Cisco UCS C シリーズ ラックマウント スタンドアロン サーバの Cisco usNIC 導入ガイド

初版：2018 年 8 月 14 日

最終更新：2019 年 1 月 11 日

Cisco usNIC の概要

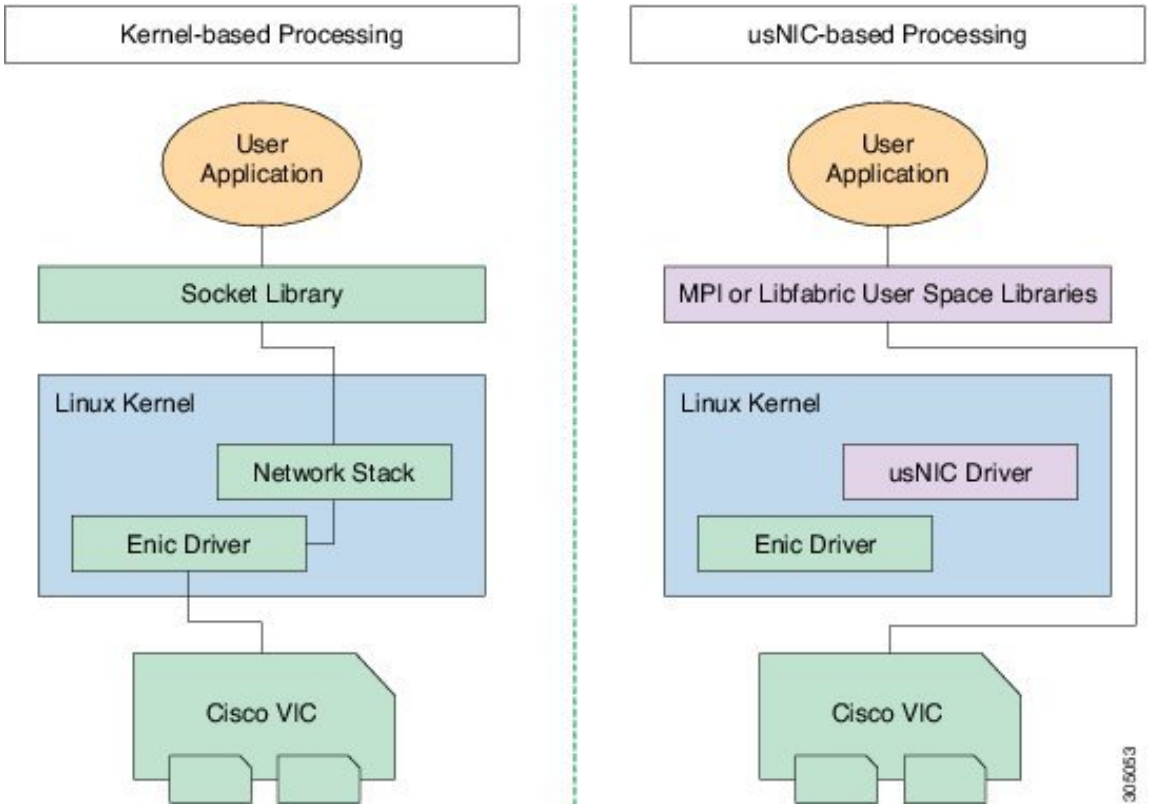
Cisco user-space NIC（Cisco usNIC）機能は、ネットワーキング パケットを送受信するときにカーネルをバイパスすることで、データセンターの Cisco UCS サーバで実行されるソフトウェア アプリケーションのパフォーマンスを改善します。アプリケーションは、第二世代以降の Cisco UCS VIC アダプタと直接やり取りすることによって、ハイ パフォーマンス コンピューティング クラスタのネットワーキング パフォーマンスが向上します。Cisco usNIC のメリットを引き出すためには、アプリケーションはソケットまたはその他の通信 API ではなく、Message Passing Interface（MPI）または Libfabric インターフェイスを使用する必要があります。

Cisco usNIC を使用すると、アプリケーションで次の利点が得られます。

- 低遅延で、高スループットの通信転送を提供します。
- 標準のアプリケーション非依存イーサネット プロトコルを実行します。
- 低ジッター、またはほぼ一定の遅延の通信。
- 次に示すシスコ データセンター プラットフォームで、低遅延の転送、ユニファイド ファブリック、統合管理のサポートを活用します。
  - Cisco UCS サーバ
  - Cisco UCS 第二世代以降の VIC アダプタ

標準イーサネットアプリケーションは、Linux カーネルのネットワーキング スタックを呼び出すユーザ領域のソケット ライブラリを使用します。次に、ネットワーキング スタックは Cisco eNIC ドライバを使用して、Cisco VIC ハードウェアと通信します。次の図は、通常のソフトウェア アプリケーションと usNIC を使用する MPI アプリケーションの対比を示します。

図 *1:* カーネル ベースのネットワーク通信と *Cisco usNIC* ベースの通信



Cisco usNIC の前提条件

Cisco usNIC から恩恵を受けるためには、設定に次の前提条件があります。

- サポートされている Linux オペレーティング システムディストリビューション リリース。サポートされている Linux オペレーティング システムのリリースの詳細についてUCS ハードウェアとソフトウェアの互換性ツールを参照してください。
- ICS ハードウェアとソフトウェア互換性ツールとして識別されたサーバインストールされたUCS サーバ モデル、Linux オペレーティング システム、およびサーバにインストールされたUCS ファーブウェアのバージョンに対応する UCS ドライバ ISO。詳細については、『Downloading Cisco UCS VIC drivers』を参照してください。
- IBM Sprectrum MPI、オープンソースの Community Open MPI パッケージ、またはバージョン 4 または 5 の Intel® MPI ライブラリなど、サポートされている MPI の実装。Intel ® MPI ライブラリが使用されている場合は、ネットワークでフロー制御を有効に設定する必要があります。

Cisco usNIC の設定

Cisco usNIC を設定する全体のフローは次のとおりです。

- usNIC をサポートする Cisco UCS VIC vNIC 設定の作成または変更
- サポートされている Linux OS のインストール (まだインストールされていない場合)
- Linux および usNIC をサポートする OS を設定します
- usNIC ドライバおよびユーティリティをインストールします
- libfabric と MPI ソフトウェアをインストールします
- usNIC のインストールの確認

（注） UCS C125 M5 ラック サーバ ノードは、Cisco usNIC ではサポートされていません。

CIMC GUI を使用した Cisco usNIC の設定

（注） [usNIC] セクションで Cisco usNIC に対していくつかのプロパティがリストされている場合、シスコはデフォルト値を使用して、**[usNIC]** フィールドおよび**サービス クラス**のフィールドのみを変更っすることを推奨します。

始める前に

このタスクを実行するには、管理者権限で CIMC GUI にログインする必要があります。

手順

ステップ 1 CIMC GUI にログインします。

CIMC へのログイン方法に関する詳細については、『Cisco UCS C-Series Servers Integrated Management Controller GUI Configuration Guide』を参照してください。

ステップ 2 ページの左上隅の [ナビゲーションの切り替え (Toggle Navigation)] アイコンをクリックします。

ステップ 3 [ネットワーク (Networking)] をクリックします。

ステップ 4 ネットワークでは、Cisco VIC アダプターを選択します。サーバの電源がオンになっている場合、アダプターの情報が [全般 (General)] タブに表示されます。

ステップ 5 [vNIC] タブをクリックします。

ステップ 6 左側のペインの [vNIC] で、イーサネット インターフェイスを選択します。

ステップ 7 右側のペインの端にある usNIC を展開します。

ステップ 8 [usNIC] ボックスで、作成する Cisco usNIC の数を指定します。

サーバで実行されている各 MPI プロセスには、専用の usNIC が必要です。64 の MPI プロセスを同時に実行させるには、最大 64 の usNIC を作成する必要がある場合があります。サーバに

ある物理的なコアと同じ数に対応する数の usNIC を作成することを推奨します。たとえば、サーバに 8 つの物理コアがある場合は、8 つの usNIC を作成します。

ステップ 9 変更を保存します。

ステップ 10 [ナビゲーションの切り替え (Toggle Navigation)] アイコンをクリックします。

ステップ 11 [コンピューティング (Compute)] をクリックします。

ステップ 12 [BIOS] タブでは、BIOS 設定、I/O で、次のプロパティを有効に設定します。

- Intel VT for directed IO
- Intel VTD ATS support
- Intel VTD coherency support

ステップ 13 [Save] をクリックします。変更内容は次のサーバのリブート時に有効になります。

CIMC CLI を使用した Cisco usNIC の作成

（注） usNIC セクションで Cisco usNIC のいくつかのプロパティが一覧表示されているにもかかわらず、Cisco はデフォルト値を使用しており、**usNIC**および**サービス クラス**のみを変更することを推奨します。

始める前に

このタスクを実行するには、管理者権限で CIMC CLI にログインする必要があります。

手順

	コマンドまたはアクション	目的
ステップ 1	server# **scope chassis**	シャーシ コマンド モードを開始します。
ステップ 2	server/chassis# **scope adapter** *index*	*index* で指定した PCI スロット番号に装着されているアダプタ カードに対してコマンド モードを開始します。 （注） アダプタの設定を表示または変更する前に、サーバの電源がオンであることを確認します。サーバに設定されたアダプタのインデックスを表示するには、**show adapter** コマンドを使用します。

	コマンドまたはアクション	目的
ステップ 3	server/chassis/adapter# **scope host-eth-if** {**eth0** | **eth1**}	vNIC のコマンドモードを開始します。お客様の環境に設定された vNIC の数に基づいてイーサネット ID を指定します。たとえば、1 個の vNIC のみを設定した場合、**eth0** を指定します。
ステップ 4	server/chassis/adapter/host-eth-if# **create usnic-config 0**	usNIC config を作成します。続いて、コマンド モードを開始します。インデックス値を必ず 0 に設定してください。 （注） CLI を使用して特定の vNIC に初めて Cisco usNIC を作成するには、**usnic-config** を最初に作成する必要があります。その後、**usnic-config** にスコープして、Cisco usNIC のプロパティを変更するだけで十分です。Cisco usNIC プロパティの変更の詳細については、「*CLI* を使用した *Cisco usNIC* 値の変更」を参照してください。
ステップ 5	server/chassis/adapter/host-eth-if/usnic-config# **set usnic-count** *number of usNICs* .	作成する Cisco usNIC の数を指定します。サーバで実行されている各 MPI プロセスには、専用の Cisco usNIC が必要です。したがって、64 の MPI プロセスを同時に実行させるには、最大 64 の Cisco usNIC を作成する必要がある場合があります。Cisco usNIC 対応 vNIC ごとに、サーバの物理コアの数と同数の Cisco usNIC を最低限作成することを推奨します。たとえば、サーバに 8 つの物理コアがある場合は、8 つの Cisco usNIC を作成します。
ステップ 6	server/chassis/adapter/host-eth-if /usnic-config# **commit**	トランザクションをシステムの設定にコミットします。 （注） 変更はサーバのリブート時に有効になります。
ステップ 7	server/chassis/adapter/host-eth-if/usnic-config# **exit**	ホストイーサネットインターフェイス コマンド モードを終了します。

	コマンドまたはアクション	目的
ステップ 8	server/chassis/adapter/host-eth-if# **exit**	アダプタ インターフェイス コマンド モードを終了します。
ステップ 9	server/chassis/adapter# **exit**	シャーシ インターフェイス コマンド モードを終了します。
ステップ 10	server/chassis# **exit**	サーバインターフェイスコマンドモードを終了します。
ステップ 11	server# **scope bios**	Bios コマンド モードを開始します。
ステップ 12	server/bios# **scope input-output**	入出力ページの BIOS セットアップ パラメータを入力します。
ステップ 13	server/bios/advanced# **set IntelVTD Enabled**	インテルバーチャライゼーションテクノロジーをイネーブルにします。
ステップ 14	server/bios/advanced# **set ATS Enabled**	プロセッサの Intel VT-d Address Translation Services（ATS）のサポートをイネーブルにします。
ステップ 15	server/bios/advanced# **set CoherencySupport Enabled**	プロセッサの Intel VT-d coherency のサポートをイネーブルにします。
ステップ 16	server /bios/advanced# **commit**	トランザクションをシステムの設定にコミットします。 （注） 変更はサーバのリブート時に有効になります。

例

次の例は、Cisco usNIC プロパティの設定方法を示します。

```
Server # scope chassis
server /chassis # show adapter
server /chassis # scope adapter 2
server /chassis/adapter # scope host-eth-if eth0
server /chassis/adapter/host-eth-if # create usnic-config 0
server /chassis/adapter/host-eth-if/usnic-config *# set usnic-count 64
server /chassis/adapter/host-eth-if/usnic-config *# commit
Committed settings will take effect upon the next server reset
server /chassis/adapter/host-eth-if/usnic-config # exit
server /chassis/adapter/host-eth-if # exit
server /chassis/adapter # exit
server /chassis # exit
server # exit
server# scope bios
server /bios # scope input-output
server /bios/advanced # set IntelVTD Enabled
```

```
server /bios/advanced *# set ATS Enabled*
server /bios/advanced *# set CoherencySupport Enabled
server /bios/advanced *# commit
Changes to BIOS set-up parameters will require a reboot.
Do you want to reboot the system?[y|N]y
A system reboot has been initiated.
```

CIMC CLI を使用した Cisco usNIC の値の変更

始める前に

このタスクを実行するには、管理者権限で CIMC GUI にログインする必要があります。

手順

	コマンドまたはアクション	目的
ステップ 1	server# **scope chassis**	シャーシ コマンド モードを開始します。
ステップ 2	server/chassis# **scope adapter** *index*	*index* で指定した PCI スロット番号に装着されているアダプタ カードに対してコマンド モードを開始します。 （注） アダプタの設定を表示または変更する前に、サーバの電源がオンであることを確認します。サーバに設定されたアダプタのインデックスを表示するには、**show adapter** コマンドを使用します。
ステップ 3	server/chassis/adapter# **scope host-eth-if** {**eth0** | **eth1**}	vNIC のコマンドモードを開始します。お客様の環境に設定された vNIC の数に基づいてイーサネット ID を指定します。たとえば、1 個の vNIC のみを設定した場合、**eth0** を指定します。
ステップ 4	server/chassis/adapter/host-eth-if# **scope usnic-config 0**	usNIC のコマンド モードを開始します。Cisco usNIC を設定する場合は、インデックス値を必ず 0 に設定してください。
ステップ 5	server/chassis/adapter/host-eth-if/usnic-config# **set usnic-count** *number of usNICs* .	作成する Cisco usNIC の数を指定します。サーバで実行されている各 MPI プロセスには、専用の Cisco usNIC が必要です。したがって、64 の MPI プロセスを同時に実行させるには、最大 64 の Cisco usNIC を作成する必要がある場合

	コマンドまたはアクション	目的
		があります。Cisco usNIC 対応 vNIC ごとに、サーバの物理コアの数と同数の Cisco usNIC を最低限作成することを推奨します。たとえば、サーバに 8 つの物理コアがある場合は、8 つの usNIC を作成します。
ステップ 6	server /chassis/adapter/host-eth-if /usnic-config# **commit**	トランザクションをシステムの設定にコミットします。 （注） 変更はサーバのリブート時に有効になります。
ステップ 7	server/chassis/adapter/host-eth-if/usnic-config# **exit**	ホストイーサネットインターフェイス コマンド モードを終了します。
ステップ 8	server/chassis/adapter/host-eth-if# **exit**	アダプタ インターフェイス コマンド モードを終了します。
ステップ 9	server/chassis/adapter# **exit**	シャーシ インターフェイス コマンド モードを終了します。
ステップ 10	server/chassis# **exit**	サーバインターフェイス コマンドモードを終了します。

例

次の例は、Cisco usNIC プロパティの設定方法を示します。

```
server # scope chassis
server /chassis # show adapter
server /chassis # scope adapter 2
server /chassis/adapter # scope host-eth-if eth0
server /chassis/adapter/host-eth-if # scope usnic-config 0
server /chassis/adapter/host-eth-if/usnic-config # set usnic-count 32
server /chassis/adapter/host-eth-if/usnic-config # commit
Committed settings will take effect upon the next server reset
server /chassis/adapter/host-eth-if/usnic-config # exit
server /chassis/adapter/host-eth-if # exit
server /chassis/adapter # exit
server /chassis # exit
server # exit
```

vNIC からの Cisco usNIC の削除

始める前に

このタスクを実行するには、admin 権限で CLI にログインする必要があります。

手順

	コマンドまたはアクション	目的
ステップ 1	server# **scope chassis**	シャーシ コマンド モードを開始します。
ステップ 2	server/chassis# **scope adapter** *index*	*index* で指定した PCI スロット番号に装着されているアダプタ カードに対してコマンド モードを開始します。 （注） アダプタの設定を表示または変更する前に、サーバの電源がオンであることを確認します。サーバに設定されたアダプタのインデックスを表示するには、**show adapter** コマンドを使用します。
ステップ 3	server/chassis/adapter# **scope host-eth-if** {**eth0** | **eth1**}	vNIC のコマンド モードを開始します。お客様の環境に設定された vNIC の数に基づいてイーサネット ID を指定します。たとえば、1 個の vNIC のみを設定した場合、**eth0** を指定します。
ステップ 4	Server/chassis/adapter/host-eth-if# **delete usnic-config 0**	vNIC の Cisco usNIC 設定を削除します。
ステップ 5	Server/chassis/adapter/host-eth-if# **commit**	トランザクションをシステムの設定にコミットします。 （注） 変更はサーバのリブート時に有効になります。

例

次に、vNIC の Cisco usNIC 設定を削除する例を示します。

```
server # scope chassis
server/chassis # show adapter
server/chassis # scope adapter 1
server/chassis/adapter # scope host-eth-if eth0
server/chassis/adapter/host-eth-if # delete usnic-config 0
server/chassis/host-eth-if/iscsi-boot *# commit
New host-eth-if settings will take effect upon the next adapter reboot

server/chassis/host-eth-if/usnic-config #
```

Linux カーネルの設定 Cisco usNIC

始める前に

次のソフトウェアとハードウェア コンポーネントが Cisco UCS サーバにインストールされていることを確認してください。

- サポートされている Linux オペレーティング システム ディストリビューション リリース。詳細については、「UCS ハードウェア/ソフトウェア互換性ツール」を参照してください。
- GCC、G++、および Gfortran
- DAT ユーザ ライブラリ（Intel ® MPI を使用している場合）
- libnl ユーザ ライブラリ開発パッケージ（バージョン 1 とバージョン 3 のどちらか）
- 第二世代以降の Cisco UCS VIC アダプタ

手順

ステップ 1 カーネルでカーネル オプション **CONFIG_INTEL_IOMMU** が選択されていることを確認します。手動で grub.conf ファイル（たとえば、`/boot/grub/grub.conf`）に「**intel_iommu=on**」を追加することによって、Linux カーネル内で Intel IOMMU ドライバを有効にします。

たとえば、`kernel (hd0,0)/vmlinuz LANG=en_US.UTF-8 KEYTABLE=us` のように、grub.conf ファイルに「kernel」行が含まれている場合、「**intel_iommu=on**」を次に示すように最後に追加します。

```
kernel (hd0,0)/vmlinuz LANG=en_US.UTF-8 KEYTABLE=us intel_iommu=on
```

Red Hat Enterprise Linux の場合、intel_iommu を設定ファイルに追加するために、grubby コマンド ライン ツールを使用してください。

```
# grubby --args="intel_iommu=on" --update-kernel /boot/vmlinuz-`uname -r`
```

SLES の場合、「intel_iommu=on」を /etc/default/grub configuration file の GRUB_CMDLINE_LINUX_DEFAULT オプションに追加し、変更を適用するために、grub2 mkconfig コマンドを実行します。

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

Ubuntu の場合、「intel_iommu=on」を /etc/default/grub configuration file の GRUB_CMDLINE_LINUX_DEFAULT オプションに追加し、変更を適用するために、update-grub コマンドを実行します。

```
# update-grub2
```

ステップ 2 Cisco UCS サーバを再起動します。

Intel IOMMU を有効にした後で、変更を反映するためにサーバを再起動します。

ステップ 3 実行中のカーネルが intel_iommu=on オプションでブートすることを確認します。

```
$ cat /proc/cmdline | grep iommu
```

ステップ 4 Cisco usNIC Linux ドライバをインストールします。

ドライバのインストールの詳細については、「Linux ドライバのインストール」のセクションを参照してください。

（注） Cisco usNIC パッケージは、オペレーティング システムのアップグレードまたはダウングレードをサポートしません。オペレーティングシステムを更新するには、まず、usNICパッケージをアンインストールして、オペレーティングシステムを更新してから、usNIC ドライバを再インストールします。

または、オペレーティングシステムを更新して、usNICドライバをアンインストールしてから、usNIC ドライバを再インストールすることができます。

Linux ソフトウェア パッケージのインストール Cisco usNIC

ここでは、UCS Drivers ISO バンドルに含まれているサポート対象の Linux オペレーティング システム ディストリビューションごとに usNIC フォルダの内容を列挙します。既知の問題やインストール手順に関するドキュメントが usNIC フォルダ内の README ファイルにも含まれています。

- **kmod-usnic_verbs-{version}.x86_64.rpm**—Cisco VIC SR-IOV Ethernet NIC の usNIC 機能のための Linux カーネル verbs ドライバ。

（注） システム SLES 12.1 以降では、cisco-enic-usnic-kmp-default-{version}.x86_64.rpm という名前の enic と usnic_verbs の両方のドライバを含む単一の RPM があります。SLES カーネル モジュールの依存関係のしくみのため、この「コンボ」RPM は個々の enic および usnic_verbs RPM をインストールする代わりにインストールする必要があります。

- **libdaplusnic-{version}.x86_64.rpm**—usNIC のユーザ スペース ライブラリ DAPL プラグイン。
- **usnic_tools-{version}.x86_64.rpm** —usNIC のユーティリティ プログラム。
- **usd_tools-{version}.x86_64.rpm** —usNIC の追加診断ツール。
- **libfabric-cisco-{version}.x86_64.rpm**—Cisco usNIC トランスポートの組み込みサポートをもつ Libfabric パッケージ。

- **libfabric-cisco-devel-{version}.x86_64.rpm** —Cisco usNIC トランスポートの組み込みサポートをもつ Libfabric パッケージの開発ヘッダ。
- **libusnic_verbs-{version}.x86_64.rpm**— libibverbs ライブラリに Cisco usNIC Linux デバイスをスキップさせるダミー ライブラリ（Cisco usNIC 機能は libibverbs ではなく、libfabric を介して公開されるため）。この RPM は libibverbs ライブラリ (RHEL 6 など) の「rdma-core」のパッケージにアップグレードされていない古い Linux ディストリビューション用でのみ必要です。

手順

ステップ 1 「Linux の Cisco UCS 仮想インターフェイス カード ドライバのインストール ガイド」 で記載されているように、Linux ディストリビューション用の Cisco UCS Drivers ISO に含まれている enic ドライバの最新バージョンにアップグレードします。

ステップ 2 Linux ディストリビューション用の Cisco UCS Drivers ISO から Cisco usNIC ソフトウェア パッケージをインストールします。

ステップ 3 Linux RDMA サービスを有効にします。有効にしたら、システムのリブート後に RDMA サービスが自動的に開始されます。Linux RDMA を有効にする正確なコマンドは、各オペレーティング システムの間で異なる場合があります。たとえば、Red Hat Enterprise Linux 6.X で、次のコマンドを使用します。

```
# chkconfig rdma on
```

（注） RHEL 6.4 などの一部の Linux オペレーティング システム ディストリビューションでこの手順を実行しなければならない場合があります。

ステップ 4 サーバをリブートすると、インストールの変更が自動的に反映されます。この時点で再起動して、次のインストールの検証手順を実行することにより、システムが正しく設定されていることを確認できます。

重要 ISO からすべてのバイナリ RPM をインストールすることをお勧めします。ソース コード パッケージの構築とインストールは、Linux デバイス ドライバ開発環境に精通している高度なユーザを対象にしてます。

Linux Cisco usNIC ソフトウェア パッケージのソース コード

Cisco usNIC ソフトウェア パッケージのソース コードは Cisco UCS Drivers ISO で提供されます。ソース コードとバイナリ パッケージのインストールを混在させないことをお勧めします。

カーネル モジュールの手動ロード Cisco usNIC

オペレーティング システムが Intel IOMMU サポートを有効にして起動されたことを前提として、次の手順で、Cisco usNIC カーネル モジュールを手動でロードすることができます。

始める前に

最新バージョンのドライバをロードする前に、既存バージョンのドライバが削除されていることを確認してください。これで、システムを正しく設定できます。

手順

	コマンドまたはアクション	目的
ステップ 1	# **rmmod enic**	既存の enic ドライバ モジュールをアンロードします。 （注） VIC IP インターフェイスに SSH を経由するなど、enic ドライバを使用して OS にログインしていることを確認してください。enic ドライバのロードが解除されるときに、ネットワーク接続が切断される可能性があるからです。代わりに、KVM、経由でまたはLOM IP インターフェイス経由では、コンソールを使用してサーバにログインする必要があります。
ステップ 2	# **modprobe enic**	enic ドライバ モジュールをロードします。
ステップ 3	# **dmesg | grep 'Cisco VIC Ethernet NIC Driver'**	enic ドライバの適切なバージョンがロードされたことを確認します。このコマンドの出力は、インストールしたばかりの Cisco UCS ドライバ ISO から enic RPM のバージョンと一致するバージョン文字列を表示するはずです。
ステップ 4	# **modprobe usnic_verbs**	usnic_verbs ドライバ モジュールをロードします。
ステップ 5	# **lsmod | grep usnic_verbs**	usnic_verbs モジュールが正常にロードされたことを確認します。これは、一部の出力が表示されます。usnic_verbs モジュールが読み込まれなかった場合、出力は表示されません。

Linux ソフトウェア パッケージのアンインストール Cisco usNIC

手順

ステップ 1 次の usNIC ソフトウェア パッケージをアンインストールします。

- libusnic_verbs (必要に応じて)
- libfabric-cisco-devel
- libfabric-cisco
- usd_tools
- usnic_tools
- libdaplusnic
- kmod usnic_verbs (または cisco-enic-usnic-kmp-default)

ステップ 2 Cisco サーバをリブートします。

Cisco usNIC 用の Linux ソフトウェア パッケージのアップグレード

手順

ステップ 1 「Linux ドライバのアンインストール」の手順に従って、usNIC ソフトウェア パッケージの以前のバージョンをアンインストールします。

ステップ 2 「Linux ドライバのインストール」の手順に従って、Linux ディストリビューション用の Cisco UCS ドライバ ISO から usNIC ソフトウェア パッケージをインストールします。

MPI のインストール

Community Open MPI のインストール

始める前に

Install the kmod-usnic_verbs, libfabric-cisco, and libfabric-cisco-devel RPMs.

手順

ステップ 1 https://www.open-mpi.org/software/ompi/current/ からの Open MPI の最新バージョンをダウンロードします。この URL では最新のリリースに自動的にリダイレクトされます。

ステップ 2 次のコマンドで Open MPI tarball を抽出します。

```
$ tar xf openmpi-VERSION.tar.bz2
```

ステップ 3 作成されたディレクトリでは、次のオプションを使用して、configure コマンドを実行します。

```
$ cd openmpi-VERSION
$ ./configure \
 --prefix=INSTALL_DIRECTORY \
 --with-usnic \
 --with-libfabric=/opt/cisco/libfabric \
 --without-memory-manager \
 --enable-mpirun-prefix-by-default \
 --enable-mca-no-build=btl-openib,common-verbs,oob-ud \
 LDFLAGS="-Wl,-rpath -Wl,/opt/cisco/libfabric/lib -Wl,--enable-new-dtags"
```

「INSTALL_DIRECTORY」(たとえば、/opt/openmpi/VERSION) を適切なディレクトリを置き換えてください。

（注） (以前のバージョンの libibverbs ライブラリでは、verbs のサポートなしで Open MPI を構築します (以前のバージョンの libibverbs ライブラリは、usNIC デバイスについての stderr への必要のない警告を発するためです)。Linux verbs API をサポートするために Open MPI が必要な場合、--enable-mca-no-build オプションを削除することができます。。

ステップ 4 設定の正常な結論では、Open MPI を構築します。

```
$ make -j
```

ステップ 5 正常な結論を作成するときに、Open MPI をインストールします (ルートまたは特定ユーザの選択した INSTALL_DIRECTORY に書き込む権限が必要です)。

```
$ make install
```

IBM スペクトル MPI のインストール

次のウェブ サイトにある IBM Spectrum MPI パッケージに記載された手順に従います。

手順

- https://www.ibm.com/support/knowledgecenter/SSZTET_10.1.0/smpi_welcome/smpi_kc_install.html

Intel MPI ライブラリのインストール

次のウェブ サイトの Intel MPI パッケージに記載された手順に従います。

手順

- https://software.intel.com/en-us/intel-mpi-library/documentation

ユーザ環境への MPI の追加

MPIアプリケーションをコンパイルして起動するには、MPI実装を各ユーザの環境に追加する必要があります。MPI 実装は一度に 1 つずつユーザの環境に追加することをお勧めします。

IBM Spectrum MPI ライブラリの環境

Spectrum MPIでは、環境のセットアップの観点からはほとんど必要ありません。絶対パスを通じるか、パスに Spectrum MPI コマンドでディレクトリを追加することにより、Spectrum MPI のコマンドを呼び出し、それらを絶対パスなしで呼び出すことができます。

Community Open MPI の環境

Community Open MPI では、その installation binary と library パスを環境変数に追加する必要があります。シェル スタートアップ ファイルにこれらのパスを追加し、クラスタでのすべてのノードのログイン時に自動的にセットアップされるようにすることが通常は最善です。

特に、INSTALL_DIRECTORY/bin で PATH 環境変数をプリペンドし、INSTALL_DIRECTORY/lib に LD_LIBRARY_PATH 環境変数をプレフィックスする必要があります。任意に、INSTALL_DIRECTORY/share/man を MANPATH 環境変数に追加することもできます。

たとえば、INSTALL_DIRECTORY of /opt/openmpi で Community Open MPI を設定した場合、Bash をログイン シェルとして使用すると、これらの行をシェル スタートアップ ファイル (一般的には $HOME/.bashrc) に追加できます。

```
export PATH=/opt/openmpi/bin:$PATH
export LD_LIBRARY_PATH=/opt/openmpi/lib:$LD_LIBRARY_PATH
export MANPATH=/opt/openmpi/share/man:$MANPATH
```

代わりに、C シェルをログイン シェルとして使用している場合、これらの行をシェル スタートアップ ファイル (一般的には $HOME/.cshrc) に追加できます。

```
set path=(/opt/openmpi/bin $path)
setenv LD_LIBRARY_PATH /opt/openmpi/lib:$LD_LIBRARY_PATH
if ("1" == "$?MANPATH") then
    setenv MANPATH /opt/openmpi/share/man:${MANPATH}
else
    setenv MANPATH /opt/openmpi/share/man:
endif
```

お使いのシステムでは若干異なるコマンドが必要になる場合があります。PATH、LD_LIBRARY_PATH、MANPATH 環境変数を設定する方法についての詳細は、「MPI ジョブの実行」の Open MPI Community FAQ (https://www.open-mpi.org/faq/) を確認してください。

Intel ® MPI ライブラリの環境

Intel ® MPI ライブラリのドキュメントに記載されている手順に加えて、各ユーザの環境で追加の環境変数を設定して Cisco usNIC 機能を有効にする必要があります。必要な環境変数を設定するための 2 つのスクリプトが libdaplusnic ソフトウェア パッケージを通してインストールされます。1 つのスクリプトは Bourne シェル ユーザ用で、もう 1 つのスクリプトは C シェル ユーザ用です。

- /opt/cisco/intelmpi-usnic-vars.sh (Bourne シェル ユーザの場合)
- /opt/cisco/intelmpi-usnic-vars.csh (C シェル ユーザの場合)

ユーザのシェル起動/ログイン シーケンスの一部として適切なスクリプトを調達する必要があります。

Intel® MPI ライブラリと usNIC を使用するには、ネットワーク上のフロー制御を有効に設定する必要があります。これは、IEEE 802.3x リンク レベル フロー制御と IEEE 802.1Qbb 優先順位ベース フロー制御（PFC）のどちらかにすることができます。この機能は、「no-drop」と呼ばれることもあります。フロー制御の有効化方法については、ネットワーク上のスイッチのコンフィギュレーションガイドを参照してください。フロー制御がネットワーク上で有効になっていない場合は、Intel® MPI ライブラリを使用するアプリケーションが正常に機能しますが、パフォーマンスが大幅に低下する可能性があります。

Intel® MPI ライブラリの開発では、MPI トラフィックは Cisco CIMC でサービス クラス(CoS) 6 をもつすべてのすべての Cisco usNIC ポートでフロー制御が有効になっている必要があります。

ユーザ環境への Libfabric の追加

Libfabric 指定のアプリケーションを開発中の場合、Libfabric テスト実行ファイル (fi_pingpong など) および/またはお使いの環境での man ページを利用できます。必要な環境変数を設定するための 2 つのスクリプトが libdaplusnic を通してインストールされます。1 つのスクリプトは Bourne シェル ユーザ用で、もう 1 つのスクリプトは C シェル ユーザ用です。

- /opt/cisco/libfabric-vars.sh
- /opt/cisco/libfabric-vars.csh

ユーザのシェル起動/ログイン シーケンスの一部として適切なスクリプトを調達する必要があります。

ユーザ環境への usNIC ツールの追加

環境に usNIC ツールを追加することは、次のスクリプトにより行うことができます。最初は Bourne シェル ユーザ用で、2 番目のものは C シェル ユーザ用です。

- /opt/cisco/usnic/bin/usnic-vars.sh
- /opt/cisco/usnic/bin/usnic-vars.csh

Cisco UCS C シリーズ ラックマウント スタンドアロン サーバ用の Cisco usNIC のインストールの確認

Cisco usNIC に必要な Linux ドライバをインストールした後、Linux プロンプトで次の手順を実行して、インストールが正常に完了したことを確認してください。

（注） 次に、Linux オペレーティング システム ディストリビューション RHEL 6.5 上で確認された設定例を示します。

手順

ステップ 1 **usnic_verbs** カーネル モジュールが OS ドライバのインストール中にロードされたかどうかを検索して確認します。

```
$ lsmod | grep usnic_verbs
```

lsmod | grep usnic_verbs コマンドを入力すると、次の詳細が表示されます。コンソールに一覧表示されているカーネル モジュールは、OS に現在ロードされているモジュールに基づいて異なる場合があります。

```
usnic_verbs            73762  2
ib_core                74355  2 ib_uverbs,usnic_verbs
enic                   73723  1 usnic_verbs
```

ステップ 2 Cisco usNIC 対応 NIC の設定を確認してください。

```
$ /opt/cisco/usnic/bin/usnic_devinfo
```

次のセクションは、**usnic_devinfo** コマンドを実行すると表示される結果の簡単な例です。結果は、現在のインストール環境によって異なる場合があります。結果がコンソールに表示されたら、一覧表示されたポートのそれぞれのリンク状態が UP と表示されていることを確認します。

次に、Cisco UCS VIC アダプタ上で設定された 2 つのポート（**usnic_1** および **usnic_0**) の例を示します。1 つの Cisco usNIC 対応 vNIC だけを設定した場合は、**usnic_0** のみのリストが表示されます。

```
usnic_0:
        Interface:              eth3
        MAC Address:            00:25:b5:31:32:10
        IP Address:             10.10.10.2
        Netmask:                255.255.255.0
        Prefix len:             24
        MTU:                    9000
        Link State:             UP
        Bandwidth:              40 Gb/s
        Device ID:              UCSC-MLOM-C40Q-03 [VIC 1387] [0x015d]
        Vendor ID:              4407
        Vendor Part ID:         207
        Firmware:               4.1(3S1)
        VFs:                    58
        CQ per VF:              6
        QP per VF:              6
        Interrupts per VF:      6
        Max CQ:                 348
        Max CQ Entries:         65535
        Max QP:                 348
        Max Send Credits:       4095
        Max Recv Credits:       4095
```

```
        Capabilities:
          Map per res:          yes
          PIO sends:            yes
          CQ interrupts:        no
usnic_1:
        Interface:              eth4
        MAC Address:            00:25:b5:31:32:20
        IP Address:             10.20.10.2
        Netmask:                255.255.255.0
        Prefix len:             24
        MTU:                    9000
        Link State:             UP
        Bandwidth:              40 Gb/s
        Device ID:              UCSC-MLOM-C40Q-03 [VIC 1387] [0x015d]
        Vendor ID:              4407
        Vendor Part ID:         207
        Firmware:               4.1(3S1)
        VFs:                    58
        CQ per VF:              6
        QP per VF:              6
        Interrupts per VF:      6
        Max CQ:                 348
        Max CQ Entries:         65535
        Max QP:                 348
        Max Send Credits:       4095
        Max Recv Credits:       4095
        Capabilities:
          Map per res:          yes
          PIO sends:            yes
          CQ interrupts:        no
```

ステップ 3 インストールされている RPM とそのバージョンを確認するには、**usnic_check** スクリプトを実行します。

```
$ /opt/cisco/usnic/bin/usnic_check
```

（注） Cisco usNIC のソフトウェア パッケージのソース コードからすべてのコンポーネントをインストールする場合、usnic_check スクリプトが不足する RPM を報告します。ソース コードとバイナリ パッケージのインストールを混在させないことをお勧めします。

ステップ 4 Cisco usNIC ネットワーク パケットがクライアントとサーバ ホスト間で正常に送信されていることを確認します。

a) サーバ ホストの Cisco usNIC に関連付けられているイーサネット インターフェイスの名前を判別します。

```
[server]$ /opt/cisco/usnic/bin/usnic_status
usnic_0: 0000:07:0.0, eth3, 00:25:b5:31:32:10, 58 VFs
 Per VF: 6 WQ, 6 RQ, 6 CQ, 4 INT

In use:
0 VFs, 0 QPs, 0 CQs


usnic_1: 0000:0c:0.0, eth4, 00:25:b5:31:32:20, 58 VFs
 Per VF: 6 WQ, 6 RQ, 6 CQ, 4 INT

In use:
0 VFs, 0 QPs, 0 CQs
```

b) イーサネット インターフェイスの IP アドレスを判別します。

```
[server]$ ip addr show dev eth1 | grep "inet[^6]"
    inet 10.10.10.2/24 brd 50.42.110.255 scope global eth1
```

c) サーバ ホストで **fi_pingpong** プログラムを実行します。

```
[server]$ /opt/cisco/libfabric/bin/fi_pingpong -p usnic
```

usd_pingpong プログラムで使用されるコマンド ライン オプションの詳細については、**fi_pingpong --help** の出力を参照してください。

d) サーバ ホストの Cisco usNIC に対応する IP アドレスを使用して、クライアント ホストで **fi_pingpong** プログラムを実行します。

```
[client]$ /opt/cisco/libfabric/bin/fi_pingpong -p usnic SERVER_IP_ADDRESS
```

次に、**fi_pingpong** プログラムを実行すると表示される結果の例を示します。

```
Server-side:
[server]$ /opt/cisco/libfabric/bin/fi_pingpong
bytes   #sent   #ack     total       time     MB/sec    usec/xfer   Mxfers/sec
64      10k     =10k     1.2m        0.07s     17.84       3.59       0.28
256     10k     =10k     4.8m        0.08s     66.23       3.87       0.26
1k      10k     =10k     19m         0.10s    199.76       5.13       0.20
4k      10k     =10k     78m         0.18s    466.60       8.78       0.11

Client-side:
[client]$ /opt/cisco/libfabric/bin/fi_pingpong -p usnic SERVER_IP_ADDRESS
bytes   #sent   #ack     total       time     MB/sec    usec/xfer   Mxfers/sec
64      10k     =10k     1.2m        0.07s     17.84       3.59       0.28
256     10k     =10k     4.8m        0.08s     66.25       3.86       0.26
1k      10k     =10k     19m         0.10s    199.77       5.13       0.20
4k      10k     =10k     78m         0.18s    466.61       8.78       0.11
```

（注） fi_pingpong は高性能なベンチマークではありません。UsNIC デバイスでの一般的なパフォーマンス レベルを示していますが、絶対的な最適なパフォーマンスを表示するような高度な調整は行われていません。fi_pingpong 出力は、期待されるパフォーマンスの一般的範囲内にあることを確認する場合にのみ使用する必要があります。

ステップ 5 **ring_c** テストプログラムをダウンロード、コンパイル、および実行して、MPI トラフィックがクライアントとサーバのホスト間で正しく送信されていることを検証します。

ring_c テスト プログラムは、https://raw.githubusercontent.com/open-mpi/ompi/v2.x/examples/ring_c.cのリンクから取得できます。

次の例は、**wget** ユーティリティを使用して **ring_c** を取得、コンパイル、および実行する方法を示します。また、テスト プログラムの取得および実行のその他の方法を使用できます。

（注） 環境内の単一の MPI 実装セットアップを使用して次のコマンドを実行します。

```
$ wget --no-check-certificate
https://raw.githubusercontent.com/open-mpi/ompi/v2.x/examples/ring_c.c
--2017-03-21 19:46:20--
https://raw.githubusercontent.com/open-mpi/ompi/v2.x/examples/ring_c.c
Resolving raw.githubusercontent.com... 151.101.192.133, 151.101.64.133, 151.101.128.133,
```

```
 ...
Connecting to raw.githubusercontent.com|151.101.192.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2416 (2.4K) [text/plain]
Saving to: ‘ring_c.c’

ring_c.c 100%[======================>] 2.36K --.-KB/s in 0s

2017-03-21 19:46:20 (29.5 MB/s) - ‘ring_c.c’ saved [2416/2416]

$ mpicc ring_c.c -o ring_c
[no output]

# IBM Spectrum MPI:
$ /path/to/mpirun --usnic --host host1,host2 -n 2 ./ring_c

# Community Open MPI:
$ mpirun --mca btl usnic,vader,self --host host1,host2 -n 2 ./ring_c

# The expected output from both IBM Spectrum MPI and Community Open MPI is:
Process 0 sending 10 to 1, tag 201 (4 processes in ring)
Process 0 sent to 1
Process 0 decremented value: 9
Process 0 decremented value: 8
Process 0 decremented value: 7
Process 0 decremented value: 6
Process 0 decremented value: 5
Process 0 decremented value: 4
Process 0 decremented value: 3
Process 0 decremented value: 2
Process 0 decremented value: 1
Process 0 decremented value: 0
Process 0 exiting
Process 2 exiting
Process 1 exiting
Process 3 exiting ...
```

（注） 必要に応じて、環境内に別のMPI実装をセットアップして、**mpicc**コマンドと**mpirun**コマンドを再実行し、Cisco usNIC 機能でその MPI 実装を確認します。

fi_pingpong プログラムおよび **ring_c**プログラムが正常に実行された場合、Cisco usNIC 上で MPI アプリケーションを実行できるはずです。

トラブルシューティング情報

問題

usnic_check を使用してインストール済み RPM のリストを表示すると、次のことが発生します。

1. 「**No usnic devices found**」などの警告。
2. 「**usnic_verbs_xxxx does not match installed version**」などのバージョン ミスマッチ エラー。

考えられる原因

1. Usnic_verbs の前にインストールされているバージョンでは、このエラーが生じる可能性があります。
2. usnic_verbs ドライバは、読み込まれた enic ドライバが usnic_verbs と互換性がない場合に、読み込みに失敗した可能性があります。

ソリューション

この問題は、通常、次の 2 つのいずれかで発生します。

1. usnic_verbs RPM の古いバージョンが引き続きインストールされています。
    1. 次のコマンドを使用して、インストールされているすべてのバージョンを一覧表示します。rpm -qa | grep usnic_verbs
    2. 次のコマンドを使用してすべてのバージョンをアンインストールします。rpm -e
    3. システムを再起動します。
    4. すべての RPM を再インストールします。
2. Linux カーネルで読み込まれた enic ドライバは、このバージョンの usnic_verbs と互換性がありません。
    1. これは、ディストリビューションが提供した enic RPM が取得した usnic_verbs ドライバと同じ UCS ドライバ ISO の enic ドライバの代わりに読み込まれた場合に発生することができます。
    2. 特に、enic と usnic_verbs ドライバは、「一致」する必要があります。一致しない場合、usnic_verbs は enic 記号が不足しているについて、usnic_verbs が「dmesg」メッセージを示して読み込みに失敗する可能性があります。
    3. この場合、UCS のドライバ ISO から、enic と usnic_verbs ドライバが読み込まれたことを確認します。
    4. また、起動時に、正しい enic ドライバ バージョンがロードされていることを確認します: enic バージョン番号をインストールする RPM enic と一致することを確認するには、「dmesg」からの出力を確認します。
    5. 一致しない場合は、depmod 出力を確認するか、新しい initrd を作成するか、その両方を行って、正しい eric ドライバがブートストラップされ、ブート時にカーネルにロードされることを保証します。

問題

fi_pingpong を使用して Cisco usNIC パケットがクライアントとサーバ間で正しく送信されていることを確認すると、次のエラーが発生します。

1. 「No such address or device」エラー次の例を参照してください。

```
$ /opt/cisco/libfabric/bin/fi_pingpong -p usnic
fi_getinfo: -61
```

考えられる原因

1. Cisco usNIC 接続ポリシーが割り当てられていないか、「not set」になっているため、vNIC インターフェイスで設定されません。
2. サーバ側がクライアント側からパケットを受信しません。

ソリューション

1. 有効な Cisco usNIC 接続ポリシーが usNIC 接続ポリシー内で設定され、サービス プロファイル内の vNIC に割り当てられていることを確認します。
2. サーバとクライアントの両方で Cisco usNIC デバイスの IP アドレスが正しく設定されていることを確認します。
3. クライアント ピンポンが Cisco usNIC デバイスの正しいサーバ IP アドレスにパケットを送信しようとしていることを確認します。

問題

`mpirun` を使用して Cisco usNIC トラフィックを処理すると、次のエラーが発生します。

MTU サイズ ミスマッチ エラー次の例を参照してください。

```
Example:
# Enter the command below at the prompt on a single line from mpirun up to Sendrecv.
# The backslash is included here as a line continuation and is not needed when the command
 is
# entered at the prompt.
$ mpirun --host node05,node06 -np 12 --mca btl usnic,vader,self \
--mca btl_usnic_if_include usnic_1 IMB-MPI1 Sendrecv

The MTU does not match on local and remote hosts.  All interfaces on
all hosts participating in an MPI job must be configured with the same
MTU.  The usNIC interface listed below will not be used to communicate
with this remote host.

  Local host:      node05
  usNIC interface: usnic_1
  Local MTU:       8958
  Remote host:     node06
  Remote MTU:      1458
```

考えられる原因

1. MTU サイズが適切な VLAN 上で正しく設定されていません。
2. MTU サイズが QoS 内で正しく設定されていません。

ソリューション

MTU サイズが VLAN と QoS 上で正しく設定されていることを確認します。

「Configuring QoS System Classes with the LAN Uplinks Manager」を参照してください。

問題

Cisco enic ドライバをインストールすると、次の Cisco enic 依存性エラーが発生します。

```
# rpm -ivh kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64.rpm
error: Failed dependencies:
                ksym(enic_api_devcmd_proxy_by_index) = 0x107cb661 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_alloc_discover) = 0xfb7e4707 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_pdev) = 0xae6ae5c9 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_res) = 0xd910c86b is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_res_bar) = 0x31710a7e is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_res_bus_addr) = 0x7be7a062 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_res_count) = 0x759e4b07 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_get_res_type_len) = 0xd122f0a1 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
                ksym(vnic_dev_unregister) = 0xd99602a1 is needed by
kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64
#
```

考えられる原因

1. enic ドライバが正しくインストールされていません。
2. enic ドライバがインストールされていません。

ソリューション

正しい enic ドライバがインストールされていることを確認します。また、次の点を確認します。

- 具体的に、次のことを確認します。enic ドライバおよびusnic_verbs ドライバは一致している必要があります。不一致があると、上記のバージョンエラーが発生する可能性があります。
- 具体的には、Cisco UCS ドライバ ISO に含まれる enic および usnic_verbs が互いに一致している必要があります。ある Cisco UCS ドライバ ISO の enic を使用し、別の Cisco UCS ドライバ ISO の usnic_verbs を使用した場合、上記のバージョン エラーが発生することになります。

（注） SLES 12.1 以降のシステムで、enic と usnic_verbs RPM の「コンボ」をインストールする際に、enic と usnic_verbs ドライバが一致することが保証されます。

問題

Intel IOMMU から次の警告が表示されます。

```
# rpm -ivh kmod-usnic_verbs-1.0.4.318.rhel6u5-1.x86_64.rpm
Preparing...                ########################################### [100%]
   1:kmod-usnic_verbs       ########################################### [100%]
WARNING -
Intel IOMMU does not appear to be enabled - please add kernel parameter
intel_iommu=on to your boot configuration for USNIC driver to function.
#
```

考えられる原因

Intel IOMMU サポートが、Linux カーネルで有効になっていません。

ソリューション

Linux カーネルの Intel IOMMU ドライバを有効にします。

問題

DAT ユーザ ライブラリをインストールすると、次の障害が発生した依存関係エラーが生じます。

```
# rpm -ivh libdaplusnic-2.0.39cisco1.0.0.317-1.el6.x86_64.rpm
error: Failed dependencies:
                dapl is needed by libdaplusnic-2.0.39cisco1.0.0.317-1.el6.x86_64
#
```

考えられる原因

`libdapl` が DAT ライブラリをインストールせずにインストールされています。

ソリューション

DAT ライブラリをインストールします。

問題

`usnic_devinfo` を使用して Cisco usNIC が有効化された VICS の構成を表示すると、コマンド出力が usNIC インターフェイスを一覧表示しません。

考えられる原因

RDMA サービスが有効になっていません。

ソリューション

次のコマンドを使用して RDMA サービスを有効にします。

```
# service rdma start
Or
# chkconfig rdma on
```