AIX バージョン 7.2

Remote Direct Memory Access

AIX バージョン 7.2

Remote Direct Memory Access

お願い

本書および本書で紹介する製品をご使用になる前に、 21 ページの『特記事項』に記載されている情報をお読みください。

本書は AIX バージョン 7.2 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： AIX Version 7.2
Remote Direct Memory Access

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

目次

本書について

本書は、経験豊かな C プログラマー向けに、 AIX® オペレーティング・システム内の Internet Wide Area RDMA Protocol (iWARP) ファブリックまたは RDMA ネットワーク・インターフェース・コントローラー (RNIC) ファブリックでの Open Fabrics Enterprise Distribution (OFED) verb を使用したプログラミングに関して、詳細情報を提供します。

本書を効果的に使用するには、コマンド、システム・コール、サブルーチン、ファイル・フォーマット、および特殊ファイルに精通している必要があります。

強調表示

本書では、次の強調表示規則を使用しています。

太字	コマンド、サブルーチン、キーワード、ファイル、構造体、ディレクトリー、およびシステムによって名前が事前に定義されているその他の項目を表します。また、ユーザーが選択するボタン、ラベル、アイコンなどのグラフィック・オブジェクトも示します。
イタリック	ユーザーが実際の名前や値を指定するパラメーターを示します。
モノスペース	具体的なデータ値の例、表示される可能性があるテキストの例、プログラマーとして作成する可能性があるプログラム・コードの一部の例、システムからのメッセージ、またはユーザーが実際に入力する必要がある情報を示します。

AIX での大/小文字の区別

AIX オペレーティング・システムでは、すべて大文字小文字の区別をします。これは、英大文字と小文字を区別するということです。例えば、**ls** コマンドを使用するとファイルをリストできます。`LS` と入力すると、システムはそのコマンドが「`is not found`」と応答します。同様に、**FILEA**、**FiLea**、および **filea** は、同じディレクトリーにある場合でも、3 つの異なるファイル名です。予期しない処理が実行されないように、常に正しい大/小文字を使用するようにしてください。

ISO 9000

当製品の開発および製造には、ISO 9000 登録品質システムが使用されました。

Remote Direct Memory Access

経験豊かな C プログラマーは、Remote Direct Memory Access (RDMA) verb と Open Fabrics Enterprise Distribution (OFED) verb を使用した、AIX オペレーティング・システムでのプログラミングについて、詳細な情報を得ることができます。

情報を効果的に使用するには、コマンド、システム・コール、サブルーチン、ファイル・フォーマット、および特殊ファイルに精通している必要があります。

Remote Direct Memory Access の新規情報

Remote Direct Memory Access トピック・コレクションの新規情報または大幅な変更に関する情報をお読みください。

新規情報または変更情報の参照方法

この PDF ファイルでは、左余白に新規情報と変更情報を識別するリビジョン・バー (|) が記されている場合があります。

2017 年 10 月

以下の情報は、このトピック・コレクションに加えられた更新の要約です。

- Remote Direct Memory Access のトピック・コレクションに、Shared Memory Communications over RDMA (SMC-R) のトピックが追加されました。

Open Fabrics Enterprise Distribution (OFED)

AIX オペレーティング・システムで Open Fabrics Enterprise Distribution (OFED) verb のプログラミングを始める方法について説明します。OFED verb を使用すると、高いスループットと少ない遅延を必要とするアプリケーションで Remote Direct Memory Access (RDMA) 機能を使用できます。

関連概念:

19 ページの『SMC-R プロトコル・ソリューションの構成』
SMC-R プロトコル・ソリューションは、RoCE で OpenFabrics Enterprise Distribution (OFED™) コア・サービスを使用します。

OFED の概念

Open Fabrics Enterprise Distribution (OFED) verb の verb 層は、InfiniBand、RDMA over Converged Ethernet (RoCE)、Internet Wide Area RDMA Protocol (iWARP)、および InfiniBand アーキテクチャーから派生した verb と特に異なるものではありません。

ハードウェア要件

AIX オペレーティング・システムは、RDMA over Converged Ethernet (RoCE) アダプターをサポートしています。AIX で RoCE RDMA をサポートするハードウェアは、PCIe2 10 GbE RoCE アダプターと呼ばれます。

ソフトウェア要件

AIX OFED verb は、OpenFabrics Alliance の OFED 1.5 コードを基礎としています。AIX オペレーティング・システム上では、OFED コードの 32 ビットおよび 64 ビットのユーザー・アプリケーションがサポートされます。RDMA のインストールとともに、以下のライブラリーが提供されます。

- Librdmacm
- Libibverbs

verb API

AIX アプリケーションは、特定の宛先と通信する必要がある verb が Open Fabrics Enterprise Distribution (OFED) verb なのか、それとも AIX InfiniBand (IB) verb なのか、verb API を判別することができます。

次の疑似コードの例では、使用できる OFED verb を判別するために、必要なリモート・アドレスに対する **rdma_resolve_addr** コマンドの結果をテストします。

このプログラムは、以下の値を返します。

- **0**- OFED verb を使用して宛先との通信を確立できる場合。
- **error**- OFED をサポートするデバイスを介して宛先との通信を確立できず、InfiniBand アーキテクチャーを使用して確立できる場合。

```
/*The following check_ofed_verbs_support routine does:
/*- Call rdma_create_event_channel to open a channel event     */
/*- Calls rdma_create_id() to get a cm_id                        */
/*- And then calls rdma_resolve_addr()             */
/*- Get the communication event                 */
/*- Returns the event status:                  */
/* 0: OK                                 */
/* error: NOK  output device may be not a RNIC device     */
/*- Calls rdma_destroy_id() to delete the cm_id created     */
/*- Call rdma_destroy_event_channel  to close a channel event  */

 int check_ofed_verbs_support (struct sockaddr *remoteaddr)
   {
    struct rdma_event channel *cm_channel;
    struct rdma_cm_id *cm_id;
    int ret=0;
    cm_channel = rdma_create_event_channel();
    if (!cm_channel)  {
      fprintf(stderr,"rdma_create_event_channel error¥n");
      return -1;
      }
      ret = rdma_create_id(cm_channel, &cm_id, NULL, RDMA_PS_TCP);
      if (ret) {
        fprintf(stderr,"rdma_create_id: %d¥n", ret);
        rdma_destroy_event_channel(cm_channel);
        return(ret);
      }
      ret = rdma_resolve_addr(cm_id, NULL, remoteaddr, RESOLVE_TIMEOUT_MS);
      if (ret) {
        fprintf(stderr,"rdma_resolve_addr: %d¥n", ret);
        goto out;
      }
      ret = rdma_get_cm_event(cm_channel, &event);
      if (ret) {
        fprintf(stderr," rdma_get_cm_event() failed¥n");
        goto out;
      }
      ret = event->status;
      rdma_ack_cm_event(event);
```

```
        out:
    rdma_destroy_id(cm_id);
    rdma_destroy_event_channel(cm_channel);
    return(ret);
}
```

Libibverbs ライブラリー

Libibverbs ライブラリーを使用すると、ユーザー・スペース・プロセスで Remote Direct Memory Access (RDMA) verb を使用することができます。

Libibverbs ライブラリーについては、InfiniBand アーキテクチャーの仕様と RDMA プロトコル verb の仕様で説明されています。

Libibverbs ライブラリーと `ib_uverbs` カーネル層の間の通信を処理するために、いくつかの `/dev/rdma/uverbsN` キャラクター型デバイス・ノードが使用されます。すべての RDMA ネットワーク・インターフェース・コントローラー (RNIC) アダプターは、uverbs1 デバイスや uverbs2 デバイスなど、Open Fabrics Enterprise Distribution (OFED) コアに登録されている 1 つのデバイスを備えています。適切なデバイス上で稼働するために、ライブラリーは、verb に対応するコマンドを書き込みます。

関連情報:

- InfiniBand
- RDMA プロトコル verb

Librdmacm ライブラリー

librdmacm ライブラリーは、コミュニケーション・マネージャー (CM) 機能と、さまざまなファブリック (InfiniBand (IB)、RDMA over Converged Ethernet (RoCE)、または Internet Wide Area RDMA Protocol (iWARP) など) で稼働する汎用の Remote Direct Memory Access (RDMA) CM インターフェース・セットを提供します。

ユーザー・スペースとカーネルとの通信には、存在するアダプターやポートの数に関係なく、単一の `/dev/rdma/rdma_cm` デバイス・ノードが使用されます。

librdmacm ライブラリーは、すべての RDMA デバイス上で稼働する必要があるアプリケーションによって使用されます。

RDMA ネットワーク・インターフェース・コントローラー (RNIC)

Internet Wide Area RDMA Protocol (iWARP)、および verb 機能を備えたネットワーク入出力アダプターまたは組み込みコントローラー。

RDMA_CM コミュニケーション・マネージャー

Remote Direct Memory Access コミュニケーション・マネージャー (RDMA_CM) は、データを転送するための高信頼性接続をセットアップするために使用されます。

このコミュニケーション・マネージャーは、接続を確立するための RDMA トランスポート中立インターフェースを提供します。この API はソケットをベースにしていますが、キュー・ペア (QP) ベースのセマンティクスに適しています。通信は特定の RDMA デバイスを介して行われ、データ転送はメッセージがベースになります。

RDMA CM は **librdmacm** ライブラリーを使用して、RDMA API の接続をセットアップおよび破棄するための通信管理を提供します。このコミュニケーション・マネージャーは、データ転送に **libibverbs** ライブラリーを使用することにより、verb API と連動します。

OFED verb を使用して管理されるリソース

OFED verb を使用して管理されるリソースをリストします。

完了キュー **(CQ, Completion Queue):**
: 完了キュー (CQ) を含んでいる先入れ先出し (FIFO) キュー。CQ は、完了通知とイベントを受信するために使用されるキュー・ペアに関連付けられます。

完了キュー項目 **(CQE, Completion Queue Entry):**
: 完了した作業要求 (WR) に関する情報 (状況とサイズなど) を記述した、CQ 内の項目。

イベント・チャネル **(Event Channel):**
: 通信イベントを報告するために使用されます。各イベント・チャネルはファイル・ディスクリプターへマップされます。関連するファイル・ディスクリプターを他のファイル・ディスクリプターと同様に使用および操作して、その動作を変更することができます。以下のいずれかのアクションをファイル・ディスクリプターに実行させることができます。

  - ファイル・ディスクリプターの非ブロッキング
  - ファイル・ディスクリプターのポーリング
  - ファイル・ディスクリプターの選択

メモリー領域 **(MR、Memory Region):**
: アクセス許可付きで登録されている一連のメモリー・バッファー。ネットワーク・アダプターでメモリー・バッファーを使用するには、メモリー領域が登録されている必要があります。

保護ドメイン **(PD、Protection Domain):**
: クライアントがドメイン内でキュー・ペアやメモリー領域などの複数のリソースを関連付けることができるようにします。その後、クライアントは、保護ドメイン内のデータを RDMA ファブリック上に置かれている他のドメインとの間で送受信するアクセス権限を付与します。

キュー・ペア **(QP、Queue Pair):**
: キュー・ペア (QP) には、送信キューと受信キューが含まれています。送信キューは、RDMA 操作を要求するアウトバウンド・メッセージを送信します。受信キューは、着信メッセージまたは即値データを受信します。

分散または集結エレメント **(SGE、Scatter or Gather Elements):**
: ローカル側に登録されたメモリー・ブロックの全部または一部を指すポインターを表す項目。このエレメントはブロックの開始アドレス、サイズ、および関連する許可を持つ lkey を保持します。

分散または集結配列 **(Scatter or Gather Array):**
: 作業要求 (WR) 内に存在する分散エレメントまたは集結エレメントの配列。この配列は命令コードに従って作業します。命令コードは複数のバッファーからデータを収集して、それを単一のストリームとして送信するか、単一のストリームを取得して、データを多数のバッファーに分離します。

作業キュー **(WQ、Work Queue):**
: 作業キューは、送信キューまたは受信キューから構成されます。作業キューは、メッセージの送信または受信に使用されます。

作業キュー・エレメント **(WQE, Work Queue Element):**
: 作業キュー・エレメントとは、作業キュー内のエレメントのことです。

作業要求 **(WR、Work Request):**
作業要求とは、ユーザーによって作業キューへ記入された要求のことです。

通信操作

RDMA デバイスに使用可能な通信操作をリストします。

send 操作および **send with immediate** 操作**:**

send 操作は、リモート・キュー・ペア (QP) の受信キューにデータを送信します。

データを受信するには、受信側がデータを受信バッファーに記入する必要があります。送信側には、リモート・ホスト内にあるデータに対する制御権が一切ありません。

データ・バッファーとともに、4 バイトの即値が送信されます。この即値は受信通知の一部として受信側に提示され、データ・バッファーには含まれません。

receive 操作**:**

receive 操作は send 操作に対応する操作です。

受信側ホストは、データ・バッファーがインライン即値と一緒に受信されたことを通知されます。受信側アプリケーションは受信バッファーを維持し、情報を記入します。

RDMA read 操作**:**

RDMA read 操作は、リモート・ホストからメモリー領域を読み取ります。

ユーザーは、リモート仮想アドレスと、読み取った情報をコピーするローカル・メモリー・アドレスを指定する必要があります。Remote Direct Memory Access (RDMA) 操作を実行する前に、リモート・ホストは、そのメモリーにアクセスするための適切な許可を提供する必要があります。許可が設定された後、RDMA read 操作はリモート・ホストへの通知なしに実行されます。

atomic 操作**:**

AIX オペレーティング・システムに使用可能な Remote Direct Memory Access (RDMA) ハードウェアは、atomic 操作をサポートしていません。

RDMA write 操作または **RDMA write with immediate** 操作**:**

RDMA write 操作は RDMA read 操作によく似ていますが、データがリモート・ホストに書き込まれます。

RDMA write 操作は、リモート・ホストへの通知なしに実行されます。RDMA write with immediate 操作は、リモート・ホストに即値に関する通知を行います。

トランスポート・モード

トランスポート・モードは、キュー・ペアの接続を確立します。

以下のトランスポート・モードがサポートされています。

- 高信頼性接続 (RC、Reliable Connection)
  - 各キュー・ペア (QP) は別の QP と関連付けられます。

- 1 つの QP の送信キューによって送信されるメッセージは、別の QP の受信キューに高い信頼度で配信されます。
- パケットは順に配信されます。
- RC は TCP 接続によく似ています。

• 低信頼性データグラム (UD、Unreliable Datagram)
- QP 間で実際の接続は形成されません。
- UD モードは UDP 接続によく似ています。

Open Fabrics Enterprise Distribution (OFED) の計画

/etc/libibverbs.d/ ディレクトリーに、システムに取り付けられているすべての Remote Direct Memory Access (RDMA) アダプター用の構成ファイルが存在する必要があります。

構成ファイルにより、**libibverbs** ライブラリーは RDMA デバイスのドライバーを使用できるようになります。例えば、**Mellanox ConnectX-2 RoCE** アダプターを使用するには、mx2.driver ファイルが /etc/libibverbs.d/ ディレクトリーに存在する必要があります。mx2.driver ファイルには、次のコードが含まれている必要があります。

```
# cat /etc/libibverbs.d/mx2.driver
driver mx2
```

/etc/libibverbs.d/ ディレクトリー以外のディレクトリーを使用するには、IBV_CONFIG_DIR 環境変数を使用します。2 つのノード間の通信を確立するには、各アダプターに IPv4 アドレスまたは IPv6 アドレスが構成されている必要があります。

コミュニケーション・マネージャー (RDMA_CM) を使用した接続の作成

Remote Direct Memory Access (RDMA) RDMA_CM コミュニケーション・マネージャーは、RDMA アプリケーション・プログラミング・インターフェース (API) のための接続のセットアップと破棄を含む通信管理を提供します。

RDMA_CM コミュニケーション・マネージャーは、**libibverbs** ライブラリーによって定義された verb API と連動します。**libibverbs** ライブラリーは、データの送受信に必要なインターフェースを提供します。

クライアント操作

アクティブ通信またはクライアント通信の基本操作の概要を説明します。

一般的な接続フローを以下に示します。

rdma_create_event_channel
イベントを受信するチャネルを作成します。

rdma_create_id
概念がソケットによく似た rdma_cm_id ID を割り振ります。

rdma_resolve_addr
リモート・アドレスに到達するために、ローカル Remote Direct Memory Access (RDMA) デバイスを取得します。

rdma_get_cm_event
RDMA_CM_EVENT_ADDR_RESOLVED イベントを待ちます。

rdma_ack_cm_event
イベントを受信したことを確認します。

rdma_create_qp
通信のキュー・ペア (QP) を割り振ります。

rdma_resolve_route
リモート・アドレスへの経路を決定します。

rdma_get_cm_event
RDMA_CM_EVENT_ROUTE_RESOLVED イベントを待ちます。

rdma_ack_cm_event
イベントを受信したことを確認します。

rdma_connect
リモート・サーバーに接続します。

rdma_get_cm_event
RDMA_CM_EVENT_ESTABLISHED イベントを待ちます。

rdma_ack_cm_event
イベントを受信したことを確認します。

ibv_post_send()
接続を介してデータ転送を実行します。

rdma_disconnect
接続を破棄します。

rdma_get_cm_event
RDMA_CM_EVENT_DISCONNECTED イベントを待ちます。

rdma_ack_cm_event
イベントを確認します。

rdma_destroy_qp
QP を破棄します。

rdma_destroy_id
rdma_cm_id ID を解放します。

rdma_destroy_event_channel
イベント・チャネルを解放します。

注: 例では、クライアントが切断を開始しました。しかし、クライアント操作とサーバー操作のどちらでも切断プロセスを開始することができます。

サーバー操作

パッシブ通信またはサーバー通信用に実行できる基本操作について説明します。

一般的な接続フローを以下に示します。

rdma_create_event_channel
イベントを受信するチャネルを作成します。

rdma_create_id
概念がソケットによく似た rdma_cm_id ID を割り振ります。

rdma_bind_addr
イベントが listen するローカル・ポート番号を設定します。

rdma_listen
接続要求の listen を開始します。

rdma_get_cm_event
新しい rdma_cm_id ID を持つ RDMA_CM_EVENT_CONNECT_REQUEST イベントを待ちます。

rdma_create_qp
新しい rdma_cm_id ID での通信のキュー・ペアを割り振ります。

rdma_accept
接続要求を受け入れます。

rdma_ack_cm_event
イベントを確認します。

rdma_get_cm_event
RDMA_CM_EVENT_ESTABLISHED イベントを待ちます。

rdma_ack_cm_event
イベントを確認します。

ibv_post_send()
その接続を介したデータ転送を実行します。

rdma_get_cm_event
RDMA_CM_EVENT_DISCONNECTED イベントを待ちます。

rdma_ack_cm_event
イベントを確認します。

rdma_disconnect
接続を破棄します。

rdma_destroy_qp
QP を破棄します。

rdma_destroy_id
接続された **rdma_cm_id** ID を解放します。

rdma_destroy_id
listen している rdma_cm_id ID を解放します。

rdma_destroy_event_channel
イベント・チャネルを解放します。

RDMA_CM コミュニケーション・マネージャーの例

LinuxConf.Europe 2007 会議のときに Open Fabrics Enterprise Distribution (OFED) コミュニティーに提示された例について説明します。

関連情報:

OFED コミュニティーに提示された例

アクティブ・クライアントの例

クライアントがアクティブである場合の通信操作の例。

```
/*
 * build:
 * cc -o client client.c -lrdmacm -libverbs
 *
```

```
 * usage:
 * client <servername> <val1> <val2>
 *
 * connects to server, sends val1 via RDMA write and val2 via send,
 * and receives val1+val2 back from the server.
 */
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>

#include <rdma/rdma cma.h>
enum   {
        RESOLVE TIMEOUT MS        = 5000,
};
struct pdata {
        uint64_t buf va;
        uint32_t buf rkey;
};

int main(int argc, char   *argv[ ])
{
   struct pdata         *server pdata;
   struct rdma_event channel  *cm_channel;
   struct rdma_cm_id      *cm_id;
   struct rdma_cm_event     *event;
   struct rdma_conn_param    conn_param = { };
   struct ibv_pd        *pd;
   struct ibv_comp_channel   *comp_chan;
   struct ibv_cq        *cq;
   struct ibv_cq        *evt_cq;
   struct ibv_mr        *mr;
   struct ibv_qp_init_attr   qp attr = { };
   struct ibv_sge        sge;
   struct ibv_send_wr      send_wr = { };
   struct ibv_send_wr      *bad send wr;
   struct ibv_recv_wr      recv wr = { };
   struct ibv_recv_wr      *bad recv wr;
   struct ibv_wc        wc;
   void             *cq context;
   struct addrinfo       *res, *t;
   struct addrinfo       hints = { .ai_family    = AF INET,
                           .ai_socktype  = SOCK STREAM
                                       };
 int              n;
 uint32_t          *buf;
 int              err;

       /* Set up RDMA CM structures */
   cm_channel = rdma_create_event_channel();
   if (!cm_channel)  return 1;
   err = rdma_create_id(cm_channel, &cm_id, NULL, RDMA_PS_TCP);
   if (err)
     return err;
   n = getaddrinfo(argv[1], "20079", &hints, &res);
   if (n < 0)
     return   1;

   /* Resolve server address and route */
   for (t = res; t; t = t->ai next) {
     err = rdma_resolve_addr(cm_id, NULL, t->ai_addr, RESOLVE_TIMEOUT_MS);
     if (!err)
       break;
```

```
}
if (err)
  return err;
err = rdma_get_cm_event(cm_channel, &event);
if (err)
  return err;
if (event->event != RDMA_CM_EVENT_ADDR_RESOLVED)
  return 1;
rdma_ack_cm_event(event);
err = rdma_resolve_route(cm_id, RESOLVE_TIMEOUT_MS);
if (err)
  return err;
err = rdma_get_cm_event(cm_channel, &event);
if (err)
  return err;
if (event->event != RDMA_CM_EVENT_ROUTE_RESOLVED)
  return 1;
rdma_ack_cm_event(event);

/* Create verbs objects now that we know which device to use */
pd = ibv_alloc_pd(cm_id->verbs);
if (!pd)
  return 1;
comp chan = ibv_create_comp_channel(cm_id->verbs);
if (!comp_chan)
  return 1;
cq = ibv_create_cq(cm_id->verbs, 2,NULL, comp_chan, 0);
if (!cq)
  return 1;
 if (ibv_req_notify_cq(cq,  0))
  return 1;
 buf = calloc(2, sizeof (uint32_t));
if (!buf)
  return 1;
mr = ibv_reg_mr(pd, buf,2 * sizeof(uint32_t), IBV_ACCESS_LOCAL_ WRITE);
if (!mr)
  return 1;
qp_attr.cap.max        send_wr = 2;
qp_attr.cap.max        send_sge = 1;
qp_attr.cap.max        recv_wr = 1;
qp_attr.cap.max        recv_sge = 1;
qp_attr.send_cq        = cq;
qp_attr.recv_cq        = cq;
qp_attr.qp_type        = IBV_QPT_RC;
err = rdma_create_qp(cm_id, pd, &qp_attr);
if (err)
  return err;
conn_param.initiator_depth = 1;
conn_param.retry_count     = 7;

/* Connect to server */
err = rdma_connect(cm_id, &conn_param);
if (err)
  return err;
err = rdma_get_cm_event(cm_channel,&event);
if (err)
  return err;
if (event->event != RDMA_CM_EVENT_ESTABLISHED)
  return 1;
memcpy(&server_pdata, event->param.conn.private_data, sizeof server_pdata);
rdma_ack_cm_event(event);

/* Prepost receive */
sge.addr    =       (uintptr_t) buf;
sge.length  =       sizeof (uint32_t);
sge.lkey    =       mr->lkey;
recv_wr.wr_id =     0;
```

```
recv_wr.sg_list =   &sge;
recv_wr.num_sge =   1;

if (ibv_post_recv(cm_id->qp, &recv_wr, &bad_recv_wr))
  return 1;

/* Write/send two integers to be added */
buf[0] = strtoul(argv[2], NULL, 0);
buf[1] = strtoul(argv[3], NULL, 0);
printf("%d + %d = ", buf[0], buf[1]);
buf[0] = htonl(buf[0]);
buf[1] = htonl(buf[1]);

sge.addr                      = (uintptr_t) buf;
sge.length                    = sizeof (uint32_t);
sge.lkey                      =  mr->lkey;
send_wr.wr_id                 = 1;
sendwr.opcode                 = IBV_WR_RDMA_WRITE;
send _wr.sg_list              = &sge;
send_wr.num_sge               = 1;
send_wr.wr.rdma.rkey          = ntohl(server_pdata.buf_rkey);
send_wr.wr.rdma.remote_addr   = ntohll(server_pdata.buf_va);

  if (ibv post send(cm id->qp, &send_wr,  &bad_send_wr))
return 1;
sge.addr                      = (uintptr_t) buf + sizeof (uint32_t);
sge.length                    = sizeof (uint32_t);
sge.lkey                      = mr->lkey;
send_wr.wr_id                 = 2;
send_wr.opcode                = IBV_WR_SEND;
send_wr.send_flags            = IBV_SEND_SIGNALED;
send_wr.sg_list               =&sge;
send_wr.num_sge               = 1;

if (ibv_post_send(cm_id->qp, &send_wr,&bad_send_wr))
return 1;

/* Wait for receive completion */
while (1) {
   if (ibv_get_cq_event(comp_chan,&evt_cq, &cq_context))
  return 1;
if (ibv_req_notify_cq(cq, 0))
  return 1;
if (ibv_poll_cq(cq, 1, &wc) != 1)
  return 1;
if (wc.status != IBV_WC_SUCCESS)
  return 1;
if (wc.wr_id == 0) {
  printf("%d¥n", ntohl(buf[0]));
  return 0;
}
 }
 return 0;
}
```

OFED コマンド

構文ステートメント、フラグの説明、および使用例も含め、Open Fabrics Enterprise Distribution (OFED) コマンドについて説明します。

ibv_devices コマンド

ユーザー・スペースから使用可能な Remote Direct Memory Access (RDMA) デバイスをリストします。

ibv_devinfo コマンド

ユーザー・スペースから使用可能な RDMA ネットワーク・インターフェース・コントローラー (RNIC) デバイスに関する情報を出力します。

構文

```
ibv_devinfo [-v] { [-d <dev>] [-i <port>] } | [-l]
```

フラグ

項目	説明
-d *dev*	*dev* RDMA デバイスを使用します。デフォルトでは、最初に検出されたデバイスが使用されます。
-i *port*	RDMA デバイスの *port* ポートを使用します。デフォルトでは、すべてのポートが使用されます。
-l	RDMA デバイスの名前だけを出力します。
-v	RDMA デバイスのすべての属性を出力します。

ofedctrl コマンド

ofed_core カーネル・エクステンションをロードおよびアンロードします。

構文

ofedctrl { [**-k** *KernextName*] **-l|u|q** } | [**-c** | **-p** *ParameterName=Value*] | **-h**

フラグ

項目	説明
-c	構成ファイルが編集された場合は、そのファイルを再ロードします。
-h	使用法を示します。
-k *KernextName*	カーネル・エクステンションのパスを指定します。デフォルトでは、/usr/lib/drivers/ofed_core パスが使用されます。
-l	カーネル・エクステンションをロードします。
-p *ParameterName=Value*	パラメーターの値をコマンド・ラインで直接設定します。 **注:** -p オプションを使用して設定された値には、永続性がありません。-p オプションは、現行の構成のみを変更します。構成ファイルを更新するわけではありません。 -p オプションを使用して加えた変更は、システムの再始動後には適用されません。
-q	カーネル・エクステンションをロードするかどうかを指示します。
-u	カーネル・エクステンションをアンロードします。

rping コマンド

RDMA ping-pong テストを使用して、RDMA コミュニケーション・マネージャー (RDMA_CM) の接続をテストします。

構文

```
rping -s [-v] [-V] [-d] [-P] [-a address] [-p port] [-C message_count] [-S message_size]
rping -c [-v] [-V] [-d] -a address [-p port] [-C message_count] [-S message_size]
```

説明

rping コマンドは、**librdmacm** ライブラリーを使用して、2 つのノード間に信頼できる Remote Direct Memory Access (RDMA) 接続を確立します。また、オプションとして、**rping** コマンドはノード間の RDMA 転送を実行した後、接続を切断します。**rping** コマンドは、RDMA_CM 接続を設定し、RDMA ping-pong テストを実行します。**rping** コマンドについては、Open Source OpenFabrics Alliance OFED 1.4 (http://www.openfabrics.org) を参照してください。

フラグ

項目	説明
-a *address*	接続をバインドするサーバーのネットワーク・アドレスを指定し、クライアントに接続するためのサーバー・アドレスを指定します。
-c	クライアントとして実行します。
-C *message_count*	各接続で転送するメッセージの数を指定します。デフォルト値は無限です。
-d	デバッグ情報を表示します。
-p	listen するサーバーのポート番号を指定します。
-P	サーバーを永続モードで実行します。これにより、複数の **rping** クライアントが単一のサーバー・インスタンスに接続でき、サーバーはインスタンスが kill されるまで稼働します。
-v	ping データを表示します。
-V	ping データを検証します。
-s	サーバーとして実行します。
-S *message_size*	転送される各メッセージのサイズをバイト単位で指定します。デフォルト値は 100 です。

関連情報:

Openfabrics

uDAPL (ユーザー・レベルの Direct Access Programming Library)

uDAPL (ユーザー・レベルの Direct Access Programming Library) は、InfiniBand や RDMA ネットワーク・インターフェース・コントローラー (RNIC) など、直接データ・アクセスをサポートするトランスポートに対して実行される直接アクセス・フレームワークです。

DAT Collaborative は、uDAPL アプリケーション・プログラミング・インターフェース (API) を指定しています。Open Fabrics からの uDAPL コードベースは AIX オペレーティング・システムに移植され、GX++ HCA および 4X DDR 拡張カード (CFFh) InfiniBand アダプター上でサポートされています。

関連概念:

14 ページの『AIX オペレーティング・システムでサポートされる uDAPL API』
AIX オペレーティング・システムは、DAT Collaborative によって指定された uDAPL (ユーザー・レベルの Direct Access Programming Library) API をどれもサポートしていません。

15 ページの『uDAPL のベンダー固有の属性』
AIX オペレーティング・システムがサポートしているベンダー固有の属性について説明します。
`delayed_ack_supported`、`vendor_extension`、`vendor_ext_version`、`debug_query`、`debug_modify` の各属性がサポートされています。

関連情報:

Datcollaborative

uDAPL のインストール

AIX オペレーティング・システムでは、uDAPL (ユーザー・レベルの Direct Access Programming Library) バージョン 2.0 がサポートされています。

uDAPL インストール・イメージは、**udapl.rte** として拡張パックで出荷されています。このイメージは DAT ヘッダー・ファイルを提供し、それらのファイルは `/usr/include/dat` ディレクトリーに置かれています。また、インストール・イメージは **libdat.a** ライブラリーと **libdapl.a** ライブラリーを提供します。

アプリケーションは DAT ヘッダー・ファイルを含んでおり、/usr/include/dat ディレクトリー内の **libdat.a** DAT ライブラリーとリンクします。DAT 層は、基礎となる適切なトランスポート固有ライブラリーを決定します。

AIX uDAPL プロバイダーは dat.conf ファイルの項目を使用して、それ自体を DAT レジストリーに登録します。/etc/dat.conf ファイルは、デフォルトの項目を入れて出荷され、項目のフォーマットに関する詳細を含んでいます。

uDAPL ライブラリーは、イベントのデバッグのために AIX システム・トレースをサポートしています。uDAPL システム・トレースは、5C3 (DAPL イベントの場合)、5C4 (DAPL エラー・イベントの場合)、5C7 (DAT イベントの場合)、および 5C8 (DAT エラー・イベントの場合) を含んでいる ID に接続します。初期トレース・レベルを変更するには、環境変数 DAT_TRACE_LEVEL および DAPL_TRACE_LEVEL を使用します。これらの環境変数には、0 から 10 までの範囲の値を指定できます。イベントの数およびトレースされるデータの量は、以下のように、キー・トレース・レベルにつれて増加します。

```
TRC_LVL_ERROR  = 1
TRC_LVL_NORMAL = 3
TRC_LVL_DETAIL = 7
```

その他の標準的な AIX 保守容易性機能 (AIX エラー・ログなど) は、イベントのトレース時に問題を識別するために使用されます。基礎となるトランスポート層の保守容易性機能 (**ibstat** コマンドや InfiniBand コンポーネント・トレースなど) も問題の分析に役立ちます。

DAT API は、/usr/include/dat/dat_error.h ファイルを使用してデコードできる標準戻りコードを返します。戻りコードに関する詳しい説明は、DAT Collaborative からの uDAPL 仕様に示されています。

AIX オペレーティング・システムでサポートされる uDAPL API

AIX オペレーティング・システムは、DAT Collaborative によって指定された uDAPL (ユーザー・レベルの Direct Access Programming Library) API をどれもサポートしていません。

以下の API は、業界で一般的な uDAPL の実装によってサポートされ、AIX オペレーティング・システムにでもサポートされています。

以下の API は、業界で一般的な uDAPL の実装によってサポートされておらず、AIX オペレーティング・システムでもサポートされていません。

API	バージョン
dat_cr_handoff	// In DAT 2.0
dat_ep_create_with_srq	// In DAT 2.0
dat_ep_recv_query	// In DAT 2.0
dat_ep_set_watermark	// In DAT 2.0
dat_srq_create	// In DAT 2.0
dat_srq_post_recv	// In DAT 2.0
dat_srq_resize	// In DAT 2.0
dat_srq_set_lw	// In DAT 2.0
dat_srq_free	// In DAT 2.0
dat_srq_query	// In DAT 2.0

以下の追加 API は AIX オペレーティング・システムではサポートされていません。

- dat_lmr_sync_rdma_read
- dat_lmr_sync_rdma_write
- dat_registry_add_provider

- dat_registry_add_provider

サポートされないすべての API の場合、AIX オペレーティング・システムは、サポートされていない API リストを示すために DAT 仕様に記されている特定のメカニズムに従います。これには、ゼロである max_srq 属性値や特定の DAT_MODEL_NOT_SUPPORTED 戻りコードが含まれます。業界の実装および DAT 仕様によれば、DAT_NOT_IMPLEMENTED コードは、サポートされていない機能に対して返されることがあります。

リモート・メモリー領域 (RMR) に関連した API (*dat_rmr_create*、*dat_rmr_bind*、*dat_rmr_free*、*dat_rmr_query* など) のサポートは、基礎となるホスト・チャネル・アダプター (HCA) の能力によって異なり、成功するか失敗するかは基礎となる InfiniBand フレームワークによって決まります。現在、GX++ HCA および 4X DDR 拡張カード (CFFh) InfiniBand アダプターは、RMR 操作をサポートしていません。

関連概念:

13 ページの『uDAPL (ユーザー・レベルの Direct Access Programming Library)』
uDAPL (ユーザー・レベルの Direct Access Programming Library) は、InfiniBand や RDMA ネットワーク・インターフェース・コントローラー (RNIC) など、直接データ・アクセスをサポートするトランスポートに対して実行される直接アクセス・フレームワークです。

『uDAPL のベンダー固有の属性』
AIX オペレーティング・システムがサポートしているベンダー固有の属性について説明します。delayed_ack_supported、vendor_extension、vendor_ext_version、debug_query、debug_modify の各属性がサポートされています。

関連情報:


uDAPL: User Direct Access Programming Library

uDAPL のベンダー固有の属性

AIX オペレーティング・システムがサポートしているベンダー固有の属性について説明します。delayed_ack_supported、vendor_extension、vendor_ext_version、debug_query、debug_modify の各属性がサポートされています。

AIX オペレーティング・システムは、InfiniBand (IB) フレームワークのトランスポート・プロバイダーであり、ベンダー固有のインターフェース・アダプター (IA) と delayed_ack_supported 属性を含んでいます。delayed_ack_supported 属性の値は、**true** または **false** のいずれかです。値が **true** の場合、IA® に関連付けられたエンドポイントは、delayed_ack という、変更可能なプロバイダー固有の属性を持ちます。delayed_ack_supported 属性が **false** の場合、エンドポイントのプロバイダー固有 delayed_ack 属性を変更することはできません。エンドポイントのプロバイダー固有 delayed_ack 属性のデフォルト値は、**false** です。delayed_ack 属性を **true** に設定するには、dat_ep_modify オプションを使用します。このオプションは、エンドポイントに関連付けられている特定の InfiniBand キュー・ペアの基礎となる InfiniBand (IB) ホスト・チャネル・アダプター (HCA) の遅延確認機能を使用可能にします。このハードウェア機能は、すべての HCA によって実装されるわけではないため、すべての IA に使用できるわけではありません。この機能が使用可能な場合、HCA によって送信された確認応答は、サーバーのシステム・メモリー内でデータ転送操作が検出されるまで遅延されます。このプロセスにより、わずかながら遅延が増加します。

エラーのデバッグのために、uDAPL ライブラリーは AIX システム・トレースをサポートしています。初期トレース・レベルは、環境変数 DAT_TRACE_LEVEL および DAPL_TRACE_LEVEL を使用して変更できます。API を使用してこれらのトレース・レベルを動的に変更するには、AIX 上で動的トレース・レベル・サポートを使用します。ライブラリーに動的トレース・レベル・サポートがあるかどうかを確認するために、ア

プリケーションはベンダー固有の IA vendor_extension 属性の照会を行うことができます。vendor_extension 属性の存在は、動的トレース・レベルがサポートされていることを示します。vendor_extension 属性が存在する場合、アプリケーションはベンダー固有の IA 属性 debug_query および debug_modify の照会を行うことにより、dat_trclvl_query() および dat_trclvl_modify() の関数ポインターにアクセスできます。これらの属性の値は、対応する関数を指しています。この vendor_extension インターフェースを将来使用可能にするためには、vendor_extension というベンダー固有 IA 属性を使用する必要があります。現在、vendor_extension 属性は 1.0 に設定されており、これがサポートされている唯一のバージョンです。vendor_extension 属性が存在しない場合、アプリケーションは、トレース・レベルを動的に変更することはできません。

これらの属性を変更する方法の例は、AIX の実装と一緒にインストールされる uDAPL サンプル・コードに含まれています。

関連概念:

14 ページの『AIX オペレーティング・システムでサポートされる uDAPL API』
AIX オペレーティング・システムは、DAT Collaborative によって指定された uDAPL (ユーザー・レベルの Direct Access Programming Library) API をどれもサポートしていません。

13 ページの『uDAPL (ユーザー・レベルの Direct Access Programming Library)』
uDAPL (ユーザー・レベルの Direct Access Programming Library) は、InfiniBand や RDMA ネットワーク・インターフェース・コントローラー (RNIC) など、直接データ・アクセスをサポートするトランスポートに対して実行される直接アクセス・フレームワークです。

Shared Memory Communications over RDMA (SMC-R)

IBM® AIX 7.2 with Technology Level 2 から、AIX オペレーティング・システムは Shared Memory Communications over Remote Direct Memory Access (SMC-R) をサポートするようになりました。

SMC-R は、Sockets over RDMA および Internet Engineering Task Force (IETF) の Request for Comments (RFC) 7609 資料に基づく、プロトコル・ソリューションです。これは、IPv4 または IPv6 を介する伝送制御プロトコル (TCP) ソケットを使用することで、ソケット・アプリケーションに制限されます。 SMC-R ソリューションにより、TCP ソケット・アプリケーションは RDMA を透過的に使用でき、これによって直接、高速、低レイテンシー、およびメモリー間 (ピアツーピア) の通信が可能になります。

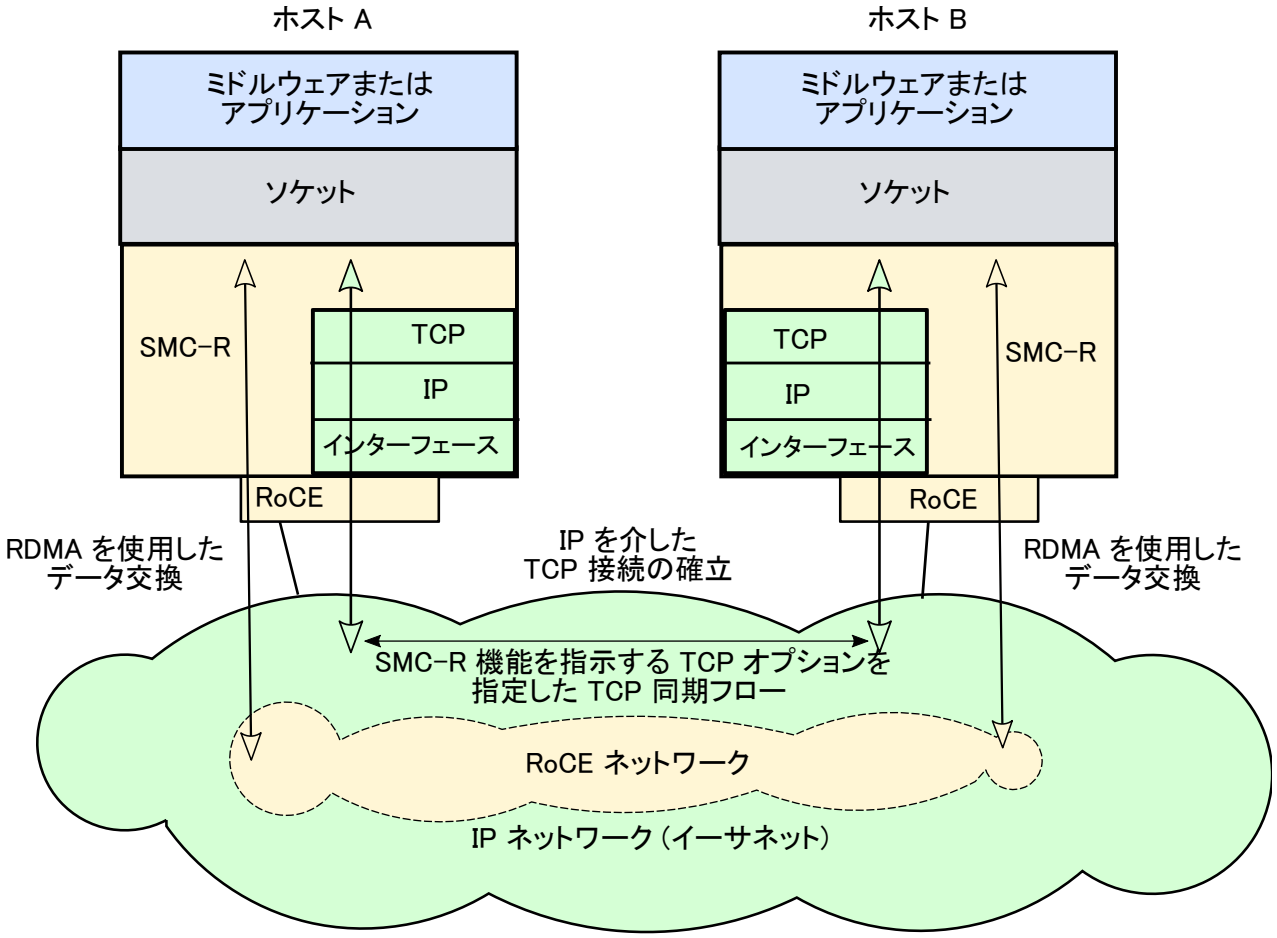
TCP/IP スタックなどのピアの通信では、従来の TCP/IP 接続確立フローを使用して、共有メモリー機能について動的に学習します。このプロセスにより、TCP/IP スタックは、TCP/IP ネットワーク・フローを、RDMA を使用する最適化された直接メモリー・アクセス・フローに切り替えることができます。

RDMA は、RDMA over Converged Ethernet (RoCE) インターフェースを使用して、標準イーサネット・ベース・ネットワーク上で使用できます。 RoCE ネットワーク・プロトコルは、InfiniBand Trade Association が提供する、業界標準イニシアチブです。 RoCE インターフェースにより、同じ物理ローカル・エリア・ネットワーク (LAN) ファブリック上の SMC-R などの、標準的な TCP/IP ソリューションと RDMA ソリューションの両方を使用できます。

SMC-R プロトコル・ソリューションにより、スループットを向上させ、待ち時間とコストを削減し、既存の機能を維持します。このモデルは、TCP/IP プロトコルの、多様で重要な既存の操作機能およびネットワーク管理機能を保持します。

SMC-R プロトコル・ソリューションは、両方の通信エンドポイントにまたがって複数の RoCE インターフェースが構成されている場合に、フェイルオーバー機能とロード・バランシング機能を提供します。

以下の図は、2 つのホスト間の SMC-R 通信フローを示しています。 TCP オプションを使用することで、TCP 同期操作は、両方のホストが SMC-R プロトコル・ソリューションをサポートしているかどうか、および RoCE ネットワークを確立するかどうかを判別します。



SMC-R プロトコル・ソリューションは、以下のような特性を示すことができるハイブリッド・ソリューションです。

- SMC-R プロトコル・ソリューションは、TCP 接続 (3 ウェイ・ハンドシェーク) を使用して、SMC-R 接続を確立します。
- 各 TCP エンドポイントは、そのエンドポイントが SMC-R プロトコルをサポートするかどうかを示す TCP オプションを交換します。高信頼の接続キュー・ペア (RC QP) 属性に関する SMC-R ランデブー情報は、SSL ハンドシェークに似ている TCP データ・ストリーム内で交換されます。
- RC QP 接続が確立されると、アプリケーション・データは RDMA 書き込み操作により交換されます。
- TCP 接続は、RC QP 接続とともにアクティブな状態を維持します。

SMC-R プロトコル・ソリューションは、RoCE で OpenFabrics Enterprise Distribution (OFED™) コア・サービスを使用します。また、データの転送に RC QP を使用します。

関連情報:

- RFC 情報: IBM's Shared Memory Communications over RDMA (SMC-R) Protocol
- Shared Memory Communications Reference Information

Shared Memory Communications の概念

Shared Memory Communications over Remote Direct Memory Access (SMC-R) ソリューションは、直接、高速、低レイテンシー、およびメモリー間 (ピアツーピア) の通信を提供します。

以下のリストは、SMC-R に関連する情報の中で使用される用語について説明しています。

RDMA over Converged Ethernet (RoCE)
: Remote Direct Memory Access over Converged Ethernet を使用可能にする InfiniBand Trade Association (IBTA) 標準。 RoCE により、同じイーサネット・ブロードキャスト・ドメイン内の任意の 2 つのホスト間での通信が可能になります。

高信頼の接続キュー・ペア **(RC QP)**
: 指定されたホストのペア間での RDMA 通信を有効にする、2 つのホスト間または論理区画間の論理接続。

Remote Direct Memory Access (RDMA)
: データが、リモート・ホスト・プロセッサーまたはオペレーティング・システムの介入なしでリモート・ホストのメモリーに直接転送される、高速で低レイテンシーのネットワーク通信プロトコル。

ランデブー処理
: 2 つのホスト間の SMC-R 通信を確立するために必要な TCP 接続管理フローのシーケンス。

SMC-R プロトコル・ソリューションの利点

ここでは、Shared Memory Communications over RDMA (SMC-R) のいくつかの利点を中心に説明しています。

SMC-R プロトコル・ソリューションには、以下のような多くの利点があります。

- SMC-R ネットワーク管理モデルは、現行の伝送制御プロトコル/インターネット・プロトコル (TCP/IP) ネットワーク管理に似ています。 SMC-R ネットワーク管理モデルは、TCP/IP ネットワーク管理モデルに基づいて作成されます。 SMC-R モデルを使用することで、AIX オペレーティング・システムは TCP/IP 通信用の類似のネットワーク管理モデルを提供します。例えば、TCP/IP モデルは、既存の IP トポロジーと IP アドレッシングを使用して、ネットワーク内の論理区画およびドメイン・ネーム・システム (DNS) リソースが (ホスト名から IP アドレスに) 変更されていないことを特定します。
- SMC-R プロトコル・ソリューションは、その使用は透過的です。つまり、同じサブネット内の SMC-R 対応オペレーティング・システムで実行するすべてのアプリケーションとミドルウェアは、SMC-R プロトコル・ソリューションのパフォーマンス面での利点の恩恵を、自動的に受けることになります。 SMC-R プロトコル・ソリューションを使用するために、アプリケーションを変更して値を派生させる必要はありません。
- トランザクション・ワークロードとストリーミング・ワークロードのパフォーマンスは、SMC-R プロトコル・ソリューションを使用して向上させることができます。 SMC-R プロトコル・ソリューションを使用することで、トランザクション・ワークロードはその総合的なトランザクション速度 (つまり 1 秒あたりのトランザクション数) を引き上げることができます。これにより CPU リソースは節約されます。ファイル転送プロトコル (FTP) などのストリーミング・ワークロードは、CPU リソースを保存して、そのスループットを向上させることができます。

注:

- 存続時間が短い TCP 接続の場合、SMC-R プロトコル・ソリューションはお勧めしません。

- SMC-R プロトコル・ソリューションを使用する場合、両方の TCP エンドポイントが同じレイヤー 2 ネットワーク (同じサブネット) に存在している必要があります。したがって、サーバーの IP アドレスは、クライアントの IP アドレスと同じ IP サブネット (または IPv6 の場合は接頭部) になければなりません。 SMC-R プロトコル・ソリューションはルーティング可能でないため、これらの条件は必須です。
- AIX オペレーティング・システム上での SMC-R プロトコル・ソリューションの実装は、VLAN 非認識モードのみをサポートします。

SMC-R プロトコル・ソリューションの構成

SMC-R プロトコル・ソリューションは、RoCE で OpenFabrics Enterprise Distribution (OFED™) コア・サービスを使用します。

SMC-R プロトコル・ソリューションを使用するには、`ofed.smcr` ファイルセットがインストールされていることが必要です。 `ofed.smcr` ファイルセットは、**installp** コマンドまたは SMIT インターフェースを使用することでインストールできます。

`ofed.smcr` ファイルセットをインストールしたら、SMC-R 機能はデフォルトでは無効になります。
SMC-R 機能を有効にするには、以下のステップを実行します。

- SMC-R モジュールをロードするには、以下のコマンドを入力します。

  ```
  mkdev -c tcpip -t smcr
  ```

- SMC-R モジュールの現在の構成設定をリストするには、以下のコマンドを入力します。

  ```
  lsattr -E -l smcr0
  ```

- 構成パラメーターを変更するには、以下のコマンドを入力します。

  ```
  chdev -l smcr0 -a <attribute_name>=<attribute_value>
  ```

- SMC-R モジュールをアンロードするには、以下のコマンドを入力します。

  ```
  rmdev -l smcr0
  ```

注**:** SMC-R 通信が進行中である場合には、SMC-R モジュールをアンロードできません。 SMC-R モジュールをアンロードする前に、ワークロードに関連しているすべての SMC-R 通信が終了していることを確認します。

SMC-R デバイスには、以下の属性がある場合があります。

enabled
: SMC-R 機能を有効または無効にします。この属性には、以下の値を指定できます。
  - 1 - SMC-R 機能を有効にします。この属性を 1 に設定すると、それ以降の一致するすべての接続は、SMC-R プロトコル・ソリューションを使用します。
  - 0 - SMC-R 機能を無効にします。この値はデフォルト値です。

ip_addr_list
: SMC-R ソリューションに使用されるインターフェースの IP アドレスを指定します。サポートされるインターフェースの最大数は 2 です。 IP アドレスを指定しない場合、接続では SMC-R プロトコル・ソリューションは使用されません。

max_memory
: SMC-R の操作に使用できる最大メモリーをメガバイト (MB) 単位で指定します。

port_range_list
: SMC-R モジュールを使用する必要があるアプリケーション・サーバー・ポートまたはポート範囲

を指定します。例えば、ポート範囲は 1 から 20、または 23、または 50 です。ポート情報を指定しない場合、接続では SMC-R プロトコル・ソリューションは使用されません。

関連情報:

1 ページの『Open Fabrics Enterprise Distribution (OFED)』
AIX オペレーティング・システムで Open Fabrics Enterprise Distribution (OFED) verb のプログラミングを始める方法について説明します。OFED verb を使用すると、高いスループットと少ない遅延を必要とするアプリケーションで Remote Direct Memory Access (RDMA) 機能を使用できます。

installp コマンド

SMC-R 統計情報

Shared Memory Communications over RDMA (SMC-R) プロトコル統計情報は、**entstat** コマンドおよび **netstat** コマンドを使用して派生させることができます。

entstat コマンドおよび **netstat** コマンドを使用して、SMC-R 統計情報を表示します。

entstat

entstat smcr0 コマンドをオプションを指定せずに実行すると、基本統計が表示されます。 **-d** オプションを使用すると、RDMA over Converged Ethernet (RoCE) インターフェース上で送信される SMC-R 通信に関する詳細統計を表示できます。さらに、**-r** オプションを使用すると、統計カウンターをリセットできます。

関連情報:

entstat コマンド

netstat コマンド

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒*103-8510*
東京都中央区日本橋箱崎町*19*番*21*号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

記載されている性能データとお客様事例は、例として示す目的でのみ提供されています。実際の結果は特定の構成や稼働条件によって異なります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年).

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

© Copyright IBM Corp. _年を入れる_.

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。 IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie などの各種テクノロジーの使用について詳しくは、『IBM オンラインでのプライバシー・ステートメントのハイライト』(http://www.ibm.com/privacy/jp/ja/)、『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』というタイトルのセクション、および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

INFINIBAND、InfiniBand Trade Association、および INFINIBAND デザイン・マークは、INFINIBAND Trade Association の商標またはサービス・マークです。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[カ行]

[タ行]

L

O

R

S

U

V

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21