存之间传递消息，尽管应用参与通信 的应用程序运行在不同的物理服务器上。因此，将 IB 传输层和软件传输接口组合起来，可以看成是一 种 RDMA 消息传输服务。包括软件传输接口的整个协议栈，包含了 IB 消息服务。

最重要的一点是，每个应用程序都能直接访问集群中的设备的虚拟内存。这意味着，应用程序 传输消息时不需要向操作系统发出请求。与传统的网络环境相比，传统网络中共享的网络资源 归操作所有，不能由用户态程序直接使用，所以，一个应用程序必须在操作系统的干预下将数据 从应用程序的虚拟内存通过网络栈传送到网线上。类似地，在另一端，应用程序必须依靠操作 系统获取网线上的数据，并将数据放到虚拟缓冲区中。

TCP/IP/Ethernet 是一种子面向字节流的传输方式 ，信息以字节的形式在套接字应用程序之间 传递。TCP/IP 本身是不可靠的（传

输过程中数据可能丢失或者失序),但是它利用传输控制协议（TCP）来实现可靠性机制。TCP/IP 在所有操作中都需要操作系统的干预，包括网络两终端结点 的缓冲区拷贝。在面向字节流的网络中，没有消息的边界概念。当一个应用想要发送一个数据包， 操作系统把这些字节数据放入内存中属于操作系统的一个匿名缓冲区，当数据传输完毕时，操作 系统把它缓冲区中的数据拷贝到应用程序的接收缓冲区。这个过程在每个包到达时都会重复执行， 直到整个字节流被接收到。TCP 负责将任何因拥塞导致的丢包进行重发。

在 IB 中，一个完整的消息被直接发送到一个应用程序。一旦一个应用程序请求了 RDMA 的读或写传输， IB 的硬件将需要传输的数据按照需要分割成一些数据包，这些数据包的大小取决于网络路径的最大 传输单元。这些数据包通过 IB 网络，被直接发送到接收程序的虚拟内存中，并在其中被组合为一个 完整的消息。当整个消息都到达时，接收程序会接收到提示。这样，发送程序和接收程序在直到整个消息被发送到达接收程序的缓冲区之前都不会被打扰中断。

12.2 RoCE 和 InfiniBand 比较

RoCE 和 InfiniBand，一个定义了如何在以太网上运行 RDMA，而另一个则定义了如何在 IB 网络中如何运行 RDMA。RoCE 期望能够将 IB 的应用，主要是基于集群的应用，迁移到融合以太网中，而在其他应用中，IB 网络仍将能够提供比 RoCE 更高的带宽和更低的时延。

12.3 RoCE 和 IB 协议的技术区别

1、拥塞控制：RoCE 所依赖的无丢包网络基于以太网流控或 PFC（Priority Flow Control）来实现。RoCEv2 则是定义了拥塞控制协议，使用 ECN 做标记和 CNP 帧来做确认。而 IB 则是使用基于信用的算法来保证 HCA-HCA 之间的无丢包通信。

2、时延：当前 IB 交换机普遍要比以太交换机拥有更低的时延，以太网交换机一般的 Port-to-Port 时延在 230ns，相比 IB 交换机在同样端口数的情况下 100ns 的时延，以太交换机还是要高出不少。

3、配置:配置一个 DCB 以太网络要远比配置一个 IB 网络要复杂的多，同理，运维也要复杂的多。

12.4 RoCE 和 iWARP 的区别

RoCE 和 iWARP，一个是基于无连接协议 UDP，一个是基于面向连接的协议，如 TCP。RoCEv1 只能局限在一个二层广播域内，而 RoCEv2 和 iWARP 都能够支持三层路由。相比 RoCE，在大型组网的情况下，iWARP 的大量 TCP 连接会占用大量的额内存资源，对系统规格要求更高。另外，RoCE 支持组播，而 iWARP 还没有相关的标准定义。

第13章　Intel Omni-Path 和 InfiniBand 对比分析

回顾 InfiniBand Trade Association (IBTA)，IBTA 的 9 个主要董事成员 CRAY、Emulex、HP、IBM、intel、Mellanox、Microsoft、Oracle、Qlogic 中只有 Mellanox 和 Emulex 专门在做 InfiniBand，其他成员只是扮演了使用 InfiniBand 的角色。而 Emulex 由于业务不景气也在 2015 年的 2 月被 Avago 收购，Qlogic 的 infiniband 业务在 2012 年也全部卖给 Intel 了。

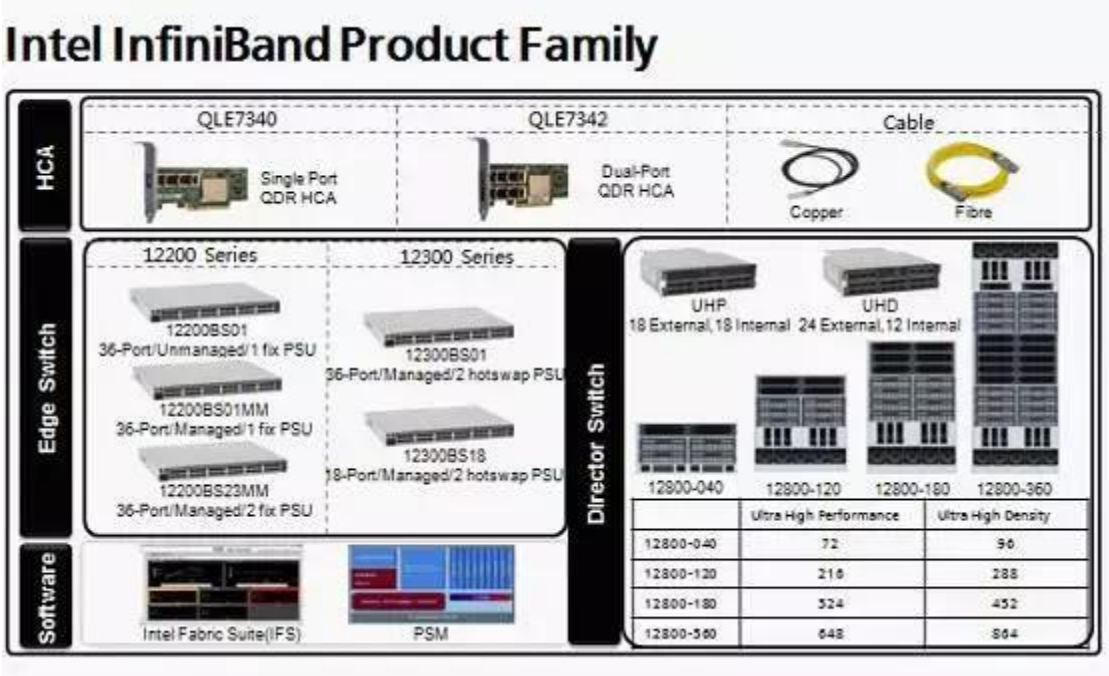
但是收购了 Qlogic 的 InfiniBand 业务的 Intel 又另辟新径，推出了自己的一整套叫做“True Scale Fabric”的高性能计算架构的解决方案，独立提出了一套 Omni-Path Host Fabric Interface 新接口和对应的交换机产品，每个端口支持 100G 速率，直接叫板 InfiniBand EDR。

13.1 Intel True Scale Fabric 介绍

Intel True Scale Fabric 软件架构和产品系列可以说是一个整体网络解决方案架构，该架构即为 InfiniBand 软件协议栈，Omni-Path 协议栈也是基于 Intel True Scale Fabric 实现。下面我们先讨论 Intel True Scale Fabric 软件架构和产品。

上图是对 InfiniBand 软件堆栈实现，针对上层 Intel 提供 Verbs 和 PSM(性能扩展消息库)两个编程接口，PSM 也是针对 MPI 设计，专门面向 MPI 通信，通过高性能计算优化的库。PSM 是一种专为优化 MPI 性能需求而构建的“轻型”库，虽然不对其他应用提供接口，但其他应用仍调用 Verbs 接口。Intel HCA 卡采用 On-Load 模式实现。

13.2 Intel True Scale InfiniBand 产品

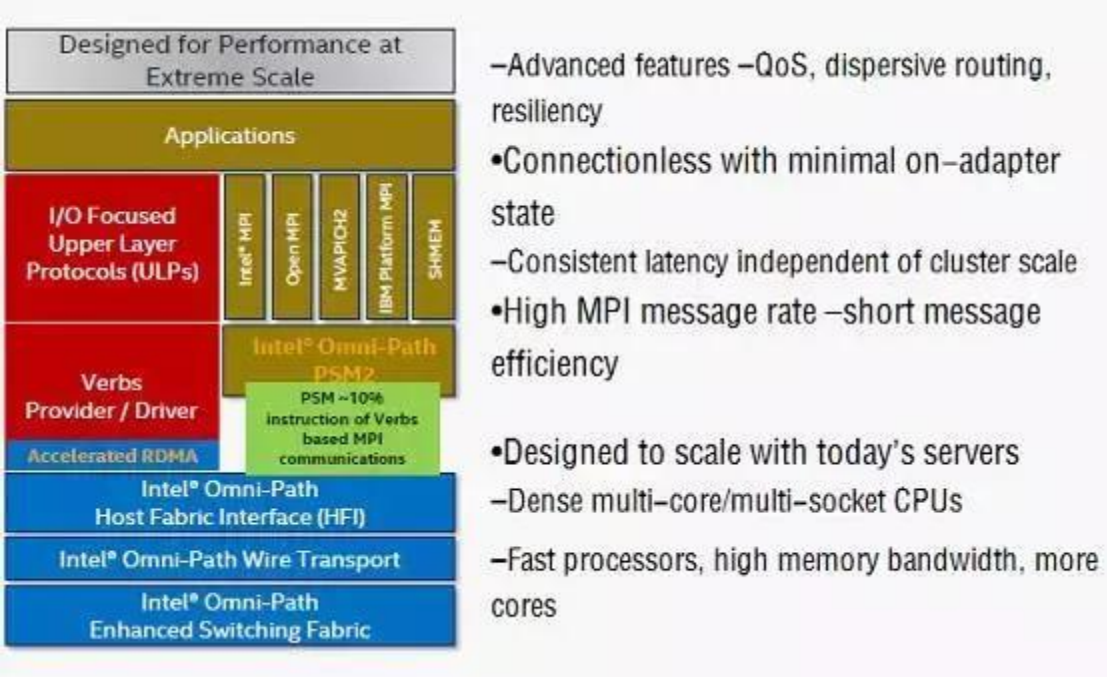


上图展现了 Intel 完善的 True Scale 品牌的 InfiniBand 产品。面对未来高性能、大带宽应用和业务,既然已经有了 InfiniBand,为什么还要另辟蹊径开发 Omni-Path 呢。

先看看简单分析，从产品支持的端口数量和网络组网层次来看，Omni-Path 是具有优势的，对网络设备来说，网络时延、服务质量保证方面 Omni-Path 也是有优势的，相比 InfiniBand，物理介质层算是比较大的优化吧。

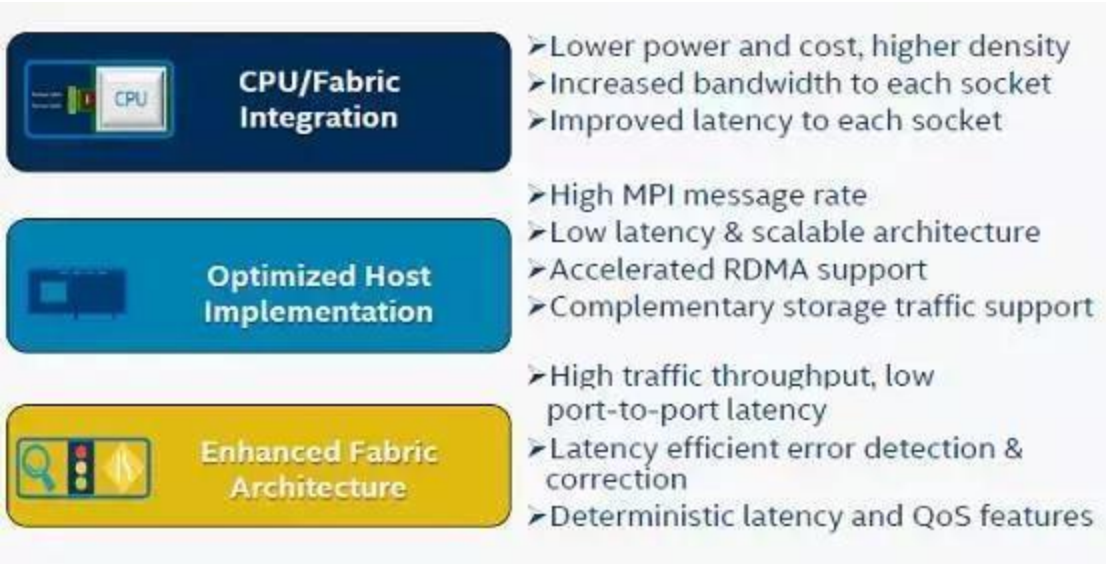
	Omni-Path	Infiniband	说明
端口速率	100Gbps	100Gbps	速率相当
基数端口数目对比	单交换芯片48口	单交换芯片36口	同等规模组网下，减少了交换数目和线缆数目
盒式交换组网层次	<48口时单层组网	<36口时单层组网	同等规模组网时，减少交换数目的层次，降低时延
框式交换组网层次	<768口时单层组网	<648口时单层组网	同上
单跳时延对比	110ns	130ns	号称时延更低
QoS调度	优先保障高优先级分片通过		对小的MPI包更优化

看到下面的个张图似乎和前面的 Intel True Scale InfiniBand 软件协议栈很相像，是的，Intel 收购 QLogic 公司的 InfiniBand 部门和 Cray 互联部门，同时也得到 QLogic InfiniPath 产品，随后网络产品的名字从 InfiniPath 更新成 True Scale。

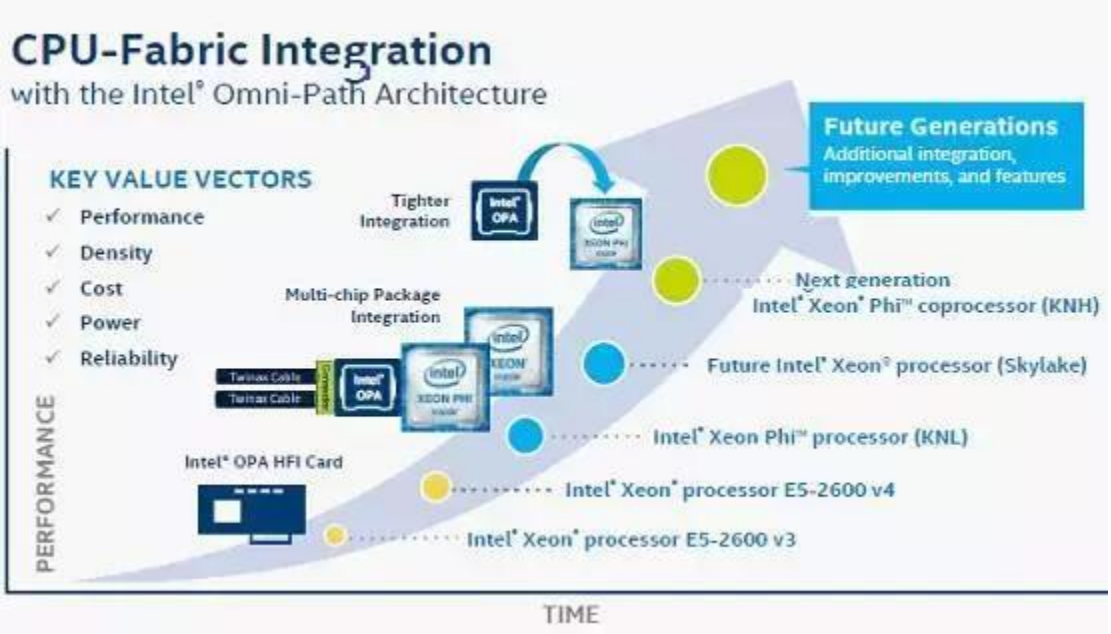


13.3 Intel Omni-Path 产品

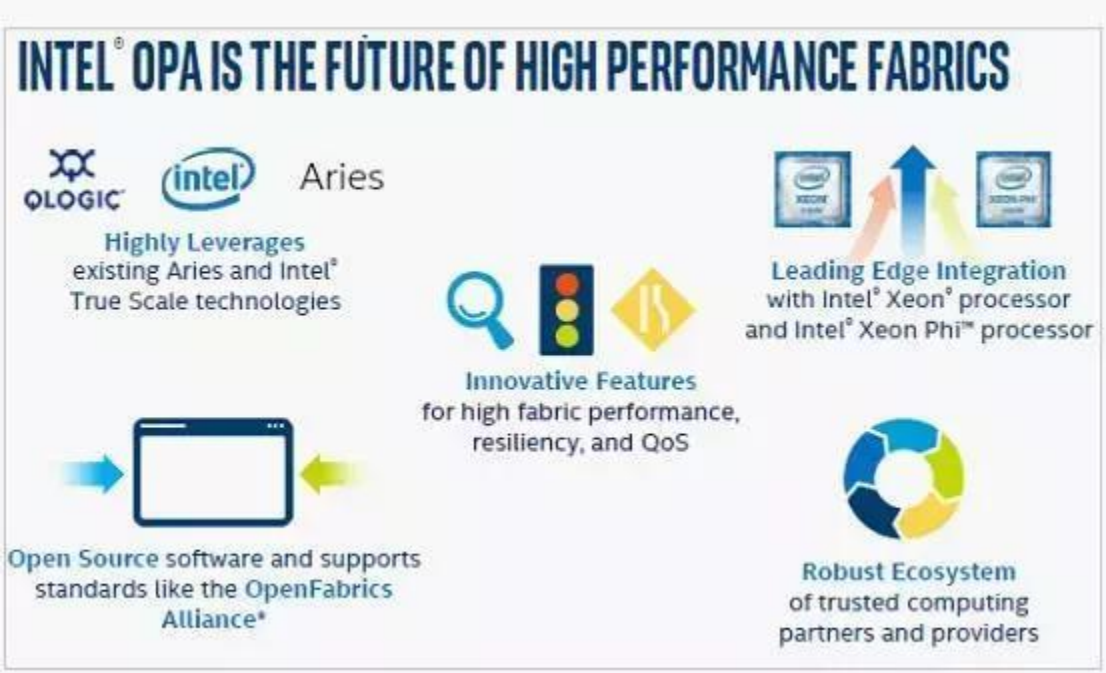
Omni-Path 沿用了 True Scale 产品名称和技术，主要的变化是在把物理层把速度从 40G 提到了 100G。为了兼容更加开源的系统，Omni-Path 也是基于开源标准的 OFED 架构(Mellanox 也是采用该框架)，并将 API 接口开放。



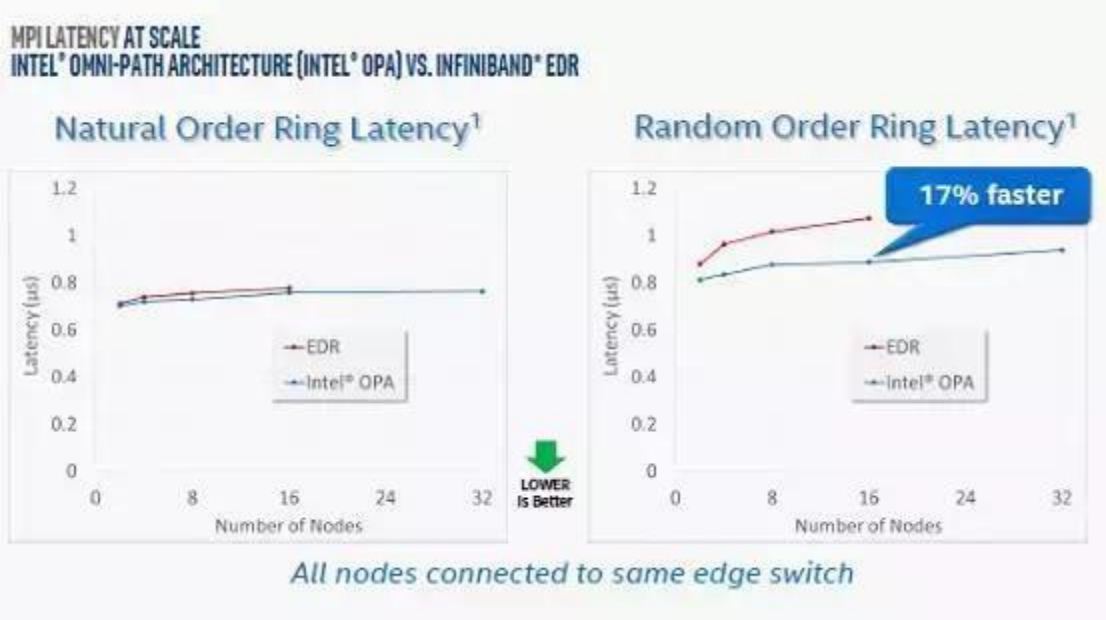
Intel 在 CPU 上集成了 Omni-Path 相关功能，这也意味着 Omni-Path 通信效率上更加高效，但会让自己的网络依赖于 CPU，至少在处理器上开放性还是做的比较有局限性。



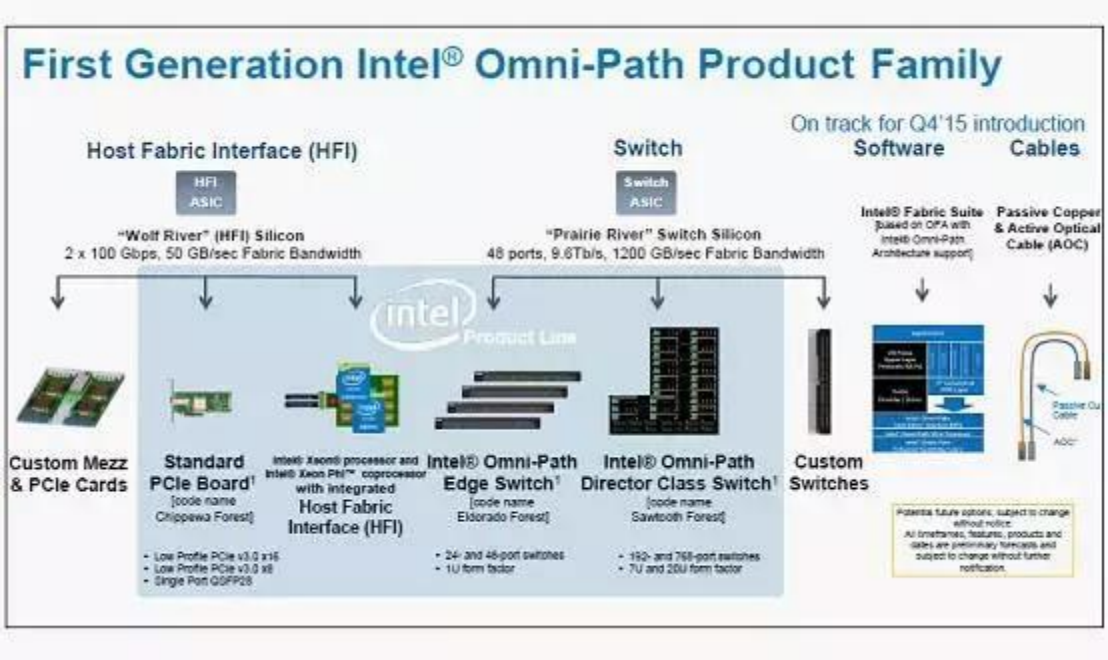
通过收购 Cray 互联部门,Intel 在 Omni-Path 引入了 1.5 层的概念。它被称作链接传输层(Link Transport Layer)，基于 Cray 的 Aries 基础互联技术优化底层数据通信,提供可靠的 2 层数据包交付、流控和单连璐控制。这也算是对收购 Cray 的 Aries 技术的充分利用。



从测试数据来看，在顺序读写 IO 模型下，Omni-Path 的时延优势并不明显；但是在随机 IO 模型和项目条件下，Omni-Path 具有比较大的优势。



Omni-Path 架构产品线也很完备(如下图)。主要包括 HFI(Host Fabric Interface)网卡、交换机、Omni-Path 软件堆栈、线缆和芯片等。



Intel Omni-Path Fabric100 系列产品系列包含边缘交换机、核心交换机等组成端到端的 HPC 产品和方案。Intel Omni-Path 边缘交换机提供 100 Gbps 端口带宽和低时延(详细规格可参考 Intel 官网资料)。

简单总结： 俗话说本是同根生、相煎何太急。这种瑜亮情节经常在公司兼并产品重叠事，显得尤为激烈。但是存在即合理，首

先，回首一下 Omni-Path 必须得承认，相比自己的 InfiniBand 还是有很都优势的。如时延更低，是采用 Aries 技术对介质层、网路层优化，在 CPU 上集成 Omni-Path 支持能力等。

其次，Intel 通过 Intel Fabric Builders Program 构建自己强大的生态环境，当前有多大型超算中心正在使用 mni-Path 网络。

从我个人观点来看，Omni-Path 在技术和协议架构上并没有太多突破性的变化，让自己的 InfiniBand 产品处于非常尴尬的状态(Intel 主攻 Omni-Path)。Omni-Path 相比 InfiniBand，在网络时延、带宽提升方面并没有跨数量级的优势。在未来高性能市场，Mellanox 的 InfiniBand 也在迅速发展。站在市场角度，我更希望 InfiniBand 和 Omni-Path 技术都能找到适合发展的市场空间，分别发挥出自己不同的优势。

第14章 RDMA 关键技术延伸

14.1 RDMA 指令的选择

RDMA 各种命令的效率

1、 Unreliable 类的连接，效率优于 reliable 类型，这是因为 reliable 类型，芯片需要处理 ACK 和重传

2、 RDMA Write 的性能优于 RDMA Read，这是因为 RDMA Read 的处理流程复杂，芯片需要保留大量的控制结构。并且大多数芯片支持的 Read 并发数远远小于 RDMA write 的并发送

3、 RDMA Send 的性能往往低于 RDMA write，这是因为 RDMA Send 存在额外的 CQE 生成、中断上报、驱动处理的流程

实际上各种命令在不同的场景下性能会有较大差异，和各种芯片的具体实现模式也有关系，需要先针对性测试，然后根据测试结果和需求来选择

14.2 慎用 atomic 类指令

当前大多数芯片的设计对 atomic 类操作并没有做细致的优化（atomic 类貌似用得也不多），且该类操作不可避免地导致芯片内部的互斥和竞争，性能低下。如果要使用，需要详细了解内部锁的实现方案，了解如何较少芯片内部的锁竞争。

1、 **协议对 atomic 指令的定义为：**

也就是：只保证针对一个 RDMA 设备(比如一个 IB 卡)的访问的 atomic，如果存在其它共存的 RDMA 设备对该地址的访问，atomic 特性就被破坏了。（这是很合理的假设，让多个 IB 卡这种玩意儿在完成 atomic 指令是相互通信是极不现实的）

2、 **Atomic 指令的性能**

Atomic 类操作的实现，往往需要在芯片内部完成一些锁的操作，这可能会给芯片性能尤其是多连接的性能带来极大地影响；具体影响的程度和芯片的设计相关，需要了解芯片的实现细节来针对性设计。比如：如果芯片内部按照 atomic 操作地址的低 12 位 HASH 来完成相关锁操作(hash 值相同的共享锁)，而业务下发的 atomic 操作地址是按照页面对齐的,就会造成固定的锁被大量争用，而其它锁空闲。

14.3 减少交互次数

14.3.1 Wr 聚合

业务层在下发 wr 的时候，如果发送链表中还有新的待发送 wr，通过 ib_send_wr 的 next 字段一次聚合多个 wr 下发给驱动，驱动挨个 wr 处理完成后一次性敲 doorbell,从而减少交互次数,提高效率。

14.3.2 SGE 聚合

当前在存储，使用 SGE 的模式下发 IO，默认的模式下每个 SGE 会在芯片生成一个 RDMA 命令(因为一个 RDMA 命令只能携带一个连续内存块)。作为改进，我们可以检测 WR 中各个 SGE 对应的远端地址，如果远端是连续的，就把这些 SGE 合并，从而减少底层交互的 RDMA 命令数量。

14.3.3 使用 imm 数据

在携带数据特别少的场景(小于 4BYTES)，可以考虑使用 imm 数据模式(XXX with with Immediate 命令)，从而避免 DMA 数据段的开销（这种模式下，芯片就不需要考虑数据段，仅完成命令头的 DMA)。

14.3.4 使用 inline 数据

通常模式下数据通过 SGE 来交互，数据的地址存放在相关描述符中，这些描述符被写入到芯片和 OS 的共享内存中，芯片处理时需要先把描述符 DMA 到芯片，解析得到地址后再把数据 DMA 到芯片。采用 inline 数据，就是允许把数据放到描述符中，这样芯片就少 DMA 一次。注意：

1、 采用这种方式的代价是数据拷贝，一般仅适用于数据较小且大于 4BYTES(这种就用 imm 数据)的场景。如果芯片支持的 inline 数据较大，需要通过测试来找到恰当地 inline 数据长度门限值。
2、 这种方式对描述符的大小由要求，在芯片设计的时候需要考虑系统的“read completion combining size ”，按照这个值对齐达到最佳的 PCIE 利用率。

14.3.5 CQE 中使用 inline 数据

和发送方向的 inline 数据一样，对 SEND 类型的数据，如果数据量较小，直接放到 CQE 中避免多次启动 DMA。该方式也会产

生 CPU 拷贝，需要测试最佳的数据长度值。

14.3.6 WC 聚合

在满足业务要求的情况下，业务层通过 single 标志位限制 CQ 中断的产生，避免过多的中断

14.4 运行模式选择

14.4.1 连接的模式

连接的模式包括 RC、UC、RD、UD、XRC 五种，每种模式支持不同的指令类型，并有不同的可扩展性和相关限制。

可扩展性而言：UD > RD > XRC > RC=UC

就性能而言，Unreliable 模式(UD,UC)要优于 reliable 模式，这是因为芯片不需要管理 ACK/NAK 和对应的状态，在高可靠的网络环境，可以采用 Unreliable 模式(Unreliable 是能够保证单个报文中的数据的完整性的)。

在连接模式的选择上，需要综合当前的规格要求、选择的指令类型、芯片的最佳 QP 数量等各种因素。在要求性能的场景，供选择的一般是 UD、XRC、和 RC，RD 存在单并发的限制，UC 很多芯片根本就不支持。

14.4.2 运行模式

1、 通知模式：应用在使能 WC 通知后，驱动在得到 WC(work completion)就会通知应用，应用通过 poll_cq 来获得已经完成接收的 send 消息或者完成发送的 wr，处理完成后重新使能 WC 通知功能，这是当前大多数应用采用的模式。这种模式下有可能出现大量中断，影响性能。

2、 在 all poll 模式下，应用不依赖驱动的通知，通过主动 poll cq 得到相关的 WC 信息。这种模式需要芯片的配合才有意义：芯片需要直接把 CQE 写入到 CQ 中，而不是在对应的中断中处理。这种模式避免了大量中断的出现，能够提高性能。但是其一般需要从整体上为 RDMA 的 IO 处理分配固定的 CPU 核心。

3、 通知+POLL 的模式，这种模式下载得到通知后并 poll 完 CQ 中的 CQE 后，会再 poll 一定次数或者时间，测试表明这种模式也能在一定程度上提高性能。比如 XIO 就采用了这种模式。

4、 除了 poll CQ 外，还有一种 POLL 内存的模式：通常情况下在数据接收端，应用感知到 RDMA WRITE 的数据写入是通过在发送端执行 RDMA WRITE 后，使用 RDMA send 消息法术控制消息，接收端收到控制消息后感知到相关的写入。采用 POLL 模式的时候，其直接读取 RDMA WRITE 目的地址中的数据来感知 RDMA WRITE 操作写入完成。(这是在一些 PAPER 中出现的方式，但是并不符合相关协议的要求，按照协议，RDMA WRITE 之间只有数据 delivery 的顺序保证，而没有 placement 的顺序保

证。也就是即使应用读取到了下一个 RDMA WRITE 写入的标志，也不表示上一个 RDMA WRITE 就已经完成）

总之，在完全考虑性能的环境，all poll 的方式能够一定程度上降低时延，提高性能。

14.5 性能与并发

适当地并发提高性能

当前的各种 RDMA 芯片，内部通常存在流水线或者多个处理核心，这些流水线和处理核心和连接之间往往存在某种绑定关系，也就是如果使用单个连接，往往不能充分发挥芯片的性能（有可能是芯片单个核心能力的限制，也可能是 CPU 单个核心的能力限制），需要使用多个连接。

连接数量并不是越多越好，其最佳数量的选择受限于很多因素，比如：CPU 核心的能力、NIC 内部处理单元的能力、芯片内部 cache、其它业务负载对 CPU 的消耗、NIC 内部连接间的并发竞争。具体的最佳连接数量需要综合选择并通过对比测试来验证。

对芯片的要求：支持多队列，能够把不同的连接的中断分散到不同的 CPU 核心，最好能够支持 Flow Director .

14.6 避免CPU缓存抖动

这其实不算是RDMA专有的要求，建议采用相关共享数据结构cache line对齐的做法。需要说明的是：首先要做的不是cache line对齐，而是如何避免数据的访问竞争，通过适当地流程设计来避免访问。比如采用连接和CPU绑定的做法。

14.7 避免芯片内部的缓存Miss

当前很多芯片都有内部cache，如果产生cache miss，就需要通过PCIE通道从OS的内存中去读取数据，降低性能。这些cache通常包括：MR相关映射、QP状态、WR等，这些资源如果占用量超过芯片内部cache的大小且频繁切换，就会带来性能上的下降。

相关的应对措施包括：

1、 了解芯片的内部结构，哪些做了cache，具体的规格是多少。

2、 通过各种优化措施，减少乃至消除cache miss。比如减少MR的数量、减少QP的数量

3、 需要说明的是：该项优化措施和提高并发等方式往往存在冲突，需要通过测试数据来寻找最后组合

4、 不是出现cache miss就一定会降低性能，取决于增加QP\MR\WR并发等因素带来的性能增加和cache miss带来的性能下降的平衡，相关参数需要通过测试来确定

注：对自研芯片，最好能够直接提供相关的统计数据，方便性能问题的分析。

14.8 时延的隐藏

时延隐藏通常是把串行的事务并行化达到降低时延的目的。

14.8.1 利用 Prefetch 预取指令

RDMA 应用中，存在一些和芯片同步交互的接口，比如：ib_post_send、ib_post_recv、mr 的注册（在用户态，可能需要在 IO 流程中做 MR 管理）等等，这些接口一般存在一定的调用时延(可达 us 级，这和各种芯片的具体实现相关，时延的大小最好测试到数据)。通过分析应用的 IO 路径上的内存访问，可能能够发现一些内存访问有较大概率不在 cache 中(比如是一个 SGL 或者链表中指向的大块数据)，这时可以先在调用 verbs 接口前执行 prefetch，让内存的读取和 verbs 接口执行并发，然后在执行内存访问操作就能够直接从 cache 中读取数据。这种分析不应当局限在一个 IO 处理内部，通常需要利用上一个 IO 的处理操作和下一个 IO 的 verbs 操作来完成并发。

14.8.2 异步交互操作优先

在整个 RDMA 应用流程中，往往涉及一些和对端的异步交互(比如 XNET 会通过 send 消息请求对端页面)和一些耗时地本地同步操

作(比如 CRC 计算、MR 注册等)，通过优先发起异步交付操作，使得在一个 IO 看来，本地同步操作和异步操作并行执行，达到降低时延的目的。

说明：一个典型的例子是用户态的 RDMA 应用需要在 IO 流程中完成注册 MR 和请求对端页面两种操作，先发起请求对端页面，在发起 MR 注册,在整个 IO 看起来 MR 注册和请求对端页面就并行了。

14.9 RDMA 性能分析

PCIE 效率和相关数据获取

通过分析 PCIeRdCur(DMA reads) 和 PCIeItoM (DMA writes) 这两个 performance counter，结合具体业务数据量，可以初步分析当前的交互模式对 PCIE 的利用率情况。

具体如何得到 PCIeRdCur、PCIeItoM 等 PCIE 相关性能数据，intel 提供了 API 和相关的说明文档，如下：

https://software.intel.com/en-us/articles/intel-performance-counter-monitor#license

http://www.correlsense.com/intel-performance-counter-monitor/