RDMA技术详解（三）：理解RDMA SGL


围城
欢迎大家follow：https://github.com/Tjcug

2 人赞同了该文章

1. 前言

在使用RDMA操作之前，我们需要了解一些RDMA API中的一些需要的值。其中在ibv_send_wr我们需要一个sg_list的数组，sg_list是用来存放ibv_sge元素，那么什么是SGL以及什么是sge呢？对于一个使用RDMA进行开发的程序员来说，我们需要了解这一系列细节。

2. SGE简介

在NVMe over PCIe中，I/O命令支持SGL(Scatter Gather List 分散聚合表)和PRP(Physical Region Page 物理(内存)区域页), 而管理命令只支持PRP;而在NVMe over Fabrics中，无论是管理命令还是I/O命令都只支持SGL。

RDMA编程中，SGL(Scatter/Gather List)是最基本的数据组织形式。 SGL是一个数组，该数组中的元素被称之为SGE(Scatter/Gather Element)，**每一个SGE就是一个Data Segment(数据段)。** RDMA支持Scatter/Gather操作，具体来讲就是RDMA可以支持一个连续的Buffer空间，进行Scatter分散到多个目的主机的不连续的Buffer空间。Gather指的就是多个不连续的Buffer空间，可以Gather到目的主机的一段连续的Buffer空间。

下面我们

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···

```
struct ibv_sge {
        uint64_t        addr;
        uint32_t        length;
        uint32_t        lkey;
};
```

- addr: 数据段所在的虚拟内存的起始地址 (Virtual Address of the Data Segment (i.e. Buffer))
- length: 数据段长度(Length of the Data Segment)
- lkey: 该数据段对应的L_Key (Key of the local Memory Region)

2. ivc_post_send接口

而在数据传输中，发送/接收使用的Verbs API为：

- ibv_post_send() - post a list of work requests (WRs) to a send queue 将一个WR列表放置到发送队列中 ibv_post_recv() - post a list of work requests (WRs) to a receive queue 将一个WR列表放置到接收队列中

下面以ibv_post_send()为例，说明SGL是如何被放置到RDMA硬件的线缆(Wire)上的。

ibv_post_send()的函数原型

```
#include <infiniband/verbs.h>

int ibv_post_send(struct ibv_qp *qp,
                  struct ibv_send_wr *wr,
```

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···

ibv_post_send（ ）将以send_wr开头的工作请求（WR）的列表发布到Queue Pair的Send Queue。 它会在第一次失败时停止处理此列表中的WR（可以在发布请求时立即检测到），并通过bad_wr返回此失败的WR。

参数wr是一个ibv_send_wr结构，如中所定义。

3. ibv_send_wr结构

```
struct ibv_send_wr {
        uint64_t                wr_id;                  /* User defined WR ID */
        struct ibv_send_wr      *next;                  /* Pointer to next WR in list,
        struct ibv_sge          *sg_list;               /* Pointer to the s/g array */
        int                     num_sge;                /* Size of the s/g array */
        enum ibv_wr_opcode      opcode;                 /* Operation type */
        int                     send_flags;             /* Flags of the WR properties
        uint32_t                imm_data;               /* Immediate data (in network
        union {
                struct {
                        uint64_t        remote_addr;    /* Start address of remote mem
                        uint32_t        rkey;           /* Key of the remote Memory Re
                } rdma;
                struct {
                        uint64_t        remote_addr;    /* Start address of remote mem
                        uint64_t        compare_add;    /* Compare operand */
                        uint64_t        swap;           /* Swap operand */
                        uint32_t        rkey;           /* Key of the remote Memory Re
                } atomic;
                struct {
```

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···

```
                        uint32_t        remote_qpn;     /* QP number of the destinatio
                        uint32_t        remote_qkey;    /* Q_Key number of the destina
                } ud;
        } wr;
};
```

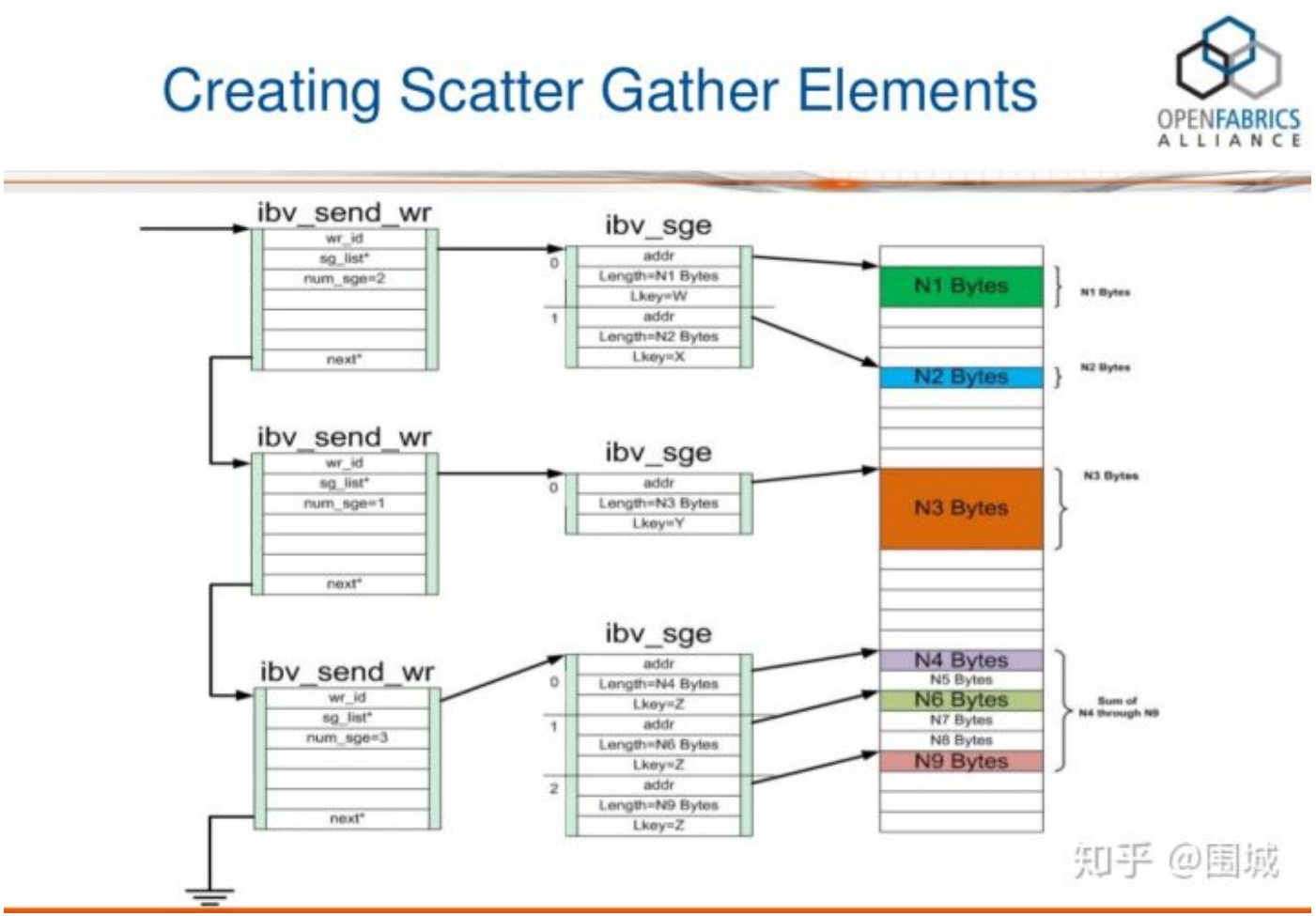
在调用ibv_post_send()之前，必须填充好数据结构wr。 wr是一个链表，每一个结点包含了一个sg_list(i.e. SGL: 由一个或多个SGE构成的数组), sg_list的长度为num_sge。

4. RDMA 提交WR流程

下面图解一下SGL和WR链表的对应关系，并说明一个SGL (struct ibv_sge *sg_list)里包含的多个数据段是如何被RDMA硬件聚合成一个连续的数据段的。

4.1 第一步：创建SGL

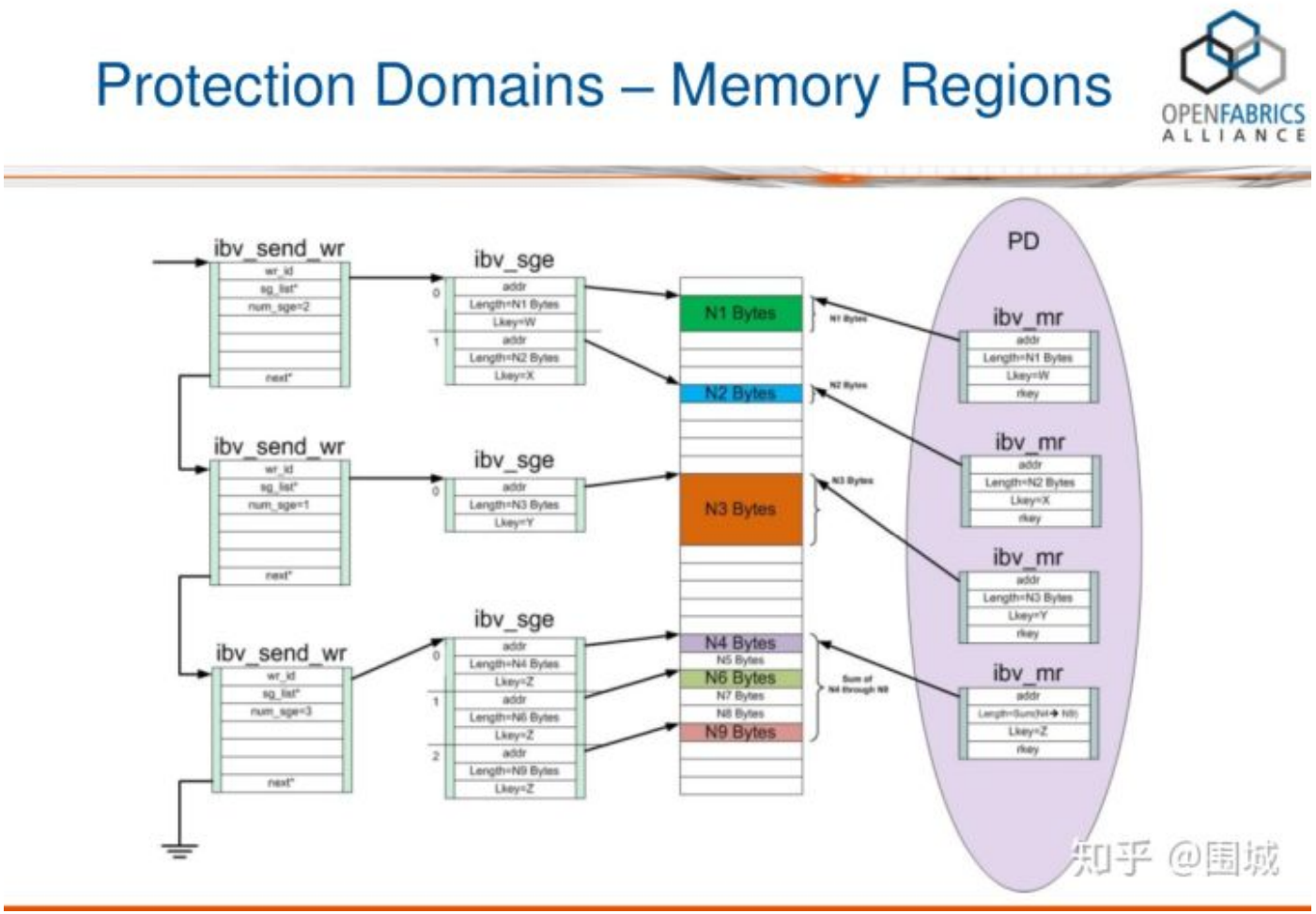
▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···



从上图中，我们可以看到wr链表中的每一个结点都包含了一个SGL，SGL是一个数组，包含一个或多个SGE。通过ibv_post_send提交一个RDMA SEND 请求。这个WR请求中，包括一个sg_list的元素。它是一个SGE链表，SGE指向具体需要发送数据的Buffer。

4.2 第二步：使用PD进行内存保护

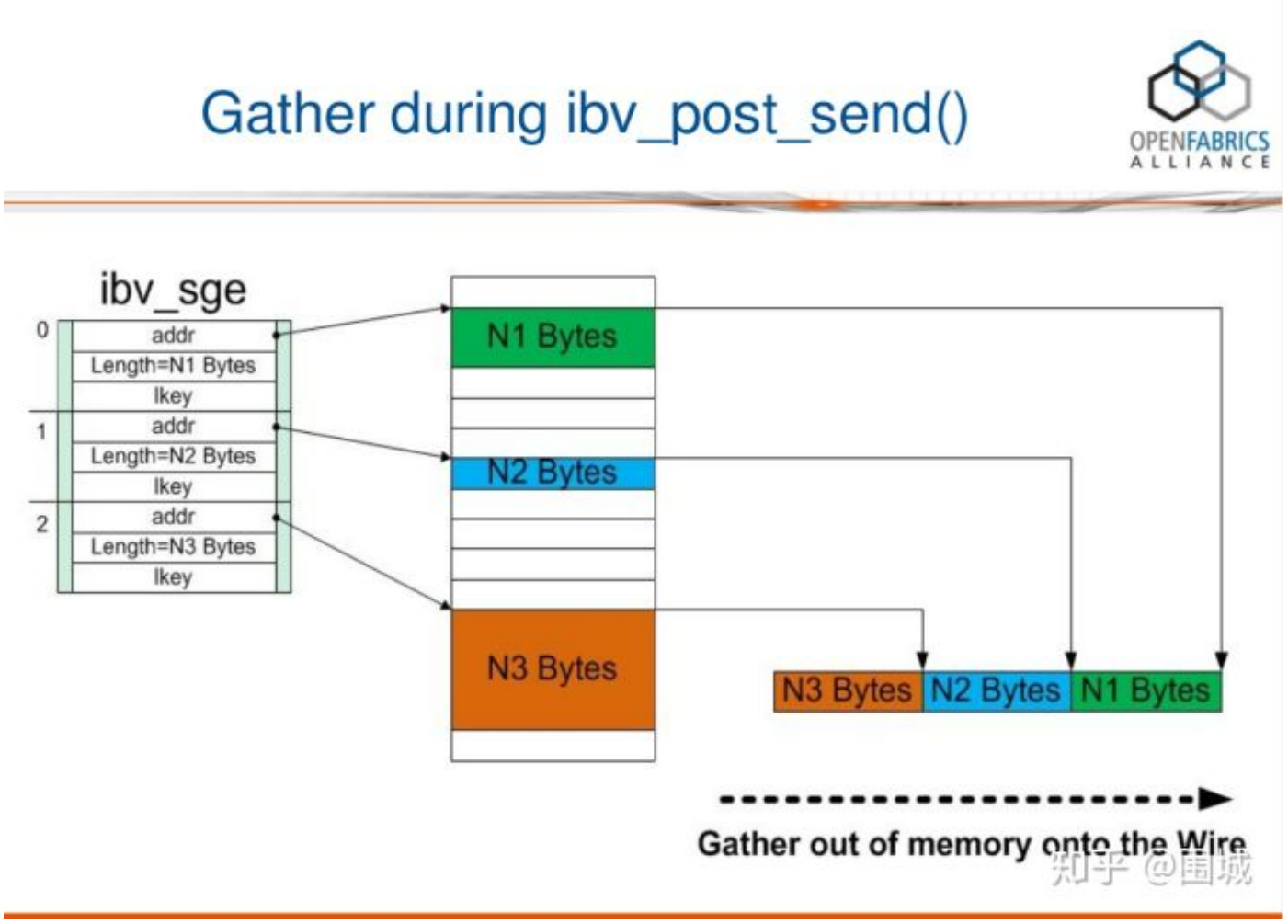
▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···



我们在发送一段内存地址的时候，我们需要将这段内存地址通过Memory Registration注册到RDMA中。也就是说注册到PD内存保护域当中。一个SGL至少被一个MR保护, 多个MR存在同一个PD中。如图所示一段内存MR可以保护多个SGE元素。

4.3 调用ibv_post_send()将SGL发送到wire上去

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···



在上图中，一个SGL数组包含了3个SGE, 长度分别为N1, N2, N3字节。我们可以看到，这3个buffer并不连续，它们Scatter(分散)在内存中的各个地方。RDMA硬件读取到SGL后，进行Gather(聚合)操作，于是在RDMA硬件的Wire上看到的就是N3+N2+N1个连续的字节。换句话说，通过使用SGL, 我们可以把分散(Scatter)在内存中的多个数据段(不连续)交给RDMA硬件去聚合(Gather)成连续的数据段。

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···

附录一： OFED Verbs

OFED Verbs

Transfer Posting **传输派送**	rdma_create_qp	ibv_post_recv ibv_post_send	rdma_destroy_qp
Transfer Completion **传输完成**	ibv_create_cq ibv_create_comp_channel	ibv_poll_cq ibv_wc_status_str ibv_req_notify_cq ibv_get_cq_event ibv_ack_cq_events	ibv_destroy_cq ibv_destroy_comp_channel
Memory Registration **内存注册**	ibv_alloc_pd ibv_reg_mr		ibv_dealloc_pd ibv_dereg_mr
Connection Management **连接管理**	rdma_create_id rdma_create_event_channel	rdma_resolve_addr rdma_resolve_route rdma_connect rdma_disconnect rdma_bind_addr rdma_listen rdma_get_cm_event rdma_ack_cm_event rdma_event_str rdma_accept rdma_reject rdma_migrate_id rdma_get_local_addr rdma_get_peer_addr	rdma_destroy_id rdma_destroy_event_channel
Misc **其他**		rdma_get_devices rdma_free_devices ibv_query_devices	
	Setup	**Use**	**Break-Down**

知乎 @围城

发布于 2019-01-17

文章被以下专栏收录

RDMA **RDMA**

RDMA(RemoteDirect Memory Access)技术全称远程直接内存访问，就是为了解决...

进入专栏

推荐阅读

RDMA技术详解（一）：RDMA概述

1. DMA和RDMA概念1.1 DMADMA(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...

围城

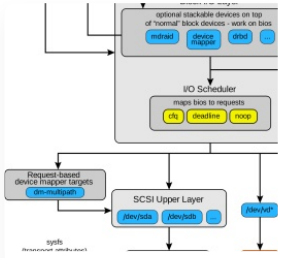
1000,000 packets/s的挑战

auxte... 发表于面向工资编...

RDMA技术详解（二）：RDMA Send Receive操作

1. 前言RDMA指的是远程直接内存访问，这是一种通过网络在两个应用程序之间搬运缓冲区里的数据的方法。RDMA与传统的网络接口不同，因为它绕过了操作系统。这允许实现了RDMA的程序具有如下...

围城 发表于RDMA



Linux 的IO栈

丁凯

1 条评论

▲ 赞同 2 ▼ 1 条评论 分享 收藏 ···

写下你的评论...

黑猫警长 4 个月前

请问发送同样大小的数据，发送连续的地址空间中的内容，和使用SGL发送不连续地址空间的内容，两者之间效率的差别是怎样的？

赞